МАТЕМАТИКА
В ТЕХНИЧЕСКОМ УНИВЕРСИТЕТЕ

$x_1 = a_{11}^* (a_{12} x_2 + \ldots + a_{1n} x_n), \quad x_n = ((ab)^* + bc)$

XIX

А.И. Белоусов, С.Б. Ткачев

ДИСКРЕТНАЯ МАТЕМАТИКА

Издательство МГТУ имени Н.Э. Баумана

Математика в техническом университете

Выпуск XIX

Серия удостоена Премии Правительства Российской Федерации в области науки и техники за 2003 год

Комплекс учебников из 21 выпуска

Под редакцией В.С. Зарубина и А.П. Крищенко

I. Введение в анализ
II. Дифференциальное исчисление функций одного переменного
III. Аналитическая геометрия
IV. Линейная алгебра
V. Дифференциальное исчисление функций многих переменных
VI. Интегральное исчисление функций одного переменного
VII. Кратные и криволинейные интегралы. Элементы теории поля
VIII. Дифференциальные уравнения
IX. Ряды
X. Теория функций комплексного переменного
XI. Интегральные преобразования и операционное исчисление
XII. Дифференциальные уравнения математической физики
XIII. Приближенные методы математической физики
XIV. Методы оптимизации
XV. Вариационное исчисление и оптимальное управление
XVI. Теория вероятностей
XVII. Математическая статистика
XVIII. Случайные процессы
XIX. Дискретная математика
XX. Исследование операций
XXI. Математическое моделирование в технике

А.И. Белоусов, С.Б. Ткачев

ДИСКРЕТНАЯ МАТЕМАТИКА

Под редакцией
д-ра техн. наук, профессора В.С. Зарубина
и д-ра физ.-мат. наук, профессора А.П. Крищенко

Издание третье, стереотипное

Рекомендовано
Министерством образования
Российской Федерации
в качестве учебника для студентов
высших технических учебных заведений

Москва
Издательство МГТУ им. Н.Э. Баумана
2004

УДК 512.5+519.1(075.8)
ББК 22.174
Б43

Рецензенты: чл.-корр. РАН Ю.Н. Павловский, проф. А.К. Платонов

Б43 **Белоусов А.И., Ткачев С.Б.** Дискретная математика: Учеб. для вузов / Под ред. В.С. Зарубина, А.П. Крищенко. – 3-е изд., стереотип. – М.: Изд-во МГТУ им. Н.Э. Баумана, 2004. – 744 с. (Сер. Математика в техническом университете; Вып. XIX).

ISBN 5-7038-1769-2 (Вып. XIX)
ISBN 5-7038-1270-4

В девятнадцатом выпуске серии „Математика в техническом университете“ изложены теория множеств и отношений, элементы современной абстрактной алгебры, теория графов, классические понятия теории булевых функций, а также основы теории формальных языков, куда включены теории конечных автоматов, регулярных языков, контекстно-свободных языков и магазинных автоматов. В анализе графов и автоматов особое внимание уделено алгебраическим методам.

Содержание учебника соответствует курсу лекций, который авторы читают в МГТУ им. Н.Э. Баумана.

Для студентов технических университетов. Может быть полезен преподавателям, аспирантам и инженерам.

Ил. 200. Табл. 27. Библиогр. 65 назв.

УДК 512.5+519.1(075.8)
ББК 22.174

ISBN 5-7038-1769-2 (Вып. XIX)
ISBN 5-7038-1270-4

К 175-летию
МГТУ им. Н.Э. Баумана

ПРЕДИСЛОВИЕ

Предлагаемая читателю книга является девятнадцатым выпуском комплекса учебников „Математика в техническом университете“. Она содержит систематическое изложение курса дискретной математики.

Развитие классической („непрерывной“) математики было обусловлено прежде всего решением задач естествознания, главным образом физики. „Дискретная“ же математика развивалась в связи с изучением законов и правил человеческого мышления, что и обусловило ее применение в тех областях техники, которые так или иначе связаны с моделированием мышления, и в первую очередь в вычислительной технике и программировании.

Мышление реализует себя прежде всего в языке. Поэтому разумно считать, что ядро дискретной математики образует именно математическая теория языков, точнее, область этой теории, называемая теорией формальных языков. Слово „формальный“ подчеркивает, что в этой теории изучаются в основном искусственные языки, специально созданные для каких-то целей: языки программирования, языки математики и т.п. Теория формальных языков является базой теории кодирования, „криптологии“, изучающей методы защиты информации, теории алгоритмов и в определенном смысле математической логики. В прикладном аспекте эта теория служит основой разработки математического обеспечения вычислительных машин.

Доминирующим в современной теории формальных языков является алгебраический подход, в котором существенно используется аппарат, базирующийся на понятии алгебраической структуры полукольца. Этот аппарат во многом похож на аппарат линейной алгебры. Систематическое изложение теории

формальных языков на базе теории полуколец и является одной из основных задач этой книги. Отметим, что в отечественной учебной литературе такой подход почти не получил отражения.

Теория формальных языков существенно опирается и на теорию графов. Многие задачи теории языков (например, задача определения языка конечного или магазинного автомата) сводится к задаче о путях во взвешенных (размеченных) ориентированных графах, где множество меток имеет алгебраическую структуру полукольца.

Изложение материала построено следующим образом. Глава 1 посвящена множествам и отношениям. Здесь напоминаются основы теории множеств, изложенные в первом выпуске комплекта учебников, причем некоторые вопросы излагаются более детально. Основное содержание главы составляет теория отношений. Центральным результатом является теорема о неподвижной точке для индуктивных упорядоченных множеств, на базе которой строятся методы решения задач о путях в графах и алгебраические методы в теории формальных языков.

Ввиду важности алгебраических методов в дискретной математике большое внимание уделяется алгебраической теории: ей посвящены три главы. В главе 2 излагаются элементы классической общей алгебры и рассматриваются группы, кольца и поля. Глава 3 посвящена полукольцам и булевым алгебрам. Приведенный здесь материал имеет важное значение с точки зрения приложения алгебраических методов как в теории формальных языков, так и в теории булевых функций. Особенностью изложения является определение булевой алгебры как частного случая полукольца. В главе 4 приведены некоторые результаты общей теории алгебраических систем.

Глава 5 посвящена теории графов. Центральное место в главе занимает изложение алгебраического метода решения задач о путях в ориентированных графах, размеченных над полукольцами. Этот материал служит, с одной стороны, иллюстрацией применения алгебраической техники в решении графовых задач, а с другой — основой решения задач в теории формальных

языков. Глава содержит также описание некоторых алгоритмов на графах: алгоритма „поиска в глубину“ и „поиска в ширину“, алгоритма Краскала для отыскания остовного дерева наименьшего веса, алгоритма топологической сортировки. Коротко рассматриваются изоморфизм графов, группы автоморфизмов графов и элементы цикломатики (анализа структуры циклов неориентированного графа).

Глава 6 посвящена классическому разделу дискретной математики — булевым функциям — и включает вопросы минимизации булевых функций и теорему Поста о функциональной полноте.

В главах 7 и 8 изложена теория формальных языков. Глава 7 содержит „линейную часть“ этой теории — теорию конечных автоматов и регулярных языков, а глава 8 — теорию контекстно-свободных языков. Это важнейший класс языков, его теоретический анализ является основой многих информационных технологий, таких, в частности, как проектирование компиляторов или разработка лингвистического обеспечения баз данных. Фундаментальным является понятие магазинного автомата — распознавателя в классе контекстно-свободных языков. Именно эта модель языка служит математической основой конкретных технологий разработки синтаксических анализаторов для языков программирования.

В дополнениях к главе 8 приведены элементарные сведения о синтаксическом анализе контекстно-свободных языков и введение в математическую теорию семантики формальных языков (в частности, языков программирования). Здесь мы пытаемся перекинуть „мостик“ от чистой теории к практической технологии анализа контекстно-свободных языков, используемой прежде всего в компиляторах. Этот материал призван проиллюстрировать связь между изложеной математической теорией и ее приложениями к разработке математического обеспечения компьютеров.

В конце каждой главы помещены задачи для самостоятельного решения. Наиболее трудные задачи снабжены указаниями. В некоторых задачах содержатся и теоретические результаты,

дополняющие основной текст. Часть задач придумана авторами, часть заимствована из других задачников и учебников.

Дискретная математика — бурно развивающаяся область. К сожалению, в этом учебнике мы не нашли возможности даже обзорно изложить некоторые результаты, развивающие классическую теорию графов (гиперграфы, сети Петри, потоковые диаграммы) и теорию языков (сверхъязыки, автоматы над структурами, отличными от слов, теорию алгоритмов как динамических систем, топологические методы в семантике). Мы рекомендуем интересующемуся читателю обстоятельно написанную „Handbook of Theoretical Computer Science“, а также последние выпуски периодического издания „Lecture notes in Computer Science“. Наиболее интересные, с нашей точки зрения, работы из этого издания указаны в списке литературы.

Для успешного освоения материала книги достаточно знания традиционных курсов математического анализа и линейной алгебры, читаемых в техническом университете. Мы в основном опирались на материал, изложенный в выпусках I–IV настоящего комплекса учебников.

В тексте книги имеются ссылки на другие выпуски комплекса учебников. Такой ссылкой служит номер выпуска. Например, [I] означает, что имеется в виду первый выпуск. Ссылки без римских цифр относятся только к этому, девятнадцатому, выпуску. Так, (см. **1.2**) отсылает читателя ко второму параграфу первой главы, а (см. **Д.7.1**) — к первому дополнению седьмой главы этой книги. Ссылки на номера формул и рисунков набраны обычным шрифтом (например, (2.1) — первая формула в главе 2, (рис. 1.5) — пятый рисунок в главе 1).

Большинство используемых в этой книге обозначений помещено в перечне основных обозначений, где наряду с их краткой расшифровкой указаны глава и параграф, в которых можно найти более подробное объяснение по каждому из обозначений. Для части обозначений, введенных в первом выпуске, указаны глава и параграф первого выпуска, а также при необходимости глава и параграф этой книги. Например, **I-1.3, 1.1** показывает, что обозначение введено в третьем параграфе первой главы

первого выпуска и пояснения к нему содержатся в первом параграфе первой главы девятнадцатого выпуска. После этого перечня приведены написание и русское произношение входящих в формулы букв латинского и греческого алфавитов.

В конце книги помещены список рекомендуемой литературы и предметный указатель, в котором расположены в алфавитном порядке (по существительному в именительном падеже) все выделенные в тексте ***полужирным курсивом*** термины с указанием страницы, где они строго определены или описаны.

Выделение термина *светлым курсивом* означает, что этот термин в данном параграфе относится к ключевым словам и читателю должно быть известно его значение. Значение этого термина можно уточнить, найдя с помощью предметного указателя необходимую страницу этого выпуска, на которой термин определен или описан. Если термин введен в другом выпуске, то дана ссылка на этот выпуск (например, III означает ссылку на третий выпуск), а также указана курсивом страница предлагаемой книги, на которой имеются некоторые пояснения к этому термину.

Авторы выражают глубокую благодарность А.А. Кирильченко и М.С. Виноградовой за многочисленные пожелания и замечания, которые были учтены при подготовке книги.

Перед чтением книги в целях самоконтроля предлагается выполнить приведенные ниже задания. В тексте заданий **прямым полужирным** шрифтом выделены термины, значение которых должно быть известно читателю, а в конце каждого задания указана ссылка на номер выпуска, в котором можно найти соответствующие разъяснения. В основном тексте книги эти термины не выделены и не входят в предметный указатель.

Задания для самопроверки

1. Что такое конечное **множество**, **подмножество**, **элемент** множества? Какими способами можно задать множество? Приведите примеры **конечных** и **счетных множеств**. [I]

2. Является ли множество всех **рациональных чисел счетным**? [I]

3. Что такое **множество** всех **действительных чисел**? Что понимают под **расширенной (пополненной) числовой прямой**? [I]

4. Является ли **множество натуральных чисел собственным подмножеством множества целых чисел**? [I]

5. Какие операции над множествами Вы знаете? Перечислите свойства этих операций. [I]

6. В чем заключается **принцип двойственности** для **законов де Моргана**? [I]

7. Из каких этапов состоит доказательство по **методу математической индукции**? [I]

8. Сформулируйте определение **взаимно однозначного отображения** двух множеств. Что такое **тождественное отображение**? Чему равна **композиция прямого и обратного отображений** двух множеств? [I]

9. При каких условиях отображение одного множества в другое называют **сюръекцией, инъекцией и биекцией**? [I]

10. Что называют **неподвижной точкой отображения**? Сколько неподвижных точек у отображения $y = \sin x$? [I]

11. Какие **элементарные функции** Вы знаете? [II]

12. Что такое **область определения и область значения** функции? [I]

13. Приведите примеры функций, **непрерывных** в интервале (a, b). В чем различие между **монотонной и строго монотонной** в некотором промежутке функциями? [I]

14. Что такое **последовательность элементов множества**? [I]

15. Какими свойствами обладает **предел последовательности**? [I]

16. Сформулируйте **признак Вейерштрасса сходимости** ограниченной последовательности. [I]

17. Какова связь между количеством **сочетаний** и количеством **размещений** из n элементов по k? [I]

18. Что такое **единичная** и **нулевая матрицы**? [III]

19. Что такое **диагональная матрица**, **верхняя треугольная** (**нижняя треугольная**) **матрица**? [III]

20. Для матриц каких **типов** (**размеров**) определены операции **сложения** и **умножения**? [III]

21. Что такое **определитель** числовой **квадратной матрицы порядка** n? Как связаны операции **транспонирования** и вычисления **обратной матрицы**? [III]

22. Какую квадратную матрицу называют **вырожденной**, а какую — **невырожденной**? [III]

23. Какие свойства имеют операции **сложения свободных векторов** в пространстве и **умножения вектора на число**? Какими алгебраическими свойствами обладают **скалярное** и **векторное произведения** векторов? [III]

24. Что такое **коллинеарные** и **компланарные** векторы? [III]

25. Что такое **линейное пространство**? Каковы **аксиомы линейного пространства**? Что такое **линейное арифметическое пространство**? [IV]

26. Что такое **размерность линейного пространства** и **базис линейного пространства**? [IV]

27. Что такое **линейный оператор**? [IV]

ОСНОВНЫЕ ОБОЗНАЧЕНИЯ

◀ и ▶ — начало и окончание доказательства

\# — окончание примера или замечания

$a \in A$ — элемент a принадлежит множеству A (множество A содержит элемент a) **I-1.1, 1.1**

$a \notin A$ — элемент a не принадлежит множеству A (множество A не содержит элемент a) **I-1.1, 1.1**

$\{a, b, c\}$ — множество, состоящее из элементов a, b, c **I-1.1, 1.1**

$A = \{x: \ldots\}$ — множество A состоит из элементов x, обладающих свойством, указанным после двоеточия **I-1.1, 1.1**

$\varnothing$ — пустое множество **I-1.1, 1.1**

U — универсальное множество **1.1**

$A = B$ — множества A и B равны **1.1**

$A \subset B$, $B \supset A$ — множество A является подмножеством множества B (A включено в B) **I-1.2, 1.1**

$A \subseteq B$, $B \supseteq A$ — множество A включено в множество B или совпадает с ним, **I-1.2, 1.1**

$A \cap B$ — пересечение множеств A и B **I-1.4, 1.1**

$A \cup B$ — объединение множеств A и B **I-1.4, 1.1**

$\overline{A}$ — дополнение множества A до универсального множества **I-1.4, 1.1**

$A \setminus B$ — разность множеств A и B **I-1.4, 1.1**

$A \triangle B$ — симметрическая разность множеств A и B **I-1.4, 1.1**

— объединение k множеств $A_1, \ldots, A_k$ **I-1.4, 1.5**

— пересечение k множеств $A_1, \ldots, A_k$ **I-1.4, 1.5**

— множество всех подмножеств множества A **1.1**

— символы дизъюнкции и конъюнкции **I-1.5, 1.1**

— булево объединение **3.4**

— булево пересечение **3.4**

— символ импликации **I-1.5, 1.1**

— символ эквивалентности **I-1.5, 1.1**

— отрицание высказывания A **I-1.5, 1.1**

— булево дополнение элемента x **3.4**

— квантор всеобщности ($\forall x$ — для любого x) и квантор существования ($\exists x$ — существует x) **I-1.5, 1.1**

— упорядоченная пара элементов x и y **1.2**

— прямое (декартово) произведение множества X на множество Y **I-2.5, 1.2**

— n-я декартова степень множества X (декартово произведение n экземпляров множества X) **I-2.5**

— число размещений (без повторений) из n элементов по m **I-2.6, 1.9**

— число сочетаний (без повторений) из n элементов по m **I-2.6, 1.9**

— число перестановок из n элементов **I-2.6, 1.9**

— множество натуральных чисел **I-1.3, 1.9**

— множество неотрицательных целых чисел **1.9**

— множество целых чисел **I-1.3**

— множество рациональных чисел **I-1.3**

$\mathbb{R}$ — множество действительных чисел **I-1.3**

$[x, y]$ — замкнутый промежуток (отрезок) **I-1.3**

(x, y) — открытый промежуток (интервал) **I-1.3**

$[x, y)$, $(x, y]$ — полуинтервалы **I-1.3**

$f\colon A \to B$ — отображение (функция) из множества A в множество B **I-2.1, 1.3**

$y = f(x)$ — элемент y есть образ элемента x при отображении f **I-2.1, 1.3**

$f\colon x \mapsto y$ — отображение (функция) переводит элемент x в элемент y, т.е. $y = f(x)$ **1.3**

f^{-1} — отображение, обратное отображению f **I-2.3, 1.3**

$f^{-1}(y)$ — полный прообраз элемента y при отображении f **I-2.1, 1.3**

$f(C)$ — образ множества C при отображении f **I-2.1, 1.3**

$f^{-1}(D)$ — полный прообраз множества D при отображении f **I-2.1, 1.3**

B^A — множество всех отображений из A в B **1.3**

$\rho \circ \sigma$, $f \circ g$ — композиция соответствий ρ и σ, композиция отображений f и g **1.3**

$\rho \subseteq A_1 \times A_2$ — соответствие из множества A_1 в множество A_2 **1.3**

$D(f)$ — область определения отображения f **I-2.1, 1.3**

$D(\rho)$ — область определения соответствия ρ **1.3**

$R(f)$ — область значения отображения f **I-2.1, 1.3**

$R(\rho)$ — область значения соответствия ρ **1.3**

$\rho(x)$ — сечение соответствия ρ **1.3**

ρ^{-1} — соответствие, обратное соответствию ρ **1.3**

$\rho \subseteq A_1 \times \ldots \times A_n$ — n-арное отношение на множествах $A_1, \ldots, A_n$ **1.3**

id_A — диагональ множества A **1.3**

ρ^* — рефлексивно-транзитивное замыкание бинарного отношения ρ **1.6**

$f^n(x)$ — результат n-кратного применения функции f к элементу x, причем $f^0(x) = x$ **1.8**

$\rho|_{C,D}$ — (C,D)-ограничение соответствия ρ **1.4**

$\rho|_C$ — C-сужение соответствия ρ **1.4**

$\rho|_{\circ C}$ — строгое C-сужение соответствия ρ **1.4**

$\rho|_M$ — ограничение бинарного отношения ρ на подмножество M **1.4**

$(A_i)_{i\in I}$ — индексированное семейство множеств (с множеством индексов I) **1.5**

$\bigcup\limits_{i\in I} A_i$ — объединение индексированного семейства множеств **1.5**

$\bigcap\limits_{i\in I} A_i$ — пересечение индексированного семейства множеств **1.5**

$[x]_\rho$ — класс эквивалентности элемента x по отношению эквивалентности ρ **1.7**

A/ρ — фактор-множество множества A по отношению эквивалентности ρ **1.7**

$a = b \pmod k$ — числа a и b равны по модулю k **1.7**

$=_{(\mathrm{mod}\, k)}$ — бинарное отношение равенства по модулю k **1.7**

$\leqslant$, $\preceq$, $\sqsubseteq$, $\preccurlyeq$ — стандартные обозначения различных отношений порядка **1.8**

$\geqslant$, $\succeq$, $\sqsupseteq$, $\succcurlyeq$ — обозначения отношений порядка, двойственных соответственно к $\leqslant$, $\preceq$, $\sqsubseteq$, $\preccurlyeq$ **1.8**

$<$, $\prec$, $\sqsubset$ — обозначения отношений строгого порядка, определяемых соответственно $\leqslant$, $\preceq$, $\sqsubseteq$ **1.8**

$>$, $\succ$, $\sqsupset$ — обозначения отношений строгого порядка, двойственных соответственно к $<$, $\prec$, $\sqsubset$ **1.8**

$\lhd$ — обозначения отношения доминирования, определяемого отношением порядка $\leqslant$ **1.8**

$\sup B$ ($\inf B$) — точная верхняя (точная нижняя) грань множества B **I-2.7, 1.8**

$\sup X_n$ ($\inf X_n$) — точная верхняя (точная нижняя) грань последовательности X_n **1.8**

$\lim\limits_{n\to\infty} x_n$ — предел последовательности x_n при $n \to \infty$ **I-6.3**

$\mathbb{O}$ — наименьший элемент индуктивного частично упорядоченного множества **1.8**

$A \sim B$ — множество A эквивалентно множеству B **1.9**

$|A|$ — мощность множества A **1.9**

$\aleph_0$ — мощность счетного множества **1.9**

c — мощность континуума **1.9**

$\mathbf{0}$ — нуль относительно операции **2.1**

$\mathbf{1}$ — единица относительно операции **2.1**

a^{-1} — элемент, обратный элементу a при мультипликативной записи группы **2.2**

$-a$ — элемент, противоположный элементу a при аддитивной записи коммутативной группы **2.2**

S_n — симметрическая группа степени n (группа подстановок n-элементного множества) **2.2**

aH (Ha) — левый (правый) смежный класс подгруппы H (какой-либо группы G), определяемый элементом $a \in G$ **2.7**

G/H — фактор-группа группы G по нормальной подгруппе H **2.8**

$\mathbb{Z}_k$ — кольцо вычетов по модулю k **2.3**

$\mathbb{Z}_k^+$ — аддитивная группа вычетов по модулю k **2.3**

$\mathbb{Z}_p^*$ — мультипликативная группа вычетов по модулю p (для простого p) **2.3**

$\mathcal{B}$ — полукольцо $(\{0,1\}, +, \cdot, 0, 1)$ (двухэлементное полукольцо) **3.1**

$\mathcal{R}^+$ — полукольцо $(\mathbb{R}^+, \min, +, +\infty, 0)$ (полукольцо неотрицательных действительных чисел вместе с $+\infty$ с операциями взятия наименьшего элемента и сложения) **3.1**

$\mathcal{S}_A$ — полукольцо $(2^A, \cup, \cap, \varnothing, A)$ (полукольцо всех подмножеств множества A) **3.1**

$\mathcal{R}_A$ — полукольцо $(2^{A\times A}, \cup, \circ, \varnothing, \mathrm{id}_A)$ (полукольцо всех бинарных отношений на множестве A) **3.1**

$\mathcal{N}$ — полукольцо $(\mathbb{N}_0, +, \cdot, 0, 1)$ (полукольцо неотрицательных целых чисел с обычными операциями сложения и умножения) **3.1**

$\mathcal{S}_{[a,b]}$ — полукольцо $([a, b] \subset \mathbb{R}, \max, \min, a, b)$ (полукольцо чисел из отрезка числовой прямой с операциями взятия максимума и минимума из двух чисел) **3.1**

$\mathbf{Matr}(\mathcal{S})$ — множество всех матриц с элементами из полукольца $\mathcal{S}$ **3.3**

$\mathrm{M}_n(\mathcal{S})$ — полукольцо квадратных матриц порядка n с элементами из полукольца $\mathcal{S}$ **3.3**

$\mathcal{D}_n$ — полукольцо всех делителей натурального числа n **3.4**

$\sum\limits_{n\in N} x_n$, $\sum x_n$ — точная верхняя грань бесконечной последовательности $x_1, \ldots, x_n, \ldots$ элементов замкнутого полукольца **3.2**

a^* — итерация (замыкание) элемента a замкнутого полукольца (или полукольца с итерацией) **3.2**

$\mathbb{B}$ — двухэлементная булева алгебра (то же, что полукольцо $\mathcal{B}$) **3.4**

$\mathcal{A} = (A, \Omega, \Pi)$ — алгебраическая система с носителем A, сигнатурой, состоящей из множества операций Ω и множества отношений Π **4.1**

$\mathcal{A} = (A, \Omega)$ — алгебра с носителем A и сигнатурой (множеством операций) Ω **4.1**

$\mathcal{A} = (A, \Pi)$ — модель с носителем A и сигнатурой (множеством отношений) Π **4.1**

$[B]_\Omega$ — замыкание множества B по операциям сигнатуры Ω (Ω-замыкание множества B) **4.2**

$h\colon \mathcal{A} \to \mathcal{B}$ — гомоморфизм h алгебраической системы $\mathcal{A}$ в алгебраическую систему $\mathcal{B}$ **4.4**

$\operatorname{Ker} h$ — ядро гомоморфизма h одной алгебраической системы в другую **4.4**

$h(\mathcal{A})$ — гомоморфный образ алгебраической системы $\mathcal{A}$ (относительно гомоморфизма h) **4.4**

$\cong$ — символ изоморфизма алгебраических систем **4.4**

$\mathcal{A}/\rho$ — фактор-система алгебраической системы $\mathcal{A}$ по конгруэнции ρ **4.3**

$u \mapsto_G v$ — вершина u соединена ребром с вершиной v в неориентированном графе G (имя графа часто опускается) **5.1**

$u \to_G v$ — существует дуга в ориентированном графе G с началом u и концом v (имя графа часто опускается) **5.1**

$u \mapsto^*_G v$ — из вершины u (неориентированного графа G) достижима вершина v (имя графа часто опускается) **5.1**

$u \mapsto^+ v$ — существует цепь ненулевой длины, соединяющая вершины u и v (неориентированного графа) **5.1**

$u \mapsto^n v$ — существует цепь длины n, соединяющая вершины u и v (неориентированного графа) **5.1**

$u \Rightarrow^* v$ — из вершины u (некоторого ориентированного графа) достижима вершина v **5.1**

$u \Rightarrow^+ v$ — существует путь ненулевой длины из вершины u в вершину v (в ориентированном графе) **5.1**

$u \Rightarrow^n v$ — существует путь длины n из вершины u в вершину v (в ориентированном графе) **5.1**

$\deg(v)$ — степень вершины v (в неориентированном или ориентированном графе) **5.1**

$\deg^+(v)$ — полустепень исхода вершины v (в ориентированном графе) **5.1**

$\deg^-(v)$ — полустепень захода вершины v (в ориентированном графе) **5.1**

$\Gamma(v)$ — множество всех таких вершин u (ориентированного или неориентированного графа), что $u \to v$ (для ориентированного графа) или $u \mapsto v$ (для неориентированного графа) **5.1**

$\Gamma^{-1}(v)$ — множество всех таких вершин u (ориентированного графа), что $u \to v$ **5.1**

$L[v]$ — список смежности вершины v (в неориентированном или ориентированном графе) **5.2**

$d(v)$, $h(v)$, $l(v)$ — глубина, высота и уровень соответственно узла v дерева **5.3**

h_T — высота дерева T **5.3**

$G_1 \cong G_2$ — графы G_1 и G_2 изоморфны **5.7**

$\overline{G}$ — дополнение графа G **5.7**

V^* — множество всех слов в алфавите V **7.1**

V^+ — множество всех непустых слов в алфавите V **7.1**

V^n — множество всех слов длины n в алфавите V **7.1**

λ — пустое слово **7.1**

$x(i)$ — i-я буква слова x **7.1**

$x(1)x(2)\ldots x(k)$ — побуквенная запись слова x **7.1**

(u, x, v) — вхождение слова x в слово $y = uxv$ **7.1**

$u \sqsubseteq v$ — слово u входит в слово v **7.1**

$L_1 \cdot L_2$ — соединение (конкатенация) языков L_1 и L_2 **7.1**

L^* — итерация языка L **7.1**

L^+ — позитивная итерация языка L **7.1**

$\mathcal{L}(V)$ — полукольцо всех языков в алфавите V **7.1**

$\alpha \to \beta$ — правило вывода (продукция) в грамматике **7.2**

$\gamma \vdash_G \delta$ — цепочка δ непосредственно выводима в грамматике G из цепочки γ (имя грамматики часто опускается) **7.2**

$\gamma \vdash_p \delta$ — цепочка δ непосредственно выводима в грамматике G из цепочки γ применением правила p **7.2**

$\gamma \vdash_G^* \delta$ — цепочка δ выводима в грамматике G из цепочки γ (имя грамматики часто опускается) **7.2**

$\gamma \vdash_G^+ \delta$ — существует вывод ненулевой длины цепочки δ в грамматике G из цепочки γ (имя грамматики часто опускается) **7.2**

$\gamma \vdash_G^n \delta$ — существует вывод длины n цепочки δ в грамматике G из цепочки γ (имя грамматики часто опускается) **7.2**

$\alpha \to \beta_1 | \beta_2 | \ldots | \beta_n$ — запись множества правил грамматики с одной и той же левой частью α **7.2**

$\mathcal{R}(V)$ — полукольцо регулярных языков в алфавите V **7.4**

$(\alpha + \beta)$ — сумма регулярных выражений **7.4**

$(\alpha \cdot \beta)$ — произведение регулярных выражений **7.4**

α^*, α^+ — итерация и позитивная итерация регулярного выражения соответственно **7.4**

$q \to_a r$ — из состояния q конечного автомата возможен переход в состояние r по символу (или пустой цепочке) a **7.5**

$qa \to r$ — запись команды конечного автомата (по символу или пустой цепочке a разрешен переход из состояния q в состояние r) **7.5**

$q \Rightarrow_x^* r$ — цепочка x читается на некотором пути из состояния q в состояние r конечного автомата **7.5**

$L(M)$ — язык конечного автомата M **7.5**

$\cong$ — символ эквивалентности конечных автоматов **7.5**

$qaZ \to r\gamma$ — запись команды МП-автомата (из состояния q с верхним символом в магазине Z по входному символу или пустой цепочке a разрешен переход в состояние r с заменой символа Z цепочкой γ) **8.4**

$\vdash_M, \vdash_M^*, \vdash_M^+, \vdash_M^n$ — отношения непосредственной выводимости, выводимости, выводимости за ненулевое число шагов и выводимости за n шагов соответственно на множестве конфигураций МП-автомата M **8.4**

$L(M)$ — язык МП-автомата M **8.4**

$S(L, L_1, \ldots, L_n)$ — суперпозиция (подстановка) языков L_1, ..., L_n в язык L **8.5**

$A \downarrow\downarrow X_1 X_2 \ldots X_m$ — упорядоченное дерево, корень которого помечен символом A, а листья имеют метки X_1, ..., X_m **8.1**

$A \downarrow X_1 X_2 \ldots X_m$ — куст упорядоченного дерева, корень которого помечен символом A, а листья имеют метки X_1, ..., X_m **8.1**

Буквы латинского алфавита

Начертание	Произношение	Начертание	Произношение
A a *A a*	а	N n *N n*	эн
B b *B b*	бэ	O o *O o*	о
C c *C c*	цэ	P p *P p*	пэ
D d *D d*	дэ	Q q *Q q*	ку
E e *E e*	е	R r *R r*	эр
F f *F f*	эф	S s *S s*	эс
G g *G g*	же	T t *T t*	тэ
H h *H h*	аш	U u *U u*	у
I i *I i*	и	V v *V v*	вэ
J j *J j*	йот	W w *W w*	дубль-вэ
K k *K k*	ка	X x *X x*	икс
L l *L l*	эль	Y y *Y y*	игрек
M m *M m*	эм	Z z *Z z*	зэт

Представлен наиболее употребительный (но не единственный) вариант произношения (в частности, вместо „йот“ иногда говорят „жи“).

Буквы греческого алфавита

Начертание	Произношение	Начертание	Произношение	Начертание	Произношение
A α	альфа	I ι	йота	P ρ	ро
B β	бета	K $\varkappa$	каппа	Σ σ	сигма
Γ γ	гамма	Λ λ	ламбда	T τ	тау
Δ δ	дельта	M μ	ми	Υ υ	ипсилон
E ε	эпсилон	N ν	ни	Φ φ	фи
Z ζ	дзета	Ξ ξ	кси	X χ	хи
H η	эта	O o	омикрон	Ψ ψ	пси
Θ ϑ θ	тэта	Π π	пи	Ω ω	омега

Наряду с указанным произношением также говорят „лямбда“, „мю“ и „ню“.

1. МНОЖЕСТВА И ОТНОШЕНИЯ

1.1. Множества

Элементы теории множеств изложены в [I]. Напомним здесь основные понятия и обозначения.

Понятие множества является исходным не определяемым строго понятием*. Приведем здесь определение множества (точнее, пояснение идеи множества), принадлежащее Г. Кантору**: „Под многообразием или множеством я понимаю вообще все многое, которое возможно мыслить как единое, т.е. такую совокупность определенных элементов, которая посредством одного закона может быть соединена в одно целое".

Множества будем, как правило, обозначать большими буквами латинского алфавита, а их элементы — малыми, хотя иногда от этого соглашения придется отступать, так как элементами некоторого множества могут быть другие множества. Тот факт, что элемент a принадлежит множеству A, записывается в виде $a \in A$.

В математике мы имеем дело с самыми различными множествами. Для элементов этих множеств мы используем два основных вида обозначений: константы и переменные.

Индивидная константа (или просто ***константа***) с областью значений A обозначает фиксированный элемент множества A. Таковы, например, обозначения (записи в опреде-

*Это имеет место в так называемой „наивной" теории множеств. В современной математической литературе фигурируют различные построения теории множеств, в которых понятие множества строго определяется посредством набора аксиом (аксиоматические теории множеств), но при этом используются уже другие неопределимые понятия. В рамках курса дискретной математики нам достаточно ограничиться „наивным" подходом.

**Г. Кантор (1845–1918) — немецкий математик, основатель теории множеств.

ленной системе счисления) действительных чисел: 0; 2; 7,34. Для двух констант a и b с областью значений A будем писать $a = b$, понимая под этим совпадение обозначаемых ими элементов множества A.

Индивидное переменное (или просто ***переменное***) с областью значений A обозначает произвольный, заранее не определенный элемент множества A. При этом говорят, что переменное x пробегает множество A или переменное x принимает произвольные значения на множестве A. Можно фиксировать значение переменного x, записав $x = a$, где a — константа с той же областью значений, что и x. В этом случае говорят, что вместо переменного x подставлено его конкретное значение a, или произведена подстановка a вместо x, или переменное x приняло значение a.

Равенство переменных $x = y$ понимается так: всякий раз, когда переменное x принимает произвольное значение a, переменное y принимает то же самое значение a, и наоборот. Таким образом, равные переменные „синхронно“ принимают всегда одни и те же значения.

Обычно константы и переменные, область значений которых есть некоторое числовое множество [I], а именно одно из множеств $\mathbb{N}$, $\mathbb{Z}$, $\mathbb{Q}$, $\mathbb{R}$ и $\mathbb{C}$, называют соответственно натуральными, целыми (или целочисленными), рациональными, действительными и комплексными константами и переменными. В курсе дискретной математики мы будем использовать различные константы и переменные, область значений которых не всегда является числовым множеством.

Для сокращения записи мы будем пользоваться логической символикой [I], позволяющей коротко, наподобие формул, записывать *высказывания*. Понятие высказывания не определяется. Указывается только, что всякое высказывание может быть истинным или ложным (разумеется, не одновременно!).

Для образования из уже имеющихся высказываний новых высказываний используются следующие ***логические операции*** (или ***логические связки***).

1. *Дизъюнкция* $\vee$: высказывание $P \vee Q$ (читается: „P или Q“) истинно тогда и только тогда, когда истинно хотя бы одно из высказываний P и Q.

2. *Конъюнкция* $\wedge$: высказывание $P \wedge Q$ (читается: „P и Q“) истинно тогда и только тогда, когда истинны оба высказывания P и Q.

3. *Отрицание* $\neg$: высказывание $\neg P$ (читается: „не P“) истинно тогда и только тогда, когда P ложно.

4. *Импликация* $\Rightarrow$: высказывание $P \Rightarrow Q$ (читается: „если P, то Q“ или „P влечет Q“) истинно тогда и только тогда, когда истинно высказывание Q или оба высказывания ложны.

5. *Эквивалентность* (или ***равносильность***) $\Leftrightarrow$: высказывание $P \Leftrightarrow Q$ (читается: „P, если и только если Q“) истинно тогда и только тогда, когда оба высказывания P и Q либо одновременно истинны, либо одновременно ложны. Любые два высказывания P и Q, такие, что истинно $P \Leftrightarrow Q$, называют ***логически эквивалентными*** или ***равносильными***.

Записывая высказывания с помощью логических операций, мы предполагаем, что очередность выполнения всех операций определяется расстановкой скобок. Для упрощения записи скобки зачастую опускают, принимая при этом определенный порядок выполнения операций („соглашение о приоритетах“).

Операция отрицания всегда выполняется первой, и потому ее в скобки не заключают. Второй выполняется операция конъюнкции, затем дизъюнкции и, наконец, импликации и эквивалентности. Например, высказывание $(\neg P) \vee Q$ записывают так: $\neg P \vee Q$. Это высказывание есть дизъюнкция двух высказываний: первое является отрицанием P, а второе — Q. В отличие от него высказывание $\neg(P \vee Q)$ есть отрицание дизъюнкции высказываний P и Q.

Например, высказывание

$$\neg P \wedge Q \vee \neg Q \wedge P \Rightarrow \neg Q$$

после расстановки скобок в соответствии с приоритетами примет вид

$$(((\neg P) \wedge Q) \vee ((\neg Q) \wedge P)) \Rightarrow (\neg Q).$$

Сделаем некоторые комментарии по поводу введенных выше логических связок. Содержательная трактовка дизъюнкции, конъюнкции и отрицания не нуждается в специальных разъяснениях. Импликация $P \Rightarrow Q$ истинна, по определению, всякий раз, когда истинно высказывание Q (независимо от истинности P) или P и Q одновременно ложны. Таким образом, если импликация $P \Rightarrow Q$ истинна, то при истинности P имеет место истинность Q, но обратное может и не выполняться, т.е. при ложности P высказывание Q может быть как истинным, так и ложным. Это и мотивирует прочтение импликации в виде „если P, то Q“. Нетрудно также понять, что высказывание $P \Rightarrow Q$ равносильно высказыванию $\neg P \vee Q$ и тем самым содержательно „если P, то Q“ отождествляется с „не P или Q“.

Равносильность $\Leftrightarrow$ есть не что иное, как „двусторонняя импликация“, т.е. $P \Leftrightarrow Q$ равносильно $(P \Rightarrow Q) \wedge (Q \Rightarrow P)$. Это означает, что из истинности P следует истинность Q и, наоборот, из истинности Q следует истинность P.

Пример 1.1. Для определения истинности или ложности сложного высказывания в зависимости от истинности или ложности входящих в него высказываний используют ***таблицы истинности.***

В первых двух столбцах таблицы записывают все возможные наборы значений, которые могут принимать высказывания P и Q. Истинность высказывания обозначают буквой „И“, а ложность — буквой „Л“. Остальные столбцы заполняют слева направо. Так для каждого набора значений P и Q находят соответствующие значения высказываний.

Наиболее простой вид имеют таблицы истинности логических операций (табл. 1.1–1.5).

Таблица 1.1

P	Q	$P \vee Q$
Л	Л	Л
Л	И	И
И	Л	И
И	И	И

Таблица 1.2

P	Q	$P \wedge Q$
Л	Л	Л
Л	И	Л
И	Л	Л
И	И	И

Таблица 1.3

P	$\neg P$
Л	И
И	Л

Таблица 1.4

P	Q	$P \Rightarrow Q$
Л	Л	И
Л	И	И
И	Л	Л
И	И	И

Таблица 1.5

P	Q	$P \Leftrightarrow Q$
Л	Л	И
Л	И	Л
И	Л	Л
И	И	И

Рассмотрим сложное высказывание

$$(\neg P \wedge Q) \Rightarrow (\neg Q \wedge P).$$

Для удобства вычислений обозначим высказывание $\neg P \wedge Q$ через A, высказывание $\neg Q \wedge P$ через B, а исходное высказывание запишем в виде $A \Rightarrow B$. Таблица истинности этого высказывания состоит из столбцов P, Q, A, B и $A \Rightarrow B$ (табл. 1.6). #

Таблица 1.6

P	Q	A	B	$A \Rightarrow B$
Л	Л	Л	Л	И
Л	И	И	Л	Л
И	Л	Л	И	И
И	И	Л	Л	И

Сложные высказывания образуются не только посредством логических связок, но и с помощью предикатов и кванторов.

Предикат есть высказывание, содержащее одно или несколько индивидных переменных. Например, „x есть четное

число“ или „x есть студент МГТУ им. Баумана, поступивший в 1999 г.“. В первом предикате x есть целочисленное переменное, во втором — переменное, пробегающее множество „человеческих индивидов“. Примером предиката, содержащего несколько индивидных переменных, может служить: „x есть сын y“, „x, y и z учатся в одной и той же группе“, „x делится на y“, „x меньше y“ и т.п. Предикаты будем записывать в виде $P(x)$, $Q(x,y)$, $R(x,y,z)$, полагая, что в скобках перечислены все переменные, входящие в данный предикат.

Подставляя вместо каждого переменного, входящего в предикат $P(x_1,\ldots,x_n)$, конкретное значение, т.е. фиксируя значения $x_1 = a_1,\ldots,x_n = a_n$, где $a_1,\ldots,a_n$ — некоторые константы с соответствующей областью значений, получаем высказывание, не содержащее переменных. Например, „2 есть четное число“, „Исаак Ньютон есть студент МГТУ им. Баумана, поступивший в 1999 г.“, „Иванов есть сын Петрова“, „5 делится на 7“ и т.п. В зависимости от того, истинно или ложно полученное таким образом высказывание, говорят, что предикат P выполняется или не выполняется на наборе значений переменных $x_1 = a_1,\ldots,x_n = a_n$. Предикат, выполняющийся на любом наборе входящих в него переменных, называют ***тождественно истинным***, а предикат, не выполняющийся ни на одном наборе значений входящих в него переменных, — ***тождественно ложным***.

Высказывание из предиката можно получать не только подстановкой значений его переменных, но и посредством кванторов. Вводят два *квантора — существования* и *всеобщности*, обозначаемые $\exists$ и $\forall$ соответственно.

Высказывание $(\forall x \in A)P(x)$ („для каждого элемента x, принадлежащего множеству A, истинно $P(x)$“, или, более коротко, „для всех $x \in A$ истинно $P(x)$“) истинно, по определению, тогда и только тогда, когда предикат $P(x)$ выполняется для каждого значения переменного x.

Высказывание $(\exists x \in A)P(x)$ („существует, или найдется, такой элемент x множества A, что истинно $P(x)$“, также „для

некоторого $x \in A$ истинно $P(x)$“) истинно, по определению, тогда и только тогда, когда на некоторых значениях переменного x выполняется предикат $P(x)$.

При образовании высказывания из предиката посредством квантора говорят, что переменное предиката связывается квантором. Аналогично связываются переменные в предикатах, содержащих несколько переменных. В общем случае используют формы высказываний вида

$$(Q_1 x_1 \in A_1)(Q_2 x_2 \in A_2)\dots(Q_n x_n \in A_n)P(x_1, x_2, \dots, x_n),$$

где вместо каждой буквы Q с индексом может быть подставлен любой из кванторов $\forall$ или $\exists$.

Например, высказывание $(\forall x \in A)(\exists y \in B)P(x,y)$ читается так: „для всякого $x \in A$ существует $y \in B$, такой, что истинно $P(x,y)$“. Если множества, которые пробегают переменные предикатов, фиксированы (подразумеваются „по умолчанию“), то кванторы записываются в сокращенной форме: $(\forall x)P(x)$ или $(\exists x)P(x)$.

Заметим, что многие математические теоремы можно записать в форме, подобной только что приведенным высказываниям с кванторами, например: „для всех f и для всех a истинно: если f — функция, дифференцируемая в точке a, то f непрерывна в точке a“.

Обсудив особенности употребления логической символики, вернемся к рассмотрению множеств.

Два множества A и B считают ***равными***, если любой элемент x множества A является элементом множества B и наоборот. Из приведенного определения равных множеств следует, что множество полностью определяется своими элементами.

Рассмотрим способы задания конкретных множеств. Для конечного множества, число элементов которого относительно невелико, может быть использован способ непосредственного перечисления элементов. Элементы конечного множества пере-

числяют в фигурных скобках в произвольном фиксированном порядке $\{1, 3, 5\}$. Подчеркнем, что поскольку множество полностью определено своими элементами, то при задании конечного множества порядок, в котором перечислены его элементы, не имеет значения. Поэтому записи $\{1, 3, 5\}$, $\{3, 1, 5\}$, $\{5, 3, 1\}$ и т.д. все задают одно и то же множество. Кроме того, иногда в записи множеств используют повторения элементов. Будем считать, что запись $\{1, 3, 3, 5, 5\}$ задает то же самое множество, что и запись $\{1, 3, 5\}$.

В общем случае для конечного множества используют форму записи $\{a_1, \ldots, a_n\}$. Как правило, при этом избегают повторений элементов. Тогда конечное множество, заданное записью $\{a_1, \ldots, a_n\}$, состоит из n элементов. Его называют также ***n-элементным множеством.***

Однако способ задания множества путем непосредственного перечисления его элементов применим в весьма узком диапазоне конечных множеств. Наиболее общим способом задания конкретных множеств является указание некоторого свойства, которым должны обладать все элементы описываемого множества, и только они.

Эта идея реализуется следующим образом. Пусть переменное x пробегает некоторое множество U, называемое ***универсальным множеством.*** Мы предполагаем, что рассматриваются только такие множества, элементы которых являются и элементами множества U. В таком случае свойство, которым обладают исключительно элементы данного множества A, может быть выражено посредством предиката $P(x)$, выполняющегося тогда и только тогда, когда переменное x принимает произвольное значение из множества A. Иначе говоря, $P(x)$ истинно тогда и только тогда, когда вместо x подставляется индивидная константа $a \in A$.

Предикат P называют в этом случае ***характеристическим предикатом*** множества A, а свойство, выражаемое с помощью этого предиката, — ***характеристическим свойством*** или ***коллективизирующим свойством*** [I].

Множество, заданное через характеристический предикат, записывается в следующей форме:

$$A = \{x\colon P(x)\}. \tag{1.1}$$

Например,

$$A = \{x\colon x \text{ есть четное натуральное число}\}$$

означает, что „A есть множество, состоящее из всех таких элементов x, что каждое из них есть четное натуральное число".

Термин „коллективизирующее свойство" мотивирован тем, что это свойство позволяет собрать разрозненные элементы в единое целое. Так, свойство, определяющее множество

$$G = \{x\colon x \text{ есть студент 2-го курса специальности ИУ5 МГТУ им. Баумана, поступивший в 1999 г.}\},$$

в буквальном смысле слова формирует некий „коллектив".

Если мы вернемся к канторовскому определению множества, то характеристический предикат множества и есть тот закон, посредством которого совокупность элементов соединяется в единое целое. Предикат, задающий коллективизирующее свойство, может быть тождественно ложным. Множество, определенное таким образом, не будет иметь ни одного элемента. Его называют *пустым множеством* и обозначают $\varnothing$.

В противоположность этому тождественно истинный характеристический предикат задает универсальное множество.

Обратим внимание на то, что не каждый предикат выражает какое-то коллективизирующее свойство (см. **Д.1.1**).

Замечание 1.1. Конкретное содержание понятия универсального множества определяется тем конкретным контекстом, в котором мы применяем теоретико-множественные идеи. Например, если мы занимаемся только различными числовыми

множествами, то в качестве универсального может фигурировать множество $\mathbb{R}$ всех действительных чисел. В каждом разделе математики рассматривается относительно ограниченный набор множеств. Поэтому удобно полагать, что элементы каждого из этих множеств суть также и элементы некоторого „объемлющего“ их универсального множества. Зафиксировав универсальное множество, мы тем самым фиксируем область значений всех фигурирующих в наших математических рассуждениях переменных и констант. В этом случае как раз и можно не указывать в кванторах то множество, которое пробегает связываемое квантором переменное. В дальнейшем изложении мы встретимся с разными примерами конкретных универсальных множеств. #

Рассмотрим операции над множествами, которые позволяют из уже имеющихся множеств образовывать новые множества [I].

Для любых двух множеств A и B определены новые множества, называемые *объединением*, *пересечением*, *разностью* и *симметрической разностью*:

$$A \cup B = \{x\colon x \in A \vee x \in B\},$$
$$A \cap B = \{x\colon x \in A \wedge x \in B\},$$
$$A \setminus B = \{x\colon x \in A \wedge x \notin B\},$$
$$A \triangle B = (A \setminus B) \cup (B \setminus A),$$

т.е. объединение A и B есть множество всех таких x, что x является элементом хотя бы одного из множеств A, B; пересечение A и B — множество всех таких x, что x — одновременно элемент A и элемент B; разность $A \setminus B$ — множество всех таких x, что x — элемент A, но не элемент B; симметрическая разность $A \triangle B$ — множество всех таких x, что x — элемент A, но не элемент B или x — элемент B, но не элемент A.

Кроме того, фиксируя универсальное множество U, мы можем определить *дополнение* $\overline{A}$ *множества* A следующим

образом: $\overline{A} = U \setminus A$ [I]. Итак, дополнение множества A — это множество всех элементов универсального множества, не принадлежащих A.

Полезно разобраться в том, как операции над множествами, введенные выше, соотносятся с логическими операциями. Пусть $A = \{x\colon P(x)\}$, $B = \{x\colon Q(x)\}$, т.е. множество A задано посредством характеристического предиката P, а множество B — посредством характеристического предиката Q.

Легко показать, что

$$A \cup B = \{x\colon P(x) \vee Q(x)\},$$
$$A \cap B = \{x\colon P(x) \wedge Q(x)\},$$
$$A \setminus B = \{x\colon P(x) \wedge \neg Q(x)\}.$$

Следующие процедуры получения новых множеств связаны с понятием подмножества. Говорят, что B есть подмножество множества A, если всякий элемент B есть элемент A. Для обозначения используют запись: $B \subseteq A$. Говорят также, что B содержится в A, B включено в A, A включает B (имеет место ***включение*** $B \subseteq A$). Считают, что пустое множество есть подмножество любого множества и, если фиксировано некоторое универсальное множество, каждое рассматриваемое множество есть его подмножество. Нетрудно проверить, что если $A = \{x\colon P(x)\}$, $B = \{x\colon Q(x)\}$, то $B \subseteq A$ тогда и только тогда, когда высказывание $Q(x) \Rightarrow P(x)$ тождественно истинно.

Сопоставляя определение подмножества и определение равенства множеств, мы видим, что множество A равно множеству B тогда и только тогда, когда A есть подмножество B и наоборот, т.е.

$$A = B \Leftrightarrow ((A \subseteq B) \wedge (B \subseteq A)). \tag{1.2}$$

Формула (1.2) является основой для построения доказательств о равенстве множеств. Ее применение состоит в следующем. Чтобы доказать равенство двух множеств X и Y, т.е.

что $X = Y$, достаточно доказать два включения $X \subseteq Y$ и $Y \subseteq X$, т.е. доказать, что из предположения $x \in X$ (для произвольного x) следует, что $x \in Y$, и, наоборот, из предположения $x \in Y$ следует, что $x \in X$. Такой метод доказательства теоретико-множественных равенств называют ***методом двух включений***. Примеры применения этого метода мы дадим позже.

Замечание. Равенство множеств $\{x\colon P(x)\}$ и $\{x\colon Q(x)\}$ означает, что предикаты $P(x)$ и $Q(x)$ эквивалентны, т.е. предикат $P(x) \Leftrightarrow Q(x)$ является тождественно истинным. #

Если $B \subseteq A$, но $B \neq A$, то пишут $B \subset A$ и B называют ***строгим подмножеством*** (или *собственным подмножеством*) множества A, а символ $\subset$ — ***символом строгого включения***.

Для всякого множества A может быть образовано множество всех подмножеств множества A. Его называют ***булеаном множества*** A и обозначают 2^A:

$$2^A = \{X\colon X \subseteq A\}.$$

Для булеана используют также обозначения $\mathcal{P}(A)$, $\mathcal{B}(A)$ и $\exp(A)$.

Пример. а. Булеан множества $\{a, b\}$ состоит из четырех множеств $\varnothing$, $\{a\}$, $\{b\}$, $\{a, b\}$, т.е. $2^{\{a,b\}} = \{\varnothing, \{a\}, \{b\}, \{a, b\}\}$.

б. Булеан $2^{\mathbb{N}}$ состоит из всех возможных, конечных или бесконечных, подмножеств множества $\mathbb{N}$. Так, $\varnothing \in 2^{\mathbb{N}}$, $\{5\} \in \in 2^{\mathbb{N}}$, вообще для любого n множество $\{n\} \in 2^{\mathbb{N}}$, множество $\{1, \ldots, n\} \in 2^{\mathbb{N}}$, $\{n\colon n = 2k, k = 1, 2, \ldots\} \in 2^{\mathbb{N}}$ и т.п. #

Для булеана 2^A мы можем рассматривать произвольные его подмножества. Таким подмножеством, например, будет одноэлементное множество $\{B\}$, где B — произвольное подмножество A. Подчеркнем, что единственным элементом множества $\{B\}$ является, в свою очередь, некоторое множество. Вообще же образование булеана открывает путь для построения множеств, элементами которых являются множества, элементами

которых, в свою очередь, являются некоторые множества, и т.д. Так можно определить множества 2^{2^A}, $2^{2^{2^A}}$ и т.д.*

Введенные выше операции над множествами обладают следующими свойствами:

1) $A \cup B = B \cup A$;
2) $A \cap B = B \cap A$;
3) $A \cup (B \cup C) = (A \cup B) \cup C$;
4) $A \cap (B \cap C) = (A \cap B) \cap C$;
5) $A \cap (B \cup C) = (A \cap B) \cup (A \cap C)$;
6) $A \cup (B \cap C) = (A \cup B) \cap (A \cup C)$;
7) $\overline{A \cup B} = \overline{A} \cap \overline{B}$;
8) $\overline{A \cap B} = \overline{A} \cup \overline{B}$;
9) $A \cup \varnothing = A$;
10) $A \cap \varnothing = \varnothing$;
11) $A \cap U = A$;
12) $A \cup U = U$;
13) $A \cup \overline{A} = U$;
14) $A \cap \overline{A} = \varnothing$;
15) $A \cup A = A$;
16) $A \cap A = A$;
17) $\overline{\overline{A}} = A$;
18) $A \setminus B = A \cap \overline{B}$;
19) $A \triangle B = (A \cup B) \setminus (A \cap B)$;
20) $(A \triangle B) \triangle C = A \triangle (B \triangle C)$;
21) $A \triangle B = B \triangle A$;
22) $A \cap (B \triangle C) = (A \cap B) \triangle (A \cap C)$.

*Применяя теорию множеств, часто выстраивают подобные „башни“ булеанов, начиная этот процесс с элементов, не рассматриваемых как множества. Такие элементы называют праэлементами. Далее в качестве праэлементов берутся любые числа, а также такие объекты, как константы, переменные, буквы какого-нибудь алфавита и т.п.

Каждое из написанных выше равенств, верное для любых входящих в них множеств, часто называют ***теоретико-множественным тождеством***. Любое из них может быть доказано методом двух включений. Докажем этим методом тождество 19.

Пусть $x \in A \triangle B$. Тогда, согласно определению симметрической разности, $x \in (A \setminus B) \cup (B \setminus A)$. Это означает, что $x \in (A \setminus B)$ или $x \in (B \setminus A)$. Если $x \in (A \setminus B)$, то $x \in A$ и $x \notin B$, т.е. $x \in A \cup B$ и при этом $x \notin A \cap B$. Если же $x \in (B \setminus A)$, то $x \in B$ и $x \notin A$, откуда $x \in A \cup B$ и $x \notin A \cap B$. Итак, в любом случае из $x \in (A \setminus B) \cup (B \setminus A)$ следует $x \in A \cup B$ и $x \notin A \cap B$, т.е. $x \in (A \cup B) \setminus (A \cap B)$. Таким образом, доказано, что

$$A \triangle B \subseteq (A \cup B) \setminus (A \cap B).$$

Покажем обратное включение $(A \cup B) \setminus (A \cap B) \subseteq A \triangle B$.

Пусть $x \in (A \cup B) \setminus (A \cap B)$. Тогда $x \in A \cup B$ и $x \notin A \cap B$. Из $x \in A \cup B$ следует, что $x \in A$ или $x \in B$. Если $x \in A$, то с учетом $x \notin A \cap B$ имеем $x \notin B$, и поэтому $x \in A \setminus B$. Если же $x \in B$, то опять-таки в силу $x \notin A \cap B$ получаем, что $x \notin A$ и $x \in B \setminus A$. Итак, $x \in A \setminus B$ или $x \in B \setminus A$, т.е. $x \in (A \setminus B) \cup (B \setminus A)$. Следовательно,

$$(A \cup B) \setminus (A \cap B) \subseteq A \triangle B.$$

Оба включения имеют место, и тождество 19 доказано.

Метод двух включений является универсальным и наиболее часто применяемым методом доказательства теоретико-множественных тождеств. Помимо метода двух включений для доказательства теоретико-множественных тождеств могут быть использованы другие методы, например *метод характеристических функций* (см. **Д.1.2**).

Кроме того, теоретико-множественные тождества можно доказывать, используя ранее доказанные тождества для преобразования левой части к правой или наоборот. Такой метод

доказательства часто называют ***методом эквивалентных преобразований.***

Докажем этим методом тождество 22, пользуясь тождествами 1–19. Преобразуем левую часть к правой:

$$
\begin{aligned}
(A\cap B)\triangle(A\cap C) &= ((A\cap B)\cup(A\cap C))\cap\overline{(A\cap B)\cap(A\cap C)} = \\
&= (A\cap(B\cup C))\cap(\overline{(A\cap B)}\cup\overline{(A\cap C)}) = \\
&= (A\cap(B\cup C))\cap(\overline{A}\cup\overline{B})\cup(\overline{A}\cup\overline{C}) = \\
&= (A\cap(B\cup C))\cap(\overline{A}\cup(\overline{B}\cup\overline{C})) = \\
&= ((A\cap(B\cup C))\cap\overline{A})\cup((A\cap(B\cup C))\cap(\overline{B}\cup\overline{C})) = \\
&= \varnothing\cup((A\cap(B\cup C))\cap(A\cap(\overline{B}\cup\overline{C}))) = \\
&= (A\cap(B\cup C))\cap(A\cap\overline{(B\cap C)}) = \\
&= A\cap((B\cup C)\cap\overline{(B\cap C)}) = \\
&= A\cap((B\cup C)\setminus(B\cap C)) = A\cap(B\triangle C).
\end{aligned}
$$

Тождество доказано.

1.2. Кортеж. Декартово произведение

Пусть A и B — произвольные множества. ***Неупорядоченная пара*** на множествах A и B — это любое множество $\{a, b\}$, где $a\in A$, $b\in B$ или $a\in B$, $b\in A$.

Если $A = B$, то говорят о неупорядоченной паре на множестве A. Исходя из понятия *равенства множеств*, можно утверждать, что неупорядоченная пара $\{a, b\}$ равна неупорядоченной паре $\{c, d\}$ если и только если $a = c$ и $b = d$ или $a = d$ и $b = c$. Заметим, что равенство элементов множества понимается здесь (и далее в аналогичных ситуациях) как равенство *индивидных констант.*

В том случае, когда в неупорядоченной паре $\{a, b\}$ элементы a и b совпадают, получаем, что $\{a, b\} = \{a, a\}$. Но такая запись, как мы условились выше, задает то же самое множество, что и $\{a\}$. Таким образом, при $a = b$ неупорядоченная пара $\{a, b\}$

„вырождается“ в одноэлементное множество $\{a\}$. При $a \neq b$ неупорядоченная пара будет двухэлементным множеством.

Упорядоченная пара на множествах A и B, обозначаемая записью (a, b), определяется не только самими элементами $a \in A$ и $b \in B$, но и порядком, в котором они записаны. И в этом состоит ее существенное отличие от неупорядоченной пары. Если $A = B$, то говорят об упорядоченной паре на множестве A.

Существенная роль порядка, в котором перечисляются элементы упорядоченной пары, фиксируется определением равенства упорядоченных пар.

Определение 1.1. Две упорядоченные пары (a, b) и (a', b') на множествах A и B называют ***равными***, если $a = a'$ и $b = b'$.

Замечание 1.2. Упорядоченную пару (a, b) не следует связывать с множеством $\{a, b\}$, так как упорядоченная пара характеризуется не только составом, но и порядком элементов в ней. Более того, определение этого объекта вообще не позволяет рассматривать его как множество. Но упорядоченную пару можно определить и как множество, полагая, что упорядоченная пара (a, b) есть неупорядоченная пара $\{\{a\}, \{a, b\}\}$, включающая в себя одноэлементное множество $\{a\}$ и неупорядоченную пару $\{a, b\}$. При $a = b$ получаем $(a, a) = \{\{a\}\}$. Такое определение не изменит сути понятия, но тогда следует не определять явно равенство упорядоченных пар, а доказывать теорему о равенстве упорядоченных пар как определенного вида множеств. #

Простейший и важнейший пример использования упорядоченных пар дает аналитическая геометрия [III]. Если на плоскости введена некоторая *прямоугольная система координат*, то каждая точка плоскости однозначно задается упорядоченной парой действительных чисел — *координатами* этой *точки*. Ни у кого не возникает сомнений в том, что порядок, в котором перечисляются координаты точки, является существенным: точка, заданная координатами (1, 3), совсем не то же самое, что точка с координатами (3, 1).

Обобщением понятия упорядоченной пары является ***упорядоченный n-набор****, или ***кортеж***. В отличие от конечного множества $\{a_1, \ldots, a_n\}$ кортеж $(a_1, \ldots, a_n)$ на множествах $A_1, \ldots, A_n$ характеризуется не только входящими в него элементами $a_1 \in A_1, \ldots, a_n \in A_n$, но и порядком, в котором они перечисляются. Как и для упорядоченных пар, роль порядка в кортеже фиксируется определением равенства кортежей.

Определение 1.2. Два кортежа $(a_1, \ldots, a_n)$ и $(b_1, \ldots, b_n)$ на множествах $A_1, \ldots, A_n$ равны, если $a_i = b_i$, $i = \overline{1, n}$.

Число n называется ***длиной кортежа*** (или ***размерностью кортежа***), а элемент a_i — i-й ***проекцией*** (компонентой) ***кортежа***. Для двух кортежей одинаковой размерности их компоненты с одинаковыми номерами называют ***одноименными компонентами***. Определение 1.2 равенства кортежей можно переформулировать так: два кортежа одинаковой размерности равны тогда и только тогда, когда их одноименные компоненты совпадают.

Простейшим примером кортежа является *арифметический вектор*.

Определение 1.3. Множество всех кортежей длины n на множествах $A_1, \ldots, A_n$ называют ***декартовым (прямым) произведением множеств*** $A_1, \ldots, A_n$ и обозначают $A_1 \times \ldots \times A_n$.

Таким образом,

$$A_1 \times \ldots \times A_n = \{(a_1, \ldots, a_n) \colon a_1 \in A_1, \ldots, a_n \in A_n\}.$$

Если все множества A_i, $i = \overline{1, n}$, равны между собой, то указанное декартово произведение называют n-й ***декартовой степенью множества*** A и обозначают A^n. В частности, при $n = 2$ получаем ***декартов квадрат***, а при $n = 3$ — ***декартов куб*** множества A.

*Говорят также: ***упорядоченная n-ка*** (например, упорядоченная тройка, четверка, пятерка и т.д.).

По определению полагают, что первая декартова степень любого множества A есть само множество A, т.е. $A^1 = A$.

Декартово произведение имеет следующие свойства:

1) $A \times (B \cup C) = (A \times B) \cup (A \times C)$;

2) $A \times (B \cap C) = (A \times B) \cap (A \times C)$;

3) $A \times \varnothing = \varnothing \times A = \varnothing$.

Эти свойства нетрудно доказать *методом двух включений*. Докажем, например, первое тождество. Если $(x, y) \in A \times (B \cup C)$, то $x \in A$ и $y \in B \cup C$. Из того, что $y \in B \cup C$, следует $y \in B$ или $y \in C$. Если $y \in B$, то $(x, y) \in A \times B$, а если $y \in C$, то $(x, y) \in A \times C$. Итак, $(x, y) \in A \times B$ или $(x, y) \in A \times C$, т.е. $(x, y) \in (A \times B) \cup (A \times C)$. Следовательно, $A \times (B \cup C) \subseteq (A \times B) \cup (A \times C)$.

Доказательство обратного включения аналогично.

Обратим внимание на последнее из записанных выше трех тождеств. Из него вытекает, что пустое множество при построении декартовых произведений множеств играет ту же роль, что и нуль при умножении чисел. Докажем справедливость этого тождества. В самом деле, множество $\varnothing \times A$ (для любого A) есть множество всех упорядоченных пар (x, y), таких, что $x \in \varnothing$ и $y \in A$. Но таких элементов x, что $x \in \varnothing$, не существует, и, следовательно, упорядоченных пар (x, y), принадлежащих декартову произведению $\varnothing \times A$, не существует, т.е. $\varnothing \times A = \varnothing$. Аналогично доказывается, что и $A \times \varnothing = \varnothing$.

1.3. Соответствия и бинарные отношения

Отображение f из множества A в множество B считается заданным, если каждому элементу $x \in A$ сопоставлен единственный элемент $y \in B$. Отображение f из множества A в множество B обозначают записью $f\colon A \to B$. Элемент $y \in B$, который отображением f сопоставляется элементу $x \in A$, называют *образом элемента x при отображении f* и обозначают $f(x)$.

Каждое отображение однозначно определяет множество *упорядоченных пар* $\{(x,y)\colon x \in A, y = f(x)\}$, являющееся подмножеством *декартова произведения* $A \times B$ множества A на множество B и называемое *графиком отображения* f.

Наоборот, пусть в декартовом произведении $A \times B$ задано такое подмножество f, что: 1) для любого $x \in A$ существует $y \in B$, для которого $(x, y) \in f$; 2) для любых двух пар (x, y) и (x', y') множества f из равенства $x = x'$ следует равенство $y = y'$. Тогда множество f единственным образом определяет некоторое отображение из A в B. Это отображение, обозначаемое также f, элементу $x \in A$ сопоставляет элемент $y \in B$, удовлетворяющий условию $(x, y) \in f$. Таким образом, мы можем отождествить отображения с их графиками и считать, что отображение есть подмножество декартова произведения.

Отображение f множества A в себя называют ***тождественным***, если $f(x) = x$ при всех x из A.

В общем случае для отображения $f\colon A \to B$ может существовать несколько различных элементов множества A, образы которых совпадают. Множество всех элементов $x \in A$, для которых $f(x) = y_0$, называют *прообразом элемента* $y_0 \in B$ *при отображении* f.

Так, прообраз числа a, $|a| \leqslant 1$, при отображении $y = \sin x$ есть множество всех решений уравнения $\sin x = a$, т.е. множество

$$\{x\colon x = \arcsin a + 2\pi n, n \in \mathbb{Z}\} \cup \{x\colon x = \pi - \arcsin a + 2\pi n, n \in \mathbb{Z}\}.$$

Прообраз элемента $y_0 \in B$ может быть *пустым множеством*. Это имеет место, например, для числа 2 при отображении $y = \sin x$.

Множество всех $y \in B$, таких, что найдется $x \in A$, для которого $y = f(x)$, называют ***областью значений отображения*** f. Область значений отображения f будем обозначать $R(f)$.

Отображение $f\colon A \to B$ называют ***инъективным*** (*инъекцией*), если каждый элемент из области его значений имеет единственный прообраз, т.е. из $f(x_1) = f(x_2)$ следует $x_1 = x_2$.

Отображение $f\colon A \to B$ называют ***сюръективным*** (*сюръекцией*), если его область значений совпадает со всем множеством B. Сюръективное отображение из A в B называют также ***отображением*** множества A ***на*** множество B.

Отображение $f\colon A \to B$ называют *биективным* (*биекцией*), если оно одновременно инъективно и сюръективно.

Таким образом, если отображение $f\colon A \to B$ биективно, то каждому элементу множества A отвечает единственный элемент множества B и наоборот. Тогда говорят, что множества A и B находятся между собой во ***взаимно однозначном соответствии.***

Биекцию множества A на себя называют ***автоморфизмом множества*** A. Используют также термин* „подстановка множества".

Пример 1.2. а. Отображение, заданное равенством $\nu(n) = n+1$, есть, как нетрудно показать, биекция множества натуральных чисел $\mathbb{N}$ на его подмножество $\mathbb{N}\setminus\{1\}$.

б. Отображение $\nu\colon n \mapsto 2n$ есть биекция множества всех натуральных чисел на множество всех четных натуральных чисел.

в. Любая *показательная функция* $y = a^x$, $a > 0$, есть биекция множества $\mathbb{R}$ всех действительных чисел на множество $\mathbb{R}^+$ всех положительных действительных чисел.

г. Функция $y = \operatorname{arctg} x$ есть биекция множества $\mathbb{R}$ на интервал $(-\pi/2, \pi/2)$.

д. Поворот окружности на заданный угол α, т.е. отображение, сопоставляющее каждой точке окружности точку, в которую она перейдет при повороте всей окружности вокруг ее центра на угол α, есть автоморфизм множества точек окружности. #

Пусть задано отображение $f\colon A \to B$ и $C \subseteq A$ — некоторое множество. Множество $f(C)$ элементов $y \in B$, таких, что

*Иногда этот термин употребляют только для автоморфизма конечного множества.

$y = f(x)$, $x \in C$, называют *образом множества C при отображении f*. Например, при отображении $y = \sin x$ отрезок $[0, 1]$ является образом множества (отрезка) $[0, \pi]$, равно как и любого объединения отрезков вида $[2\pi k, (2k+1)\pi]$ (для произвольного целого k). При $k = 0$ это можно записать следующим образом: $\sin([0, \pi]) = [0, 1]$.

Заметим, что для любого отображения $f\colon A \to B$ образ $f(A)$ всего множества A есть область значений данного отображения.

Для произвольного множества $D \subseteq B$ множество всех элементов $x \in A$, таких, что $f(x) \in D$, называют *прообразом множества D* при отображении f.

Например, для любого действительного числа $a \in [0, 1)$ множество, которое является объединением всех отрезков вида $[\arcsin a + 2\pi k, \pi - \arcsin a + 2\pi k]$, $k \in \mathbb{Z}$, есть прообраз отрезка $[a, 1]$ при отображении $y = \sin x$.

Прообраз области значений произвольного отображения $f\colon A \to B$ совпадает со всем множеством A.

Множество всех отображений из A в B будем обозначать как B^A.

Понятие отображения можно обобщить. Обобщение может проходить по двум позициям. Во-первых, можно отказаться от полной определенности отображения, полагая, что образ определен не для каждого элемента множества A, а для некоторых элементов этого множества. Тогда придем к понятию ***частичного отображения***. При этом подмножество всех элементов A, для которых определен образ, называют ***областью определения*** данного ***частичного отображения***.

Многие элементарные функции являются частичными отображениями множества $\mathbb{R}$ всех действительных чисел в себя. Например, функция $y = \operatorname{tg} x$ есть частичное отображение с областью определения $\mathbb{R} \setminus \left\{x\colon x = \frac{\pi}{2} + \pi k, k \in \mathbb{Z}\right\}$.

Во-вторых, можно отказаться от однозначности отображения, полагая, что данному $x \in A$ сопоставлен не один, а несколь-

ко образов (множество образов) в множестве B. В этом случае говорят, что задано ***соответствие**** из множества A в множество B.

Примером могут служить обратные тригонометрические функции: скажем, „большой“ арксинус [I], сопоставляющий каждому $x \in \mathbb{R}$ множество всех таких чисел y, что $\sin y = x$, т.е. множество, являющееся прообразом элемента x при отображении, определяемом графиком функции $y = \sin x$.

Если задано соответствие ρ из A в B, будем использовать обозначение $\rho(x)$ по аналогии с обозначением $f(x)$ для отображений, понимая при этом, что $\rho(x)$ есть уже не элемент множества B, а его подмножество.

Аналогично графику отображения можно определить ***график соответствия*** ρ из множества A в множество B как множество C_ρ упорядоченных пар (x, y), таких, что $x \in A$, $y \in B$ и элементы x, y связаны соответствием ρ, т.е. $y \in \rho(x)$. Указанное множество C_ρ упорядоченных пар есть подмножество декартова произведения $A \times B$.

Обратно, фиксируя на декартовом произведении $A \times B$ какое-либо подмножество C, мы тем самым однозначно определяем некоторое соответствие ρ_C из A в B, а именно $\rho_C(x) = \{y\colon y \in B \wedge (x, y) \in C\}$. Нетрудно заметить, что графиком соответствия ρ_C будет как раз множество C, а соответствием, отвечающим графику C_ρ, будет ρ. Поэтому можно отождествить соответствие с его графиком и считать, что соответствие из множества A в множество B есть некоторое подмножество ρ декартова произведения $A \times B$, т.е. $\rho \subseteq A \times B$. В частности, при $\rho = \varnothing$ получаем ***пустое соответствие***, а при ρ, совпадающем со всем указанным декартовым произведением, — ***универсальное соответствие***.

При этом будем писать $(x, y) \in \rho$ для упорядоченных пар, связанных соответствием ρ.

*Используют также термины „частичное мультиотображение“, „частичная многозначная функция“.

Пример 1.3. Рассмотрим множество программистов $A = \{И, П, С\}$ и множество программ $B = \{n_1, n_2, n_3, n_4, n_5\}$. Зададим соответствие τ из A в B, связывающее программистов и разрабатываемые ими программы:

$$\tau = \{(И, n_1), (И, n_3), (И, n_5), (П, n_2), (П, n_4), (С, n_2), (С, n_5)\} \subseteq A \times B. \quad \#$$

Область определения соответствия $\rho \subseteq A \times B$ из множества A в множество B — это множество всех первых компонент упорядоченных пар из ρ:

$$D(\rho) = \{x\colon (\exists y \in B)(x, y) \in \rho\}.$$

Область значения соответствия ρ — это множество всех вторых компонент упорядоченных пар из ρ:

$$R(\rho) = \{y\colon (\exists x \in A)(x, y) \in \rho\}.$$

Из определения вытекает, что $D(\rho) \subseteq A$, $R(\rho) \subseteq B$. Соответствие из A в B называют ***всюду определенным***, если его область определения совпадает с множеством A: $D(\rho) = A$.

Сечением соответствия $\rho \subseteq A \times B$ для фиксированного элемента $x \in A$ будем называть множество $\rho(x) = \{y\colon (x, y) \in \rho\}$. Можно сказать, что сечение соответствия $\rho(x)$ есть множество всех „образов“ элемента x при данном соответствии.

Сечением соответствия ρ ***по множеству*** $C \subseteq A$ будем называть множество $\rho(C) = \{y\colon (x, y) \in \rho, x \in C\}$.

Пример 1.4. Область определения соответствия τ из примера 1.3есть все множество A, а область значения — все множество B. Сечением соответствия τ по элементу $П$ будет множество $\tau(П) = \{n_2, n_4\}$. $\#$

Соответствие $\rho \subseteq A \times A$ из множества A в себя, т.е. подмножество множества A^2, называют ***бинарным отношением на множестве*** A.

Пример 1.5. Простейшим примером бинарного отношения является отношение нестрогого неравенства на множестве действительных чисел $\mathbb{R}$. Здесь каждому $x \in \mathbb{R}$ поставлены в соответствие такие $y \in \mathbb{R}$, для которых справедливо $x \leqslant y$. #

Для произвольного бинарного отношения на некотором множестве часто используют запись $x\,\rho\,y$ вместо $(x, y) \in \rho$, говоря при этом об ***элементах, связанных бинарным отношением*** ρ. Это согласуется с традиционной формой записи некоторых часто используемых бинарных отношений. Так, пишут $x \leqslant y$, а не $(x, y) \in \leqslant$. Для таких бинарных отношений употребляют устоявшиеся словосочетания. Например, запись $x \leqslant y$ читается так: „x не больше y“.

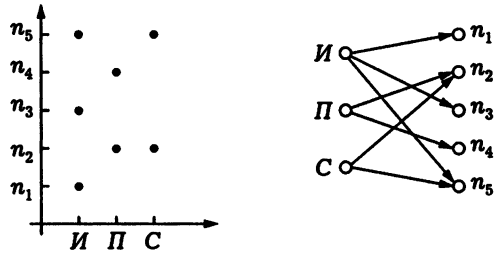
Бинарное отношение на множестве A, состоящее из всех пар (x,x), т.е. пар с совпадающими компонентами, называют ***диагональю*** множества* A и обозначают id_A. Нетрудно понять, что диагональ A есть *тождественное отображение* A на себя.

Для наглядного изображения соответствий из A в B (бинарных отношений, в частности) будем использовать два способа. Первый из этих способов состоит в интерпретации соответствия как подмножества декартова произведения, которое можно изображать примерно так же, как на плоскости можно изображать подмножества декартова квадрата числовых множеств. Второй способ, применяемый для конечных множеств A и B, — построение так называемого ***графа соответствия***. В этом случае элементы множеств A и B изображаются на плоскости кружочками. Если и только если пара (u, v) принадлежит соответствию ρ, то в графе соответствия из кружочка, обозначающего элемент $u \in A$, проводим стрелку к кружочку, обозначающему элемент $v \in B$. Для бинарного отношения на конечном множестве A часто удобнее использовать граф другого вида. Элементы множества A изображаются кружочками только один раз, а стрелки проводятся по тем же правилам,

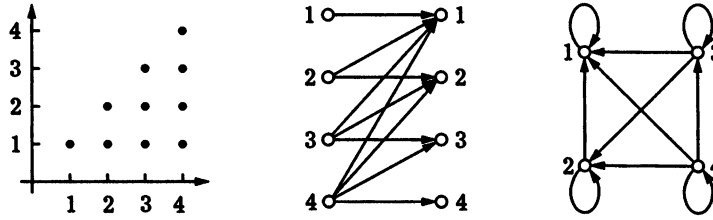
*Иногда говорят о диагонали в множестве A, хотя правильнее было бы называть это отношение диагональю декартова квадрата множества A.

что и в графе соответствия. Заметим, что при таком построении возможно соединение кружочка стрелкой с самим собой (петля).

Пример 1.6. а. На рис. 1.1, *а* изображены график и граф бинарного соответствия из примера 1.3.



а



б

Рис. 1.1

б. Пусть $A = \{1, 2, 3, 4\}$. Бинарное отношение ρ на A определим как множество всех упорядоченных пар (x, y), таких, что $x \geqslant y$. Тогда

$$\rho = \{(1, 1), (2, 1), (2, 2), (3, 1), (3, 2), (3, 3), (4, 1), (4, 2), (4, 3), (4, 4)\}.$$

Область определения отношения $D(\rho) = \{1, 2, 3, 4\}$, область значений $R(\rho) = \{1, 2, 3, 4\}$. График и два варианта графа отношения ρ изображены на рис. 1.1, *б*.

в. Множество точек окружности $x^2 + y^2 = 1$ есть график бинарного отношения на множестве действительных чисел, состоящего из всех таких упорядоченных пар (x, y), что $y = \pm\sqrt{(1 - x^2)}$, или, что равносильно, компоненты пары удовлетворяют уравнению $x^2 + y^2 = 1$. Область определения бинарного отношения есть отрезок $[-1, 1]$, область значения — также отрезок $[-1, 1]$. #

Соответствие $\rho \subseteq A \times B$ называют ***функциональным по*** второй (первой) ***компоненте***, если для любых двух упорядоченных пар $(x, y) \in \rho$ и $(x', y') \in \rho$ из равенства $x = x'$ следует $y = y'$ (и из $y = y'$ следует $x = x'$). Функциональность соответствия по второй компоненте означает, что, фиксируя в любой упорядоченной паре, принадлежащей данному соответствию, первую компоненту, мы однозначно определяем и вторую компоненту. Таким образом, мы можем сказать, что соответствие, функциональное по второй компоненте, есть отображение (возможно, частичное).

Поэтому соответствие $f \subseteq A \times B$ является отображением из A в B, если и только если оно всюду определено (т.е. $D(f) = A$) и функционально по второй компоненте. Отметим также, что отображение из A в B является инъекцией тогда и только тогда, когда оно функционально по первой компоненте.

В заключение обобщим понятие соответствия, определив отношения произвольной арности.

Определение 1.4. Произвольное подмножество ρ декартова произведения $A_1 \times \ldots \times A_n$ называют (***n-арным*** или ***n-местным***) ***отношением*** на множествах $A_1, \ldots, A_n$.

В случае если все множества $A_1, \ldots, A_n$ совпадают, т.е. $A_1 = \ldots = A_n = A$, говорят об ***n-арном отношении на множестве*** A.

Если ρ — n-арное отношение на множествах $A_1, \ldots, A_n$ и $(a_1, \ldots, a_n) \in \rho$, то говорят об ***элементах*** $a_1, \ldots, a_n$, ***связанных отношением*** ρ.

Замечание 1.3. При $n = 2$ получаем ***бинарное отношение*** на множествах A_1, A_2. Это не что иное, как соответствие из A_1 в A_2, где множества A_1 и A_2, вообще говоря, различны.

При $A_1 = A_2 = A$ получаем введенное ранее бинарное отношение на множестве, т.е. подмножество декартова квадрата A.

Таким образом, в общем случае (при произвольном $n \geqslant 2$) следует, строго говоря, различать термины „n-арное отношение“ и „n-арное отношение на множестве“.

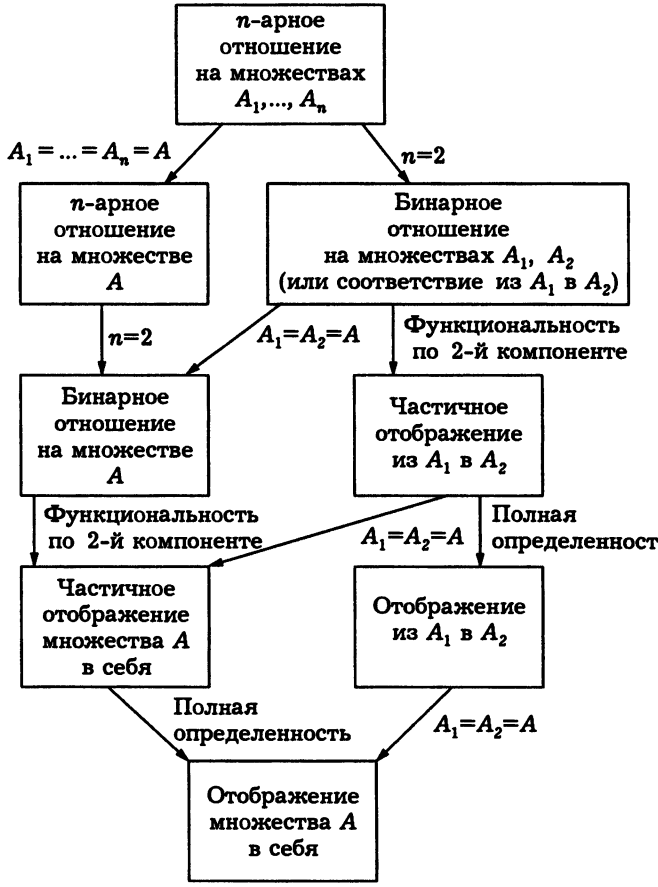


Рис. 1.2

Связь между введенными понятиями отношения, соответствия и отображения проиллюстрирована на рис. 1.2. #

Пусть n-арное отношение $\rho \subseteq A_1 \times \ldots \times A_n$ удовлетворяет условию: для любых двух кортежей $(x_1, \ldots, x_i, \ldots, x_n) \in \rho$ и $(y_1, \ldots, y_i, \ldots, y_n) \in \rho$ из выполнения равенств $x_k = y_k$ для любого $k \neq i$ $(0 \leqslant k \leqslant n)$ следует, что и $x_i = y_i$. Тогда отношение ρ называют ***функциональным по i-й компоненте*** $(1 \leqslant i \leqslant n)$.

Другими словами, функциональность n-местного отношения по i-й $(i \leqslant n)$ компоненте равносильна условию, что, фиксируя все компоненты, кроме i-й, мы однозначно определяем и i-ю компоненту.

Пример 1.7. а. Представим строку учебного расписания как кортеж вида

(преподаватель, группа, дисциплина, аудитория, день, час).

Тогда расписание можно рассматривать как секстарное (шестиместное) отношение на соответствующих множествах. Оно будет функционально по первой компоненте, если, конечно, предположить, что два преподавателя или более не проводят одно и то же занятие одновременно в одном и том же месте (хотя, например, на лабораторных работах это возможно). Оно также функционально по третьей компоненте (один преподаватель не может вести одновременно занятия по разным дисциплинам), по четвертой (преподаватель со своей группой не могут находиться в разных аудиториях) и не будет, вообще говоря, функционально по второй, пятой и шестой компонентам.

б. Рассмотрим на множестве V_3 геометрических векторов в пространстве тернарное (трехместное) отношение ρ, состоящее из всех упорядоченных троек $(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z})$ компланарных векторов. Это отношение не является функциональным ни по одной компоненте, так как любым двум векторам соответствует бесконечно много векторов, образующих с ними компланарную тройку.

1.4. Операции над соответствиями

Поскольку *соответствия* можно считать множествами, то все операции над множествами (*пересечение, объединение, разность, дополнение* и т.д.) можно применить и к соответствиям. Заметим, что, говоря о дополнении соответствия из A в B, мы имеем в виду дополнение до *универсального соответствия* из A в B, т.е. до *декартова произведения* $A \times B$. Естественно, что и равенство соответствий можно трактовать как *равенство множеств*.

В то же время на соответствия можно распространить операции, определяемые для отображений. Мы рассмотрим здесь две такие операции.

Композиция соответствий. Следуя аналогии с композицией отображнений, ***композицией*** (***произведением***) ***соответствий*** $\rho \subseteq A \times B$ и $\sigma \subseteq B \times C$ называют соответствие

$$\rho \circ \sigma = \{(x, y)\colon (\exists z \in B)((x, z) \in \rho) \wedge ((z, y) \in \sigma)\}. \qquad (1.3)$$

Поясним построение композиции двух соответствий. Обратимся сначала к отображениям (как частным случаям соответствий). Пусть заданы отображения (возможно, *частичные*): f из A в B и g из B в C. Композиция* $f \circ g$ определяется как отображение из A в C, задаваемое формулой $y = g(f(x))$. Тем самым задается *график отображения* $f \circ g$, т.е. множество *упорядоченных пар* (x, y), таких, что $y = g(f(x))$. При этом упорядоченная пара (x, y) будет принадлежать графику отображения $f \circ g$, если и только если найдется элемент $z \in B$,

*Необходимо заметить, что в [I] запись $g \circ f(x)$ означает $g(f(x))$, т.е. отображения в композиции пишутся в порядке, обратном тому, в каком они применяются. Мы же будем везде использовать запись $f \circ g$, полагая, что $f \circ g(x) = g(f(x))$ и порядок записи отображений в композиции совпадает с порядком их применения. Это обусловлено тем, что композиция отображений определяется нами как частный случай композиции соответствий, при записи которой естественным оказывается именно такой порядок.

такой, что $z = f(x)$ и $y = g(z)$. Таким образом, график композиции отображений f и g есть

$$f \circ g = \{(x, y): (\exists z)(z = f(x) \text{ и } y = g(z))\} = \\ = \{(x, y): y = g(f(x))\}. \quad (1.4)$$

Легко видеть, что (1.4) есть частный случай (1.3). Отметим, что при построении композиции отображений обычно предполагают, что пересечение области значений отображения f и области определения отображения g не пусто ($R(f) \cap D(g) \neq \varnothing$), поскольку в противном случае композиция была бы пуста. Для отображений, не являющихся частичными, $R(f) \subseteq D(g)$, так как $D(g) = B$. Поэтому в данном случае пересечение $R(f) \cap$ $\cap D(g)$ всегда не пусто.

Полезно отметить также, что если f и g — биекции, то и композиция их тоже будет биекцией.

Вернемся к рассмотрению композиции соответствий $\rho \circ \sigma$. Полагая, что область определения $D(\rho)$ соответствия ρ не пуста, возьмем произвольный элемент $x \in D(\rho)$. Пусть *сечение* $\rho(x) \subseteq B$ соответствия ρ не пусто и найдется такой элемент $z \in \rho(x)$, что сечение $\sigma(z) \subseteq C$ также не пусто. Тогда непустое множество $\{(x, t): t \in \sigma(z)\}$ будет подмножеством сечения соответствия $\rho \circ \sigma$ в точке x. Сечением соответствия $\rho \circ \sigma$ в точке x будет непустое в силу сделанных предположений множество всех таких упорядоченных пар $(x, t) \in A \times C$, что $x \in D(\rho)$, а $t \in \sigma(z)$ для некоторого $z \in \rho(x)$. Говоря неформально, нужно перебрать все элементы z из сечения $\rho(x)$. Таким образом, различие в построении композиции соответствий и композиции отображений заключается в том, что „промежуточный“ элемент z в общем случае не единственный и каждому такому элементу также ставится в соответствие не единственный элемент $y \in C$.

Пример 1.8. Соответствие ρ возьмем из примера 1.3. Соответствие σ зададим как соответствие из множества про-

грамм $\{n_1, n_2, n_3, n_4, n_5\}$ в множество заказчиков программного обеспечения $\{З_1, З_2, З_3, З_4\}$. Пусть

$$\sigma = \{(n_1, З_3), (n_1, З_4), (n_2, З_1), (n_3, З_2), (n_4, З_4), (n_5, З_3)\}.$$

Рассмотрим процесс построения композиции соответствий ρ и σ. Начнем с элемента *И*. Имеем $\rho(И) = \{n_1, n_3, n_5\}$, $\sigma(n_1) = \{З_3, З_4\}$, $\sigma(n_3) = \{З_2\}$ и $\sigma(n_5) = \{З_3\}$. Отсюда получаем

$$\sigma(n_1) \cup \sigma(n_3) \cup \sigma(n_5) = \{З_2, З_3, З_4\} \quad —$$

сечение композиции по элементу *И*. Рассуждая аналогично, получим $(\rho \circ \sigma)(П) = \{З_1, З_4\}$ и $(\rho \circ \sigma)(С) = \{З_1, З_3\}$.

Построение графа композиции $\rho \circ \sigma$ проиллюстрировано на рис. 1.3. #

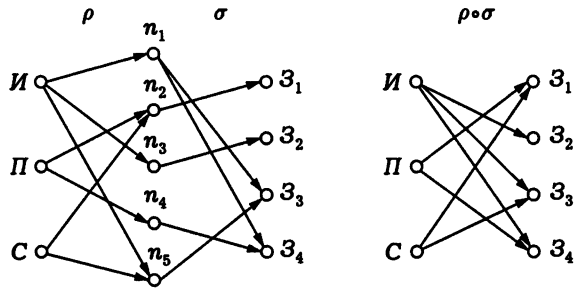


Рис. 1.3

Отметим, что *область определения* композиции *соответствий* содержится в области определения первого соответствия, а *область значений* композиции соответствий — в области значений второго соответствия. Из приведенных рассуждений следует, что для того, чтобы композиция соответствий была отлична от пустого соответствия, необходимо и достаточно, чтобы пересечение области значений первого соответствия и области определения второго соответствия было не пусто.

К определению композиции соответствий можно подойти с более общих позиций. Пусть $\rho \subseteq A \times B$ и $\sigma \subseteq C \times D$. При этом

на множества A, B, C и D априори не накладывается никаких органичений. Композиция $\rho \circ \sigma$ соответствий ρ и σ в этом случае также определяется соотношением (1.3). Чтобы такая композиция была отлична от пустого соответствия, необходимо и достаточно выполнение условия $R(\rho) \cap D(\sigma) \neq \varnothing$. В частности, $\rho \circ \sigma = \varnothing$ всякий раз, когда $B \cap C = \varnothing$.

Пример 1.9. Рассмотрим соответствие

$$\tau = \{(1,\ a), (2,\ a), (3,\ d)\}$$

из множества $A = \{1, 2, 3\}$ в множество $B = \{a, b, d\}$ и соответствие

$$\varphi = \{(b,\ e), (b,\ f), (c,\ f)\}$$

из множества $C = \{b, c, d\}$ в множество $D = \{e, f\}$. В данном случае $B \cap C \neq \varnothing$, но $\tau \circ \varphi = \varnothing$, поскольку $R(\tau) = \{a, d\}$, $D(\varphi) = \{b, c\}$ и $R(\tau) \cap D(\varphi) = \varnothing$. #

Заметим, что композиция соответствий $\rho \subseteq A \times B$ и $\sigma \subseteq C \times D$ не коммутативна, т.е. в общем случае $\rho \circ \sigma \neq \sigma \circ \rho$, поскольку $\rho \circ \sigma \subseteq A \times D$, а $\sigma \circ \rho \subseteq C \times B$.

Бинарное отношение на множестве является частным случаем соответствия. Для двух бинарных отношений ρ и σ, заданных на множестве A, их композиция $\rho \circ \sigma$ (1.3) как соответствий является бинарным отношением на том же множестве A. В этом случае говорят о ***композиции бинарных отношений на множестве*** A.

Композицию $\rho \circ \rho$ бинарного отношения ρ на некотором множестве с самим собой называют ***квадратом бинарного отношения*** ρ и обозначают ρ^2.

Рассмотрим пример построения композиции бинарных отношений на множестве и покажем, что в общем случае для двух бинарных отношений τ и φ также имеет место неравенство $\tau \circ \varphi \neq \varphi \circ \tau$, хотя обе композиции, в отличие от аналогичных композиций двух произвольных соответствий, заданы на одном и том же множестве.

Пример 1.10. а. Зададим на множестве $A = \{1, 2, 3, 4\}$ бинарные отношения $\tau = \{(x, y)\colon x + 1 < y\}$, $\varphi = \{(x, y)\colon |x - y| = 2\}$ и найдем композицию $\tau \circ \varphi$.

Имеем $\tau(1) = \{3, 4\}$, $\varphi(3) = \{1\}$ и $\varphi(4) = \{2\}$. Следовательно, $(\tau \circ \varphi)(1) = \varphi(3) \cup \varphi(4) = \{1, 2\}$. Далее $\tau(2) = \{4\}$, $\varphi(4) = \{2\}$ и $(\tau \circ \varphi)(2) = \{2\}$. Так как $\tau(3) = \tau(4) = \varnothing$, то в итоге получим $\tau \circ \varphi = \{(1, 1), (1, 2), (2, 2)\}$. Построение композиции проиллюстрировано на рис. 1.4, *а*.

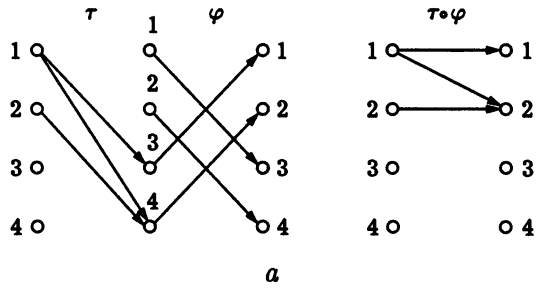


Рис. 1.4

Найдем композицию $\varphi \circ \tau$. Поскольку $\varphi(1) = \{3\}$, а $\tau(3) = \varnothing$, то $(\varphi \circ \tau)(1) = \varnothing$. Аналогично $\varphi(2) = \{4\}$, а $\tau(4) = \varnothing$, поэтому $(\varphi \circ \tau)(2) = \varnothing$. Далее $\varphi(3) = \{1\}$, $\tau(1) = \{3, 4\}$, поэтому $(\varphi \circ \tau)(3) = \{3, 4\}$, а $\varphi(4) = \{2\}$, $\tau(2) = \{4\}$ и $(\varphi \circ \tau)(4) = \{4\}$. Построение композиции проиллюстрировано на рис. 1.4, *б*.

Легко видеть, что $\tau \circ \varphi \neq \varphi \circ \tau$.

6. Пусть отношение ρ на множестве действительных чисел определено как функция $y = ax + b$. Найдем квадрат этого отношения (линейной функции от одного переменного).

Согласно (1.4), это будет функция h, такая, что $h(x) = a(ax + b) + c$, т.е. $h(x) = a^2x + (ab + c)$. Это тоже линейная функция, но с другими коэффициентами. #

Приведем некоторые свойства композиции соответствий:

1) $\rho \circ (\sigma \circ \tau) = (\rho \circ \sigma) \circ \tau$;

2) для любого соответствия ρ имеет место $\rho \circ \varnothing = \varnothing \circ \rho = \varnothing$;

3) $\rho \circ (\sigma \cup \tau) = (\rho \circ \sigma) \cup (\rho \circ \tau)$;

4) для любого бинарного отношения на множестве A имеет место равенство $\rho \circ \mathrm{id}_A = \mathrm{id}_A \circ \rho = \rho$.

Эти свойства нетрудно доказать *методом двух включений*. Рассмотрим в качестве примера доказательство свойства 3. Пусть некоторая упорядоченная пара (x, y) принадлежит композиции $\rho \circ (\sigma \cup \tau)$. Тогда, согласно (1.3), найдется такой элемент z, что $(x, z) \in \rho$ и $(z, y) \in \sigma \cup \tau$. Последнее означает, что $(z, y) \in \sigma$ или $(z, y) \in \tau$. Таким образом, для элемента z имеем $(x, z) \in \rho$ и $(z, y) \in \sigma$ или $(x, z) \in \rho$ и $(z, y) \in \tau$. Первая альтернатива имеет место при $(x, y) \in \rho \circ \sigma$, а вторая — при $(x, y) \in \rho \circ \tau$, что означает $(x, y) \in \rho \circ \sigma \cup \rho \circ \tau$. Тем самым включение $\rho \circ (\sigma \cup \tau) \subseteq \rho \circ \sigma \cup \rho \circ \tau$ доказано.

Доказательство включения $\rho \circ \sigma \cup \rho \circ \tau \subseteq \rho \circ (\sigma \cup \tau)$ запишем коротко, используя логическую символику:

$$
\begin{aligned}
(x, y) \in \rho \circ \sigma \cup \rho \circ \tau \Rightarrow (\exists u)(((x, u) \in \rho) \wedge ((u, y) \in \sigma)) \vee \\
\vee (\exists v)(((x, v) \in \rho) \wedge ((v, y) \in \tau)) \Rightarrow \\
\Rightarrow (\exists z)(((x, z) \in \rho) \wedge (((z, y) \in \sigma) \vee ((z, y) \in \tau))) \Rightarrow \\
\Rightarrow (\exists z)(((x, z) \in \rho) \wedge ((z, y) \in \sigma \cup \tau)) \Rightarrow \\
\Rightarrow (x, y) \in \rho \circ (\sigma \cup \tau).
\end{aligned}
$$

В данном случае доказательства двух включений не совсем симметричны: элементы u и v во второй части доказательства не обязаны совпадать.

Замечание 1.4. В тождестве, выражающем свойство 3, нельзя вместо объединения поставить пересечение, так как в этом случае тождество нарушится. Можно доказать, что сохранится лишь включение

$$\rho \circ (\sigma \cap \tau) \subseteq \rho \circ \sigma \cap \rho \circ \tau,$$

а обратное включение в общем случае не имеет места. #

Анализ свойств 2 и 4 показывает, что роль *пустого соответствия* аналогична роли нуля при умножении чисел, а *диагональ множества* A играет роль, аналогичную роли единицы, на множестве всех бинарных отношений на A.

Обратное соответствие. ***Соответствие, обратное*** к соответствию $\rho \subseteq A \times B$, есть соответствие из B в A, обозначаемое ρ^{-1} и равное, по определению, $\rho^{-1} = \{(y, x) : (x, y) \in \rho\}$.

Для соответствия τ из примера 1.3

$$\tau^{-1} = \{(n_1, И), (n_2, П), (n_2, С), (n_3, И), (n_4, П), (n_5, И), (n_5, С)\}.$$

Обратное соответствие обладает следующими легко проверяемыми свойствами:

1) $(\rho^{-1})^{-1} = \rho$;
2) $(\rho \circ \sigma)^{-1} = \sigma^{-1} \circ \rho^{-1}$.

Для бинарного отношения ρ на множестве A обратное соответствие есть бинарное отношение на том же множестве. В этом случае говорят о ***бинарном отношении*** ρ^{-1} ***на множестве*** A, ***обратном*** к ρ.

Заметим, что соответствия $\rho \circ \rho^{-1}$ и $\rho^{-1} \circ \rho$ в общем случае не совпадают. Даже для бинарного отношения ρ на множестве A $\rho \circ \rho^{-1} \neq \rho^{-1} \circ \rho$, а также $\rho \circ \rho^{-1} \neq \mathrm{id}_A$ и $\rho^{-1} \circ \rho \neq \mathrm{id}_A$.

Например, для бинарного отношения $\rho = \{(3,1), (4,1), (4,2)\}$ на множестве $A = \{1, 2, 3, 4\}$ графы самого отношения, обратного отношения ρ^{-1}, композиций $\rho \circ \rho^{-1}$ и $\rho^{-1} \circ \rho$ представлены на рис. 1.5.

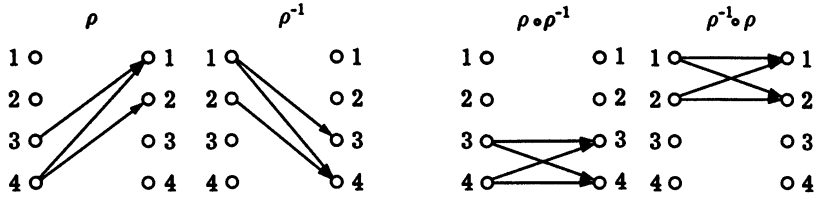


Рис. 1.5

Если $f\colon A \to B$ — отображение, то оно является соответствием. Обратное к f соответствие из B в A в общем случае не является отображением. Действительно, соответствие f^{-1}, обратное к f, состоит из всех упорядоченных пар вида $(f(x), x)$, $x \in A$. Поскольку в общем случае могут найтись такие два различных элемента x и x', что $f(x) = f(x')$, то соответствие f^{-1} в общем случае не будет *функционально по второй компоненте* и поэтому не будет отображением. Если *отображение f инъективно*, то обратное соответствие есть *частичное отображение* из B в A. Если *отображение f биективно*, то обратное соответствие является отображением из B в A, причем имеют место равенства

$$f \circ f^{-1} = \mathrm{id}_A, \quad f^{-1} \circ f = \mathrm{id}_B.$$

Отображение f^{-1} в этом случае называют *отображением, обратным* к f.

Ограничение соответствия. Пусть $\rho \subseteq A \times B$ — соответствие из A в B и $C \subseteq A$, $D \subseteq B$. ***Ограничением соответствия ρ на подмножества C и D*** (или ***(C, D)-ограничением*** соответствия ρ) называется соответствие из C в D, обозначаемое $\rho|_{C,D}$, такое, что

$$(x, y) \in \rho|_{C,D} \Leftrightarrow ((x, y) \in \rho) \wedge (x \in C) \wedge (y \in D).$$

Таким образом, (C, D)-ограничение соответствия ρ есть „то же самое“ соответствие ρ, но из последнего берутся только

упорядоченные пары, первая компонента которых принадлежит подмножеству C, а вторая — подмножеству D. Можно записать

$$\rho|_{C,D} = \rho \cap (C \times D).$$

Так, „малый“ арксинус, т.е. функция $y = \arcsin x$, есть ограничение „большого“ арксинуса $y = \operatorname{Arcsin} x$, который является соответствием на подмножества $[-1, 1]$ и $[-\frac{\pi}{2}, \frac{\pi}{2}]$.

Рассмотрим некоторые важные частные случаи ограничений соответствий (в частности, бинарных отношений и отображений).

Всякое (C, B)-ограничение соответствия $\rho \subseteq A \times B$ будем называть ***сужением соответствия ρ на подмножество*** C (коротко — ***C-сужением соответствия*** ρ), а всякое $(C, \rho(C))$-ограничение соответствия ρ — ***строгим сужением соответствия ρ на подмножество*** C (***строгим C-сужением соответствия*** ρ). C-сужения соответствия ρ будем обозначать $\rho|_C$, а строгое сужение — $\rho|_{\circ C}$ соответственно.

Полезно заметить, что для любого отображения $f\colon A \to B$ строгое сужение $f|_{\circ A}$ есть сюръекция A на $f(A)$. Если, сверх этого, f является инъекцией, то $f|_{\circ A}$ есть биекция A на $f(A)$. Допуская некоторую вольность речи*, можно сказать, что любое отображение сюръективно отображает свою область определения на свою область значений, в частности, любая инъекция устанавливает *взаимно однозначное соответствие* между областью определения и областью значений. Так, функция $y = \sin x$ сюръективно отображает множество $\mathbb{R}$ всех действительных чисел на отрезок $[-1, 1]$, а любая показательная функция биективно отображает $\mathbb{R}$ на подмножество всех положительных действительных чисел.

Для бинарного отношения $\rho \subseteq A^2$ и любого подмножества $M \subseteq A$ (M, M)-ограничение бинарного отношения называют

*Вольность состоит в том, что мы отождествляем функцию f с функцией $f|_{\circ A}$.

ограничением бинарного отношения ρ на подмножество M и обозначают $\rho|_M$. Можно записать $\rho|_M = \rho \cap M^2$.

Рассмотрим, например, отношение естественного порядка $\leqslant$ на множестве действительных чисел [I]. Тогда отношение $\leqslant|_{\mathbb{Z}} = \{(m, n)\colon m \leqslant n;\ m,n \in \mathbb{Z}\}$ есть ограничение этого порядка на подмножество целых чисел. Но ни в коем случае нельзя путать это отношение с $\mathbb{Z}$-сужением отношения $\leqslant$! Это последнее состоит из всех таких упорядоченных пар (m, x), что $m \in \mathbb{Z}$, $x \in \mathbb{R}$ и $m \leqslant x$, т.е. вторая компонента пары может быть произвольным действительным числом, не меньшим заданного целого m.

1.5. Семейства множеств

Рассматриваемое ниже понятие семейства множеств обобщает аналогичное понятие, сформулированное в [I].

Пусть U — ***универсальное множество***. Если каждому натуральному числу n взаимно однозначно сопоставлено некоторое подмножество $A_n \subseteq U$, то тем самым определена последовательность множеств $A_1, \ldots, A_n, \ldots$, или, в короткой записи, $(A_n)_{n\in\mathbb{N}}$. Предположим теперь, что вместо множества $\mathbb{N}$ натуральных чисел задано произвольное множество I и каждому элементу $i \in I$ взаимно однозначно сопоставлено подмножество $A_i \subseteq U$. Тогда говорят, что задано ***(индексированное) семейство множеств*** $(A_i)_{i\in I}$. Множество I называют множеством индексов, а множества A_i — элементами семейства $(A_i)_{i\in I}$.

В случае $I = \mathbb{N}$ получаем последовательность множеств, или ***счетное семейство множеств***; если множество I конечно, получаем ***конечное семейство множеств***. Таким образом, семейство $(A_i)_{i\in I}$ определено, если задано *отображение* $\nu\colon I \to 2^U$.

Отметим, что любое множество, элементы которого есть некоторые подмножества универсального множества U, т.е. любое множество $A \subseteq 2^U$, можно считать семейством $(A_i)_{i\in I}$,

где $I = A$, а ν — тождественное отображение множества A на себя.

Пример 1.11. Рассмотрим в качестве множества индексов множество точек некоторой гладкой плоской кривой [II] (рис. 1.6) и каждой точке сопоставим касательную, проведенную к кривой в этой точке (которая будет единственной в силу гладкости). Тогда получаем семейство множеств, элементами которого служат множества точек различных касательных. #

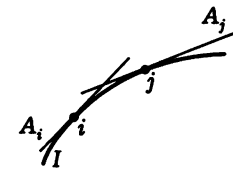


Рис. 1.6

Операции *объединения* и *пересечения множеств* можно распространить на произвольные семейства множеств [I].

1. Объединение семейства множеств:

$$\bigcup_{i\in I} A_i = \{x\colon (\exists i)(x \in A_i)\}.$$

2. Пересечение семейства множеств:

$$\bigcap_{i\in I} A_i = \{x\colon (\forall i)(x \in A_i)\}.$$

Методом двух включений можно доказать следующие тождества:

$$A \cup \Big(\bigcup_{i\in I} B_i\Big) = \bigcup_{i\in I}(A \cup B_i), \quad A \cap \Big(\bigcap_{i\in I} B_i\Big) = \bigcap_{i\in I}(A \cap B_i).$$

Аналогично можно доказать тождества

$$A \cap \Big(\bigcup_{i\in I} B_i\Big) = \bigcup_{i\in I}(A \cap B_i), \quad A \cup \Big(\bigcap_{i\in I} B_i\Big) = \bigcap_{i\in I}(A \cup B_i), \tag{1.5}$$

$$\overline{\bigcup_{i\in I} A_i} = \bigcap_{i\in I} \overline{A}_i, \qquad \overline{\bigcap_{i\in I} A_i} = \bigcup_{i\in I} \overline{A}_i. \tag{1.6}$$

Тождества (1.5) выражают свойство ***бесконечной дистрибутивности*** операций пересечения и объединения, а тождества (1.6) называют ***бесконечными законами де Моргана.***

1.6. Специальные свойства бинарных отношений

В этом параграфе дана определенная классификация *бинарных отношений на множестве.* В основе этой классификации лежат специальные свойства отношений.

Бинарное отношение ρ на множестве A называют ***рефлексивным***, если *диагональ* множества A *содержится в* ρ: $\mathrm{id}_A \subseteq \rho$, т.е. $x \rho x$ для любого элемента x множества A.

Если же $\mathrm{id}_A \cap \rho = \varnothing$, то бинарное отношение ρ на множестве A называют ***иррефлексивным.***

Указанные свойства бинарных отношений на множестве A называют ***рефлексивностью*** и ***иррефлексивностью.***

Бинарные отношения равенства и подобия на множестве геометрических фигур рефлексивны: каждый треугольник равен самому себе и подобен самому себе. На самом деле рефлексивны все отношения равенства: равенство чисел, равенство векторов, равенство множеств и т.п. Также рефлексивными являются, например, бинарное отношение нестрогого неравенства на множестве действительных чисел, поскольку для любого числа x всегда $x \leqslant x$, и ***отношение*** $\subseteq$ ***включения*** множеств, так как для любого множества X всегда $X \subseteq X$.

Напротив, бинарное отношение на множестве действительных чисел, задаваемое строгим неравенством $x < y$, иррефлексивно, равно как и отношение $\subset$ *строгого включения* множеств.

Не следует путать иррефлексивное отношение с нерефлексивным, т.е. не являющимся рефлексивным, отношением. Иррефлексивное отношение нерефлексивно, но не всякое нерефлексивное отношение иррефлексивно. Иррефлексивному отношению на A не принадлежит ни один элемент диагонали id_A, а нерефлексивное отношение может содержать некоторые (но не все!) элементы диагонали. На рис. 1.7 приведены примеры графиков иррефлексивного и нерефлексивного отношений (пунктиром указаны диагонали множеств).

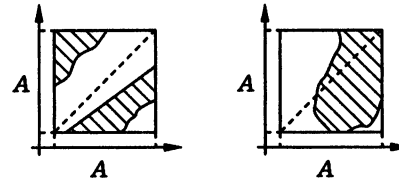


Рис. 1.7

Бинарное отношение ρ на множестве A называют:

1) ***симметричным***, если для любых $x, y \in A$ из $x \rho y$ следует $y \rho x$;

2) ***антисимметричным***, если для любых $x, y \in A$ из одновременной справедливости $x \rho y$ и $y \rho x$ следует, что $x = y$.

Соответствующие свойства бинарных отношений на множестве A называют ***симметричностью*** и ***антисимметричностью***.

График симметричного бинарного отношения на множестве A симметричен относительно диагонали (рис. 1.8).

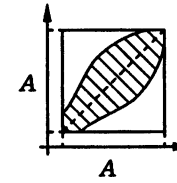


Рис. 1.8

Теорема 1.1. Бинарное отношение ρ на множестве A симметрично, если и только если *бинарное отношение на множестве* A, *обратное* к ρ, совпадает с ρ: $\rho^{-1} = \rho$.

◀ Пусть $(x, y) \in \rho^{-1}$, т.е. $(y, x) \in \rho$. Тогда, в силу симметричности ρ, $(x, y) \in \rho$. Следовательно, $\rho^{-1} \subseteq \rho$. Аналогично доказывается включение $\rho \subseteq \rho^{-1}$.

Теперь пусть $\rho = \rho^{-1}$. Тогда $(x, y) \in \rho$ и $(x, y) \in \rho^{-1}$. Из определения обратного отношения вытекает, что $(y, x) \in \rho$. Следовательно, ρ — симметричное бинарное отношение. ▶

Теорема 1.2. Бинарное отношение ρ на множестве A антисимметрично, если и только если $\rho \cap \rho^{-1} \subseteq \mathrm{id}_A$.

◀ Действительно, если $(x, y) \in \rho \cap \rho^{-1}$, то $(x, y) \in \rho$ и $(x, y) \in \rho^{-1}$ (т.е. $(y, x) \in \rho$). Но из выполнения соотношений $x \rho y$ и $y \rho x$ ввиду антисимметричности ρ следует, что $x = y$, т.е. $(x, y) \in \mathrm{id}_A$.

Обратно, пусть $\rho \cap \rho^{-1} \subseteq \mathrm{id}_A$. Предположим, что $(x, y) \in \rho$ и $(y, x) \in \rho$, причем $x \neq y$. Тогда $(x, y) \in \rho^{-1}$ и $(x, y) \in \rho \cap \rho^{-1}$, но $(x, y) \notin \mathrm{id}_A$. Получаем противоречие. ▶

Отметим, что для антисимметричного бинарного отношения на множестве A может иметь место равенство $\rho \cap \rho^{-1} = \varnothing$.

Все бинарные отношения в геометрии типа равенства или подобия симметричны. Так, если треугольник ABC подобен треугольнику $A'B'C'$, то и второй из этих треугольников подобен первому. Бинарные отношения неравенства чисел и включения множеств, как строгие, так и не строгие, антисимметричны.

Бинарное отношение ρ на множестве A называют ***транзитивным***, если для любых $x, y, z \in A$ из того, что $x \rho y$ и $y \rho z$, следует $x \rho z$. Соответствующее свойство бинарного отношения называют ***транзитивностью***.

Пример 1.12. **а.** Пусть M — некоторое множество населенных пунктов. Зададим на нем бинарное отношение достижимости: из пункта A достижим пункт B, если есть дорога, по которой можно доехать из A в B. Это отношение транзитивно, поскольку если из пункта A можно доехать до пункта B, а из B есть дорога до C, то из A можно проехать в C.

б. Бинарные отношения равенства и подобия в геометрии являются транзитивными: если треугольник ABC подобен треугольнику $A_1B_1C_1$, а этот последний подобен треугольнику $A_2B_2C_2$, то первый треугольник подобен третьему.

в. Бинарное отношение неравенства на множестве действительных чисел не транзитивно, так как из того, что $x \neq y$ и $y \neq z$, вовсе не следует, что $x \neq z$. Аналогично, если x друг y, а y друг z, то — вопреки известной поговорке — это не означает, что x друг z. #

Докажем следующее важное свойство транзитивного бинарного отношения.

Теорема 1.3. Бинарное отношение ρ на множестве A транзитивно тогда и только тогда, когда его *квадрат* содержится в нем, т.е. $\rho \circ \rho \subseteq \rho$.

◀ Пусть бинарное отношение ρ на множестве A транзитивно и $(x, z) \in \rho^2 = \rho \circ \rho$. В силу определения *композиции бинарных отношений на множестве* A существует такой элемент y, что $x \rho y$ и $y \rho z$, откуда ввиду транзитивности ρ получаем $x \rho z$, т.е. $(x, z) \in \rho$, а значит, $\rho^2 \subseteq \rho$.

Обратно, пусть бинарное отношение ρ на множестве A таково, что $\rho^2 \subseteq \rho$, а $(x, y) \in \rho$ и $(y, z) \in \rho$. Тогда в силу определения композиции бинарных отношений на множестве A $(x, z) \in \rho^2$. Поскольку $\rho^2 \subseteq \rho$, то $(x, z) \in \rho$. Таким образом, из того, что $(x, y) \in \rho$ и $(y, z) \in \rho$, следует, что $(x,z) \in \rho$, т.е. бинарное отношение ρ на множестве A транзитивно. ▶

Доказанное свойство целесообразно использовать для проверки транзитивности бинарного отношения ρ на некотором множестве в тех случаях, когда построение квадрата ρ является более легкой задачей по сравнению с исследованием свойства транзитивности ρ на основе определения.

Бинарное отношение ρ на множестве A называется ***плотным***, если для любых $x, y \in A$, отличных друг от друга и таких, что $x \rho y$, найдется z, отличный и от x и от y, такой, что $x \rho z$ и $z \rho y$.

Образно говоря, для любой пары *элементов*, *связанных* плотным *отношением*, всегда найдется третий элемент, который „встраивается между ними“ и связан с каждым из них тем же отношением. Так, отношения неравенства (строгого и нестрогого) на множествах рациональных и действительных чисел плотны, но аналогичные отношения на множествах целых и натуральных чисел плотными не являются. В самом деле, каковы бы ни были рациональные (или действительные) числа x и y, из того, что $x < y$, следует, что существует число z, отличное как от x, так и от y, такое, что $x < z < y$. Например,

подходит число $z = (x+y)/2$. Но для целых чисел m и $m+1$ такого „промежуточного" целого числа нет.

Если ρ — плотное бинарное отношение на множестве A и для некоторых $x, y \in A$ имеет место $x \rho y$, то найдется $z \in A$, такой, что $x \neq z$, $y \neq z$ и $x \rho z$, $z \rho y$. Отсюда в силу определения композиции отношений следует, что $x \rho^2 y$. Значит, из $(x, y) \in \rho$ следует $(x, y) \in \rho^2$, т.е. $\rho \subseteq \rho^2$.

Итак, если ρ плотно, то оно содержится в своем квадрате. Напомним, что для транзитивного бинарного отношения $\rho^2 \subseteq \rho$. Следовательно, если бинарное отношение ρ одновременно плотно и транзитивно, то $\rho = \rho^2$.

Среди всех бинарных отношений на произвольном множестве выделяют классы отношений в зависимости от свойств, которыми эти отношения обладают.

Бинарное отношение на некотором множестве называют:

1) ***эквивалентностью***, если оно рефлексивно, симметрично и транзитивно;

2) ***толерантностью***, если оно рефлексивно и симметрично;

3) ***порядком*** (или ***частичным порядком***), если оно рефлексивно, антисимметрично и транзитивно;

4) ***предпорядком*** (или ***квазипорядком***), если оно рефлексивно и транзитивно;

5) ***строгим порядком***, если оно иррефлексивно, антисимметрично и транзитивно;

6) ***строгим предпорядком***, если оно иррефлексивно и транзитивно.

Определенные выше бинарные отношения называют ***отношениями эквивалентности***, ***толерантности***, ***порядка*** (***частичного порядка***), ***предпорядка*** (***квазипорядка***), ***строгого порядка***, ***строгого предпорядка***.

Пример 1.13. **а.** Бинарное отношение параллельности двух прямых или двух плоскостей в евклидовой геометрии, если

считать каждую прямую (плоскость) параллельной самой себе, есть отношение эквивалентности.

б. Бинарное отношение ρ на множестве всех непустых подмножеств некоторого множества U, для которого $A \rho B$ тогда и только тогда, когда $A \cap B \neq \varnothing$, является толерантностью. Это отношение рефлексивно и симметрично, но не транзитивно. Действительно, из того, что $A \cap B \neq \varnothing$ и $B \cap C \neq \varnothing$, никак не следует, что $A \cap C \neq \varnothing$ (рис. 1.9).

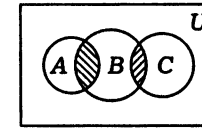


Рис. 1.9

в. Примером отношения порядка является ***естественный числовой порядок****, т.е. отношение неравенства на любом числовом множестве.

г. На множестве натуральных чисел $\mathbb{N}$ зададим бинарное отношение $a|b$, означающее, что a делит b (a является делителем b). Это отношение рефлексивно, поскольку любое число является делителем самого себя. Покажем антисимметричнсть. Пусть a делит b и в то же время b делит a. Тогда найдется натуральное число t_1, такое, что $b = at_1$, и найдется t_2, такое, что $a = bt_2$. Отсюда $b = bt_2t_1$, что на множестве натуральных чисел возможно только при $t_1 = t_2 = 1$. Следовательно, $a = b$. Покажем транзитивность. Если a делит b, а b делит c, то найдутся натуральные числа t_1, t_2, такие, что $b = at_1$ и $c = bt_2$. Отсюда имеем $c = at_1t_2$, т.е. a — делитель c. Таким образом, „отношение делимости“ на множестве $\mathbb{N}$ является отношением порядка.

Если распространить это отношение на множество целых чисел, то оно будет уже только предпорядком, поскольку теряется свойство антисимметричности. Например, 2 делится на -2 и -2 делится на 2, однако $2 \neq -2$.

*В [I] это отношение названо просто *естественным порядком*. Поскольку в дискретной математике нам приходится иметь дело со многими порядками на нечисловых множествах, мы все время будем говорить „естественный числовой порядок“, подчеркивая тем самым, что речь идет об отношении порядка на множестве действительных чисел (или об его *ограничении* на множества рациональных, целых или натуральных чисел).

д. Рассмотрим множество 2^A всех подмножеств множества A. Покажем, что отношение включения $\subseteq$ на множестве 2^A есть порядок. Это отношение рефлексивно, так как для любого множества X справедливо включение $X \subseteq X$. Поскольку для любых двух множеств X и Y из $X \subseteq Y$ и $Y \subseteq X$ следует, что $X = Y$, рассматриваемое отношение антисимметрично. Из определения включения вытекает, что если $X \subseteq Y$ и $Y \subseteq Z$, то $X \subseteq Z$. Следовательно, отношение транзитивно.

е. Отношение строгого неравенства на числовом множестве, равно как и отношение строгого включения множеств, есть отношение строгого порядка.

ж. В качестве примера отношения строгого предпорядка можно привести отношение „строгой достижимости“ на некотором множестве населенных пунктов: пункт A считаем строго достижимым из отличного от него пункта B, если есть дорога (автомобильная, железная и т.п.) из A в B, причем принимается, что никакой пункт не является строго достижимым из себя самого. #

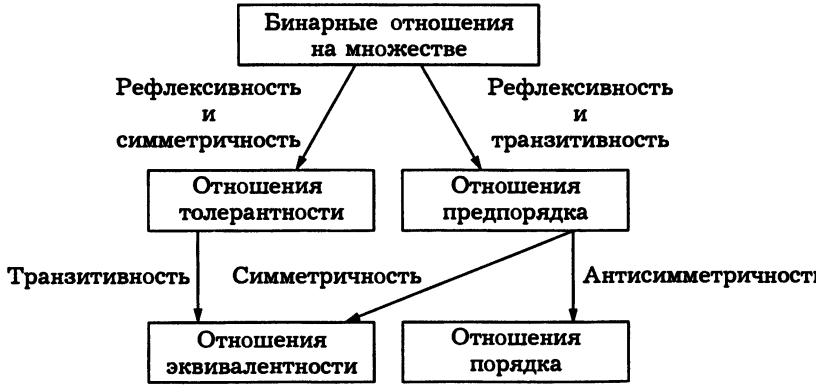


Рис. 1.10

Отношения толерантности, эквивалентности, предпорядка и порядка — важнейшие в современной математике. Связь

между этими четырьмя классами бинарных отношений показана на рис. 1.10. Можно сказать, что эквивалентность есть транзитивная толерантность или симметричный предпорядок. Порядок же есть антисимметричный предпорядок.

Для любого бинарного отношения $\rho \subseteq A^2$ можно построить отношение ρ^* следующим образом: $x\,\rho^*\,y$ тогда и только тогда, когда $x = y$ или существует последовательность x_0, x_1, ..., x_n, $n \geqslant 1$, такая, что $x_0 = x$, $x_n = y$ и для каждого $i = \overline{0, n-1}$ выполняется $x_i\,\rho\,x_{i+1}$. В частности, если $(x, y) \in \rho$, т.е. $x\,\rho\,y$, то это означает, что приведенное условие выполняется при $n = 1$. Следовательно, $(x, y) \in \rho^*$, т.е. $\rho \subseteq \rho^*$.

Отношение ρ^* называют ***рефлексивно-транзитивным замыканием*** бинарного отношения ρ на соответствующем множестве.

Можно также обозначить

$$\rho^0 = \mathrm{id}_A, \quad \rho^1 = \rho, \quad \rho^n = \rho \circ \rho^{n-1}, \quad n > 1,$$

и тогда

$$\rho^* = \bigcup_{i=0}^{\infty} \rho^i.$$

Отношение ρ^* является рефлексивным, так как $\mathrm{id}_A \subseteq \rho^*$. Докажем его транзитивность. Пусть для каких-то x, y, z выполняется $x\,\rho^*\,y$ и $y\,\rho^*\,z$. Докажем, что $x\,\rho^*\,z$. Будем считать, что элементы x, y, z попарно различны (так как при $x = y$ или $y = z$ доказывать нечего). Тогда существуют последовательности $x = x_0$, x_1, ..., $x_n = y$ и $y = y_0$, y_1, ..., $y_m = z$, такие, что $x_i\,\rho\,x_{i+1}$ для каждого $i = \overline{0, n-1}$ и $y_j\,\rho\,y_{j+1}$ для каждого $j = \overline{0, m-1}$ $(n, m \geqslant 1)$.

В итоге получаем последовательность z_0, ..., z_n, z_{n+1}, ..., z_{n+m}, где $z_0 = x$, $z_n = y$, $z_{n+m} = z$, $z_i = x_i$ для всякого $i = \overline{1, n-1}$, $z_{n+j} = y_j$ для всякого $j = \overline{1, m-1}$, такую, что $z_i\,\rho\,z_{i+1}$ для любого $i = \overline{0, n+m-1}$, т.е. $x\,\rho^*\,z$.

1.7. Отношения эквивалентности

Пусть A — произвольное множество. *Семейство* $(B_i)_{i\in I}$ непустых и попарно не пересекающихся *множеств* называют ***разбиением*** множества A, если *объединение множеств* семейства $(B_i)_{i\in I}$ равно A, т.е. $\bigcup\limits_{i\in I} B_i = A$.

Сами множества B_i называют ***элементами*** (или ***членами***) ***разбиения*** $(B_i)_{i\in I}$.

Например, множества $[0, 1/3)$, $[1/3, 2/3)$ и $[2/3, 1]$ образуют разбиение отрезка $[0, 1]$. ***Тривиальными разбиениями*** A являются, по определению, разбиение $\{A\}$, состоящее только из самого A, и разбиение, состоящее из всех одноэлементных подмножеств множества A.

Пусть ρ — *эквивалентность* на множестве A и $x \in A$. Множество всех элементов A, *эквивалентных* x, т.е. множество $\{y\colon y\,\rho\,x\}$, называют ***классом эквивалентности*** по отношению ρ и обозначают $[x]_\rho$. Отметим, что в силу рефлексивности для любого элемента $x \in A$ класс эквивалентности не пуст, так как $x \in [x]_\rho$.

Теорема 1.4. Для любого *отношения эквивалентности* на множестве A множество классов эквивалентности образует разбиение множества A. Обратно, любое разбиение множества A задает на нем отношение эквивалентности, для которого классы эквивалентности совпадают с элементами разбиения.

◀ Покажем, что отношение эквивалентности ρ на множестве A определяет некоторое разбиение этого множества. Убедимся вначале, что любые два класса эквивалентности по отношению ρ либо не пересекаются, либо совпадают.

Пусть два класса эквивалентности $[x]_\rho$ и $[y]_\rho$ имеют общий элемент $z \in [x]_\rho \cap [y]_\rho$. Тогда $z\,\rho\,x$ и $z\,\rho\,y$. В силу *симметричности* отношения ρ имеем $x\,\rho\,z$, и тогда $x\,\rho\,z$ и $z\,\rho\,y$. В силу *транзитивности* отношения ρ получим $x\,\rho\,y$. Пусть $h \in [x]_\rho$, тогда $h\,\rho\,x$. Так как $x\,\rho\,y$, то $h\,\rho\,y$ и, следовательно, $h \in [y]_\rho$.

Обратно, если $h \in [y]_\rho$, то в силу симметричности ρ получим $h \rho y$, $y \rho x$ и в силу транзитивности — $h \rho x$, т.е. $h \in [x]_\rho$. Таким образом, $[x]_\rho = [y]_\rho$.

Итак, любые два не совпадающих класса эквивалентности не пересекаются. Так как для любого $x \in A$ справедливо $x \in [x]_\rho$ (поскольку $x \rho x$), т.е. каждый элемент множества A принадлежит некоторому классу эквивалентности по отношению ρ, то множество всех классов эквивалентности по отношению ρ образует разбиение исходного множества A. Таким образом, любое отношение эквивалентности однозначно определяет некоторое разбиение.

Теперь пусть $(B_i)_{i \in I}$ — некоторое разбиение множества A. Рассмотрим отношение ρ, такое, что $x \rho y$ имеет место тогда и только тогда, когда x и y принадлежат одному и тому же элементу B_i данного разбиения:

$$x \rho y \Leftrightarrow (\exists i \in I)(x \in B_i) \wedge (y \in B_i).$$

Очевидно, что введенное отношение рефлексивно и симметрично. Если для любых x, y и z имеет место $x \rho y$ и $y \rho z$, то x, y и z в силу определения отношения ρ принадлежат одному и тому же элементу B_i разбиения. Следовательно, $x \rho z$ и отношение ρ транзитивно. Таким образом, ρ — эквивалентность на A. ▶

Теорема 1.4 позволяет отождествлять отношения эквивалентности и разбиения: любая эквивалентность определяет единственное разбиение и наоборот.

Множество всех классов эквивалентности по данному отношению эквивалентности ρ на множестве A называют ***фактор-множеством*** множества A по отношению ρ и обозначают A/ρ.

Пример 1.14. **а.** На множестве целых чисел $\mathbb{Z}$ определим ***отношение равенства по модулю*** k, где $k \in \mathbb{N}$. Положим* $x =_{(\mathrm{mod}\, k)} y$, если и только если $x - y$ делится на k.

*Традиционное обозначение: $m = n\,(\mathrm{mod}\, k)$.

Легко проверяется, что это отношение эквивалентности. Действительно, *рефлексивность* следует из того, что для любого $m \in \mathbb{Z}$ $m - m = 0$ и делится на k; симметричность — из того, что если $m - n$ делится на k, то и $n - m$ делится на k. Для доказательства транзитивности заметим, что если $m - n$ делится на k, $n - p$ делится на k, то и их сумма $(m - n) + (n - p) = m - p$ делится на k. Другими словами, для любых целых m, n, p из $m =_{(\mathrm{mod}\, k)} n$ и $n =_{(\mathrm{mod}\, k)} p$ следует $m =_{(\mathrm{mod}\, k)} p$, что доказывает транзитивность отношения $=_{(\mathrm{mod}\, k)}$.

Равенство чисел m и n по модулю k означает, что при делении на k эти числа дают одинаковые остатки. Действительно, для каждого $x \in \mathbb{Z}$ имеем $x = m \cdot k + r$, где r — остаток от деления x на k. Следовательно, $x - r = m \cdot k$, т.е. $x =_{(\mathrm{mod}\, k)} r$. Таким образом, каждое число попадает в тот же класс эквивалентности по отношению $=_{(\mathrm{mod}\, k)}$, что и остаток от деления его на k. Поскольку всего различных остатков может быть ровно k: 0, 1, ..., $k - 1$, получаем ровно k попарно различных классов эквивалентности по данному отношению: $[0]_{=_{(\mathrm{mod}\, k)}}$, $[1]_{=_{(\mathrm{mod}\, k)}}$, ..., $[k-1]_{=_{(\mathrm{mod}\, k)}}$, где класс $[r]_{=_{(\mathrm{mod}\, k)}}$ состоит из всех целых чисел, дающих при делении на k остаток r.

Отметим, что мы установили *взаимно однозначное соответствие* между фактор-множеством $\mathbb{Z}/=_{(\mathrm{mod}\, k)}$ и множеством $\mathbb{Z}_k$, состоящим из чисел 0, 1, ..., $k - 1$.

Второе множество дает нам как бы „наглядный образ“ построенного фактор-множества. Нельзя считать, что фактор-множество $\mathbb{Z}/=_{(\mathrm{mod}\, k)}$ равно множеству $\{0, 1, \ldots, k-1\}$. Нет, указанное фактор-множество состоит из k элементов, каждый из которых есть не число, а множество всех целых чисел, при делении на k дающих фиксированный остаток. Но каждому такому классу эквивалентности однозначно сопоставляется целое число от 0 до $k - 1$, и, наоборот, каждому целому числу от 0 до $k - 1$ соответствует единственный класс эквивалентности по отношению $=_{(\mathrm{mod}\, k)}$. Заметим, что в математике часто используется прием сопоставления фактор-множеству такого

находящегося с ним во взаимно однозначном соответствии множества, которое легко представить и описать.

б. На множестве $\mathbb{R}$ действительных чисел зададим отношение $a =_{(\mathrm{mod}\,1)} b$, полагая, что числа a и b равны по модулю 1 тогда и только тогда, когда число $a - b$ является целым. Из определения следует, что каждое число по модулю 1 равно своей дробной части*.

Так как отношение $=_{(\mathrm{mod}\,1)}$ определено через равенство, то легко понять, что все свойства отношения эквивалентности для него выполняются. Каждый класс эквивалентности будет содержать числа с равными дробными частями. Это значит, что каждый класс эквивалентности по данному отношению однозначно определяет некоторое число из полуинтервала $[0, 1)$ и, наоборот, каждому числу $\gamma \in [0, 1)$ однозначно сопоставляется класс эквивалентности, состоящий из всех действительных чисел, дробная часть которых равна γ. Таким образом, фактор-множество $\mathbb{R}/=_{(\mathrm{mod}\,1)}$ и полуинтервал $[0, 1)$ на числовой прямой находятся во взаимно однозначном соответствии. Этот полуинтервал можно рассматривать как представление определенного выше фактор-множества. #

Установим теперь связь между понятиями эквивалентности и отображения. Заметим, что для любого отношения эквивалентности ρ на множестве A можно определить отображение $f_\rho\colon A \to A/\rho$, положив $f_\rho(x) = [x]_\rho$, т.е. сопоставив каждому $x \in A$ содержащий его класс эквивалентности. Это *отображение сюръективно*, так как каждый элемент множества A принадлежит некоторому классу эквивалентности, т.е. для каждого $[x]_\rho \in A/\rho$ справедливо $[x]_\rho = f_\rho(x)$.

Отображение f_ρ, определенное таким образом, называют ***канонической сюръекцией*** множества A.

*Под дробной частью $<a>$ числа a понимается число из полуинтервала $[0, 1)$, такое, что $a = n + <a>$ для некоторого целого n. Поэтому дробной частью отрицательного числа $-a$, где $a > 0$, будет число $1 - <a>$. Так, дробной частью $-1{,}23$ будет не $0{,}23$, а $0{,}77 = 1 - 0{,}23$.

Покажем, что любое отображение однозначно определяет некоторое отношение эквивалентности.

Теорема 1.5. Пусть $f\colon A \to B$ — произвольное отображение. Отношение ρ_f на множестве A, для которого $x\,\rho_f\,y$, если и только если $f(x) = f(y)$, является отношением эквивалентности, причем существует биекция фактор-множества A/ρ_f на множество $f(A)$.

◀ Рефлексивность, симметричность и транзитивность отношения ρ_f следуют непосредственно из его определения, т.е. ρ_f — эквивалентность.

Зададим отображение φ фактор-множества A/ρ_f в множество $f(A)$ следующим образом: $\varphi([x]_{\rho_f}) = f(x)$. Из способа задания отношения ρ следует, что отображение определено корректно, т.е. каждому классу эквивалентности поставлен в соответствие единственный элемент $y \in f(A)$.

Докажем, что φ — биекция, для чего убедимся в том, что это инъекция и сюръекция одновременно. Пусть классы эквивалентности $[x]_{\rho_f}$ и $[y]_{\rho_f}$ не совпадают. В силу теоремы 1.4 это означает, что они не пересекаются, т.е. x не эквивалентно y. Из определения отношения ρ_f следует, что $f(x) \neq f(y)$. Таким образом, φ — инъекция. Если элемент $u \in f(A)$, то найдется такой элемент $x \in A$, что $u = f(x) = \varphi([x]_{\rho_f})$, т.е. φ — сюръекция фактор-множества A/ρ_f на множество $f(A)$. Итак, φ — биекция. ▶

Следовательно, в силу доказанных теорем 1.4 и 1.5 существует связь между тремя понятиями — отображением множества, отношением эквивалентности на множестве и разбиением множества. Но неверно, что существует взаимно однозначное соответствие между отображениями и отношениями эквивалентности*. Два разных отображения могут определять одно и то же разбиение отображаемого множества, тем самым задавая на нем одно и то же отношение эквивалентности. Так,

*Заметим, что теорема 1.5 этого и не утверждает.

например, любое биективное отображение $f\colon A \to B$ задает на A одно и то же разбиение — тривиальное разбиение на одноэлементные множества.

1.8. Упорядоченные множества. Теорема о неподвижной точке

Множество вместе с заданным на нем *отношением порядка* называют ***упорядоченным множеством.***

Отношение порядка будем, как правило, обозначать $\leqslant$ (или значками $\preccurlyeq$, $\sqsubseteq$ и т.п., похожими на $\leqslant$). При этом следует понимать, что даже на некотором множестве $S \subseteq \mathbb{R}$ рассматриваться может любое отношение порядка, а не только *естественный числовой порядок.* Множество M с заданным на нем отношением порядка $\leqslant$ будем записывать как пару $(M, \leqslant)$. Записывая $x \leqslant y$, мы будем говорить, что элемент x не больше элемента y.

Каждому отношению порядка $\leqslant$ на множестве M можно сопоставить следующие отношения.

1. Отношение, которое будем обозначать $<$, получается из исходного отношения порядка $\leqslant$ выбрасыванием всех элементов диагонали id_M, т.е. $x < y$ для любых $x, y \in M$ тогда и только тогда, когда $x \leqslant y$ и $x \neq y$. Записывая $x < y$, мы будем говорить, что элемент x строго меньше элемента y. Из определения следует, что отношение $<$ есть *иррефлексивное*, *антисимметричное* и *транзитивное* бинарное отношение на множестве M, т.е. оно является *отношением строгого порядка.*

2. ***Двойственный порядок.*** Это бинарное отношение на множестве M, называемое также и ***отношением, двойственным к отношению порядка*** $\leqslant$, определяется как *бинарное отношение на множестве M, обратное* к отношению $\leqslant$. Его обозначают $\geqslant$. Тогда для любых x, y условие $x \geqslant y$ равносильно тому, что $y \leqslant x$. Можно без труда доказать, что отношение $\geqslant$ тоже является отношением порядка.

Записывая $x \geqslant y$, мы будем говорить, что элемент x не меньше элемента y. Отношение строгого порядка, ассоциированное

с $\geqslant$, договоримся обозначать $>$, говоря при этом, что элемент x строго больше элемента y, если $x \geqslant y$ и $x \neq y$.

3. Отношение доминирования. Для двух элементов x и y, по определению, $x \lhd y$ тогда и только тогда, когда x строго меньше y и не существует такого элемента z, что $x < z < y$. Отношение $\lhd$ называют ***отношением доминирования*** (или просто ***доминированием***), ассоциированным с отношением порядка $\leqslant$. Если имеет место $x \lhd y$, то говорят, что элемент y доминирует над элементом x.

Из определения следует, что отношение доминирования иррефлексивно, антисимметрично, но не транзитивно. Оно может быть и пусто. Например, легко видеть, что пустым будет отношение доминирования, если исходный порядок является *плотным бинарным отношением на* соответствующем *множестве.*

Пример 1.15. а. Рассмотрим множество действительных чисел с естественным числовым порядком. Пусть $a < c$. Известно, что для любых a и c найдется такое b, что $a < b < c$, т.е. это отношение порядка на множестве действительных чисел является плотным. Поэтому отношение доминирования будет пустым.

По той же причине будет пустым отношение доминирования, ассоциированное с естественным числовым порядком на множестве рациональных чисел. Но на множестве целых чисел (опять-таки с естественным числовым порядком) отношение доминирования не пусто. Так, $1 \lhd 2$, $-5 \lhd -4$, но, конечно, неверно, что $1 \lhd 3$, поскольку между единицей и тройкой существует „промежуточный“ элемент — двойка.

б. На множестве всех подмножеств трехэлементного множества $\{a, b, c\}$, где в качестве отношения порядка взято отношение теоретико-множественного включения $\subseteq$, подмножество $\{a, b\}$ доминирует над подмножествами $\{a\}$ и $\{b\}$, но не доминирует над пустым множеством. В свою очередь, все множество

$\{a, b, c\}$ доминирует над любым своим двухэлементным подмножеством, но не доминирует над одноэлементным и над пустым.

в. По отношению делимости на множестве натуральных чисел 15 доминирует над 3 и 5, но 20 не доминирует над 5, так как существует „промежуточный" элемент — 10, делитель 20, который делится на 5, но не равен ни 20, ни 5. #

Рассмотрим упорядоченное множество $(M, \leqslant)$ и его произвольное непустое подмножество $B \subseteq M$. Упорядоченное множество $(B, \leqslant|_B)$, где $\leqslant|_B$ — *ограничение отношения* $\leqslant$ *на подмножество* B, называют ***упорядоченным подмножеством*** упорядоченного множества $(M, \leqslant)$.

Таким образом, можно переносить отношения порядка на непустые подмножества исходного упорядоченного множества. Как правило, вместо $\leqslant|_B$ будем писать просто $\leqslant$ (если ясно, о каком подмножестве B идет речь). Порядок $\leqslant|_B$ на подмножестве B называют также ***порядком, индуцированным*** исходным порядком $\leqslant$ на всем множестве A. Часто прибегают к такому выражению: „рассмотрим подмножество B упорядоченного множества $(M, \leqslant)$ с индуцированным порядком", понимая под этим порядок $\leqslant|_B$.

Элементы x и y упорядоченного множества $(M, \leqslant)$ называют ***сравнимыми*** по отношению порядка $\leqslant$, если $x \leqslant y$ или $y \leqslant x$. В противном случае элементы x и y называются ***несравнимыми***.

Упорядоченное множество, все элементы которого попарно сравнимы, называют ***линейно упорядоченным***, а соответствующее отношение — ***отношением линейного порядка*** (или просто ***линейным порядком***). Если индуцированный порядок на подмножестве упорядоченного множества является линейным, то это линейно упорядоченное подмножество называют ***цепью***. Любое подмножество попарно не сравнимых элементов данного упорядоченного множества называют ***антицепью***.

Замечание 1.5. Обратим внимание на то, что термину „упорядоченное множество“ (в смысле приведенного определения) в [I] отвечает термин „частично упорядоченное множество“, а то, что мы называем линейно упорядоченным множеством, в [I] называется просто упорядоченным множеством. Терминология этого выпуска более принята в алгебраической литературе и литературе по дискретной математике. Употребление в [I] термина „частично упорядоченное множество“ мотивировано желанием подчеркнуть, что в общем случае в упорядоченном множестве существуют не сравнимые элементы.

Пример 1.16. **а.** Отношение естественного числового порядка на множестве $\mathbb{R}$ действительных чисел является отношением линейного порядка, поскольку для любых двух чисел a, b имеет место или неравенство $a \leqslant b$, или неравенство $b \leqslant a$.

б. Отношение делимости (см. пример 1.13.г) на множестве $\mathbb{N}$ и отношение включения $\subseteq$ на $\mathcal{B}(A)$ (см. пример 1.13.д) не являются линейными порядками, за исключением случая, когда A — одноэлементное множество. #

Пусть $(A, \leqslant)$ — упорядоченное множество. Элемент $a \in A$ называют ***наибольшим элементом*** множества A, если для всех $x \in A$ выполняется неравенство $x \leqslant a$.

Элемент b называют ***максимальным элементом*** множества A, если для всякого $x \in A$ имеет место одно из двух: или $x \leqslant b$, или x и b не сравнимы.

Аналогично определяются ***наименьший*** и ***минимальный элементы*** упорядоченного множества, а именно: наименьший элемент упорядоченного множества A — это такой его элемент a, что $a \leqslant x$ для каждого $x \in A$, а минимальный элемент — это такой элемент $b \in A$, что для любого $x \in A$ элементы b и x не сравнимы или $b \leqslant x$.

Покажем, что наибольший (наименьший) элемент множества, если он существует, является единственным. Действительно, полагая, что a и a' — наибольшие элементы A по

отношению порядка $\leqslant$, получаем, что для всякого $x \in A$ выполняется $x \leqslant a$ и $x \leqslant a'$. В частности, $a' \leqslant a$ и $a \leqslant a'$, откуда ввиду антисимметричности любого отношения порядка следует, что $a = a'$. Аналогично доказывается единственность наименьшего элемента.

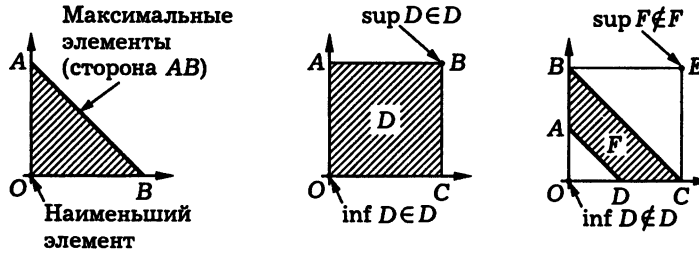
Замечание 1.6. Поскольку на одном и том же множестве могут быть определены разные отношения порядка (например, на множестве натуральных чисел — естественный числовой порядок и отношение делимости), то, когда это необходимо, мы будем говорить о наибольших, наименьших (соответственно максимальных и минимальных) элементах по данному отношению порядка, уточняя тем самым, о каком отношении порядка идет речь. #

Следующие примеры показывают, что максимальных (минимальных) элементов может быть сколько угодно*.

Пример 1.17. Рассмотрим множество точек плоскости с некоторой фиксированной *прямоугольной декартовой системой координат*. *Координаты* каждой *точки* плоскости задаются *упорядоченной парой* (x, y) действительных чисел. Отношение порядка на множестве точек плоскости определим следующим образом: $(a, b) \leqslant (c, d)$, если и только если $a \leqslant c$ и $b \leqslant d$. Рассмотрим множество точек треугольника OAB (рис. 1.11, *а*). Точка с координатами (0, 0) является наименьшим элементом этого множества. Максимальными элементами являются все точки, лежащие на стороне AB. Наибольшего элемента нет. #

Пусть $(A, \leqslant)$ — упорядоченное множество и $B \subseteq A$. Элемент $a \in A$ называется ***верхней*** (соответственно ***нижней***) ***гранью множества*** B, если для всех элементов $x \in B$ имеет место $(x \leqslant a)$ (соответственно $(x \geqslant a)$).

*Но заметим, что если у множества есть наибольший (соответственно наименьший) элемент, то он является единственным максимальным (соответственно минимальным) элементом данного множества.



а *б* *в*

Рис. 1.11

Наименьший элемент множества всех верхних граней (соответственно наибольший элемент множества всех нижних граней) множества B называют ***точной верхней гранью*** B (соответственно ***точной нижней гранью*** B) и обозначают $\sup B$ ($\inf B$).

Множество всех верхних (нижних) граней множества B называют ***верхним*** (***нижним***) ***конусом*** B и обозначают B^{∇} (соответственно B^{Δ}).

В отличие от наибольшего и наименьшего элементов множества B элементы $\sup B$ и $\inf B$ не обязаны принадлежать множеству B. Точная верхняя (нижняя) грань множества существует не всегда.

Пример 1.18. а. Рассмотрим множество D точек прямоугольника $OABC$ (рис. 1.11, *б*) с заданным в примере 1.17 отношением порядка. Точка O является точной нижней гранью, а точка B — точной верхней гранью этого множества. Обе точки принадлежат множеству.

Если рассмотреть множество F (рис. 1.11, *в*) с тем же отношением порядка, то увидим, что точная нижняя грань (точка O) и точная верхняя грань (точка E) множества F существуют, но не принадлежат множеству.

б. На числовой прямой с „выколотой“ точкой b для полуинтервала $[a, b)$ множество верхних граней есть $(b, +\infty)$, но точной верхней грани нет. #

Упорядоченное множество $(M, \leqslant)$ называют ***вполне упорядоченным***, если его любое непустое подмножество имеет наименьший элемент.

Множество натуральных чисел с отношением естественного числового порядка вполне упорядоченное. Множество целых чисел не вполне упорядоченное, поскольку оно не имеет наименьшего элемента. Аналогично множества рациональных и действительных чисел не являются вполне упорядоченными.

Можно показать, что справедлив ***принцип двойственности для упорядоченных множеств***. Пусть $(M, \leqslant)$ — произвольное упорядоченное множество. Тогда любое утверждение, доказанное для порядка $\leqslant$, останется справедливым для двойственного порядка $\geqslant$, если в нем:

1) порядок $\leqslant$ заменить на порядок $\geqslant$ и наоборот;

2) наименьший (минимальный) элемент заменить наибольшим (максимальным) элементом и наоборот;

3) inf заменить на sup и наоборот.

Например, если для некоторого $a \in M$ и для $B \subseteq M$ мы доказали, что $a = \sup B$ при заданном отношении порядка, то для двойственного порядка $a = \inf B$.

Говорят также и о ***взаимно двойственных определениях***: если в любом определении, связанном с упорядоченным множеством, произвести взаимные замены согласно принципу двойственности, то получится новое определение, называемое двойственным к исходному. Так, определение наибольшего (максимального) элемента множества двойственно к определению наименьшего (минимального) элемента, и наоборот. Часто употребляют оборот речи: „двойственным образом...“ (или „двойственно...“), понимая под этим переход к утверждению или определению, которое двойственно к исходному.

Рассмотрим теперь некоторые способы наглядного представления упорядоченных множеств.

Конечное упорядоченное множество можно графически изобразить в виде так называемой ***диаграммы Хассе***. На этой диаграмме элементы множества изображаются кружочками.

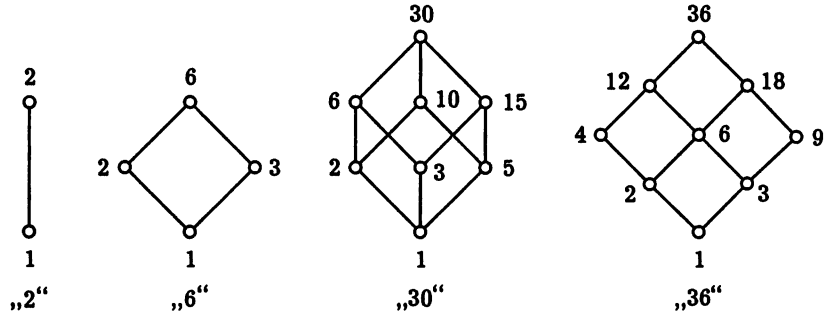


Рис. 1.12

При этом если элемент b доминирует над элементом a, то кружочек, изображающий элемент b, располагается выше кружочка, изображающего элемент a, и соединяется с ним прямой линией. Иногда для большей наглядности из a в b ведут стрелку. На рис. 1.12 изображены диаграммы Хассе для упорядоченных множеств делителей чисел 2, 6, 30 и 36 по рассмотренному выше отношению делимости (см. пример 1.13.г).

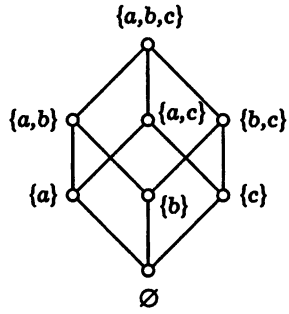


Рис. 1.13

На рис. 1.13 приведена диаграмма Хассе для упорядоченного множества всех подмножеств трехэлементного множества $\{a, b, c\}$ по отношению включения (см. пример 1.13.д).

Последовательность $\{x_i\}_{i\in\mathbb{N}}$ элементов упорядоченного множества называют ***неубывающей***, если для каждого $i \in \mathbb{N}$ справедливо неравенство $x_i \leqslant x_{i+1}$.

Элемент a упорядоченного множества $(M, \leqslant)$ называют ***точной верхней гранью последовательности*** $\{x_i\}_{i\in\mathbb{N}}$, если он есть точная верхняя грань множества всех членов последовательности*.

*Другими словами, точная верхняя грань последовательности есть точная верхняя грань области ее значений как функции натурального аргумента.

Двойственно определяется ***точная нижняя грань последовательности.***

Упорядоченное множество $(M, \leqslant)$ называют ***индуктивным***, если:

1) оно содержит наименьший элемент;

2) всякая неубывающая последовательность элементов этого множества имеет точную верхнюю грань.

Например, множество всех подмножеств некоторого множества по отношению включения будет индуктивным. Наименьший элемент — пустое множество, а точной верхней гранью произвольной неубывающей последовательности множеств будет объединение всех членов этой последовательности (наименьшее по включению множество, содержащее в качестве подмножества любой член последовательности).

Определение 1.5. Пусть $(M_1, \leqslant)$ и $(M_2, \preccurlyeq)$ — индуктивные упорядоченные множества. Отображение $f\colon M_1 \to M_2$ одного индуктивного упорядоченного множества в другое называют ***непрерывным***, если для любой неубывающей последовательности $a_1, \ldots, a_n, \ldots$ элементов множества M_1 образ ее точной верхней грани равен точной верхней грани последовательности образов $f(a_1), \ldots, f(a_n), \ldots$, т.е. справедливо равенство $f(\sup a_n) = \sup f(a_n)$.

Определение 1.6. Отображение $f\colon M_1 \to M_2$ упорядоченных множеств $(M_1, \leqslant)$ и $(M_2, \preccurlyeq)$ называют ***монотонным***, если для любых $a, b \in M_1$ из $a \leqslant b$ следует $f(a) \preccurlyeq f(b)$.

Теорема 1.6. Всякое непрерывное отображение одного индуктивного упорядоченного множества в другое монотонно.

◀ Пусть f — непрерывное отображение индуктивного упорядоченного множества $(M_1, \leqslant)$ в индуктивное упорядоченное множество $(M_2, \preccurlyeq)$. Пусть $a, b \in M_1$ и $a \leqslant b$. Образуем последовательность $\{x_n\}_{n\in\mathbb{N}}$, где $x_1 = a$, а $x_n = b$, $n \geqslant 2$. Эта последовательность неубывающая. Для нее $\sup x_n = \sup\{a, b\} = b$. В силу

непрерывности отображения f

$$f(b) = f(\sup x_n) = f(\sup\{a, b\}) = \sup\{f(a), f(b)\},$$

откуда следует, что $f(a) \preccurlyeq f(b)$. ▶

Заметим, что функция $f\colon \mathbb{R} \to \mathbb{R}$, непрерывная в смысле определений математического анализа, не обязана быть монотонным отображением упорядоченных множеств $\mathbb{R}$ с естественным числовым порядком, т.е. приведенное выше определение 1.5 непрерывности не вполне аналогично определению непрерывности в анализе [I]. Например, рассмотрим непрерывное в смысле определений математического анализа отображение $y = -x$ числовой прямой с естественным числовым порядком на себя. Это отображение не является монотонным в смысле данного выше определения 1.6, поскольку, например, $0 \leqslant 1$, однако неравенство $f(0) = 0 \leqslant f(1) = -1$ не выполняется.

В общем случае монотонное в смысле определения 1.6 отображение не является непрерывным в смысле определения 1.5. Приведем пример, показывающий, что утверждение, обратное теореме 1.6, неверно.

Пример 1.19. Рассмотрим множество всех точек отрезка $[0, 1]$ числовой прямой с порядком, индуцированным естественным числовым порядком. Это множество индуктивно: его наименьший элемент — 0, а любая неубывающая последовательность элементов ограничена сверху и по признаку Вейерштрасса [I] имеет предел, который и будет ее точной верхней гранью. Любая кусочно непрерывная (но не непрерывная!) и монотонная в смысле обычных определений из курса математического анализа функция [I], отображающая этот отрезок на любой отрезок с порядком, индуцированным естественным числовым порядком, дает пример монотонного в смысле определения 1.6, но не непрерывного в смысле определения 1.5 отображения между индуктивными частично упорядоченными множества-

ми. Например, пусть функция f имеет вид

$$f = \begin{cases} 0{,}5x, & 0 \leqslant x < 0{,}5; \\ 0{,}5 + 0{,}5x, & 0{,}5 \leqslant x \leqslant 1. \end{cases}$$

Это отображение монотонно. Для последовательности $\{x_n\} = \left\{\frac{n}{2n+1}\right\}$, $n \in \mathbb{N}$, точная верхняя грань равна 0,5. Точная верхняя грань последовательности $\{f(x_n)\}$ равна 0,25, а $f(\sup x_n) = f(0{,}5) = 0{,}75$. Следовательно, отображение не является непрерывным в смысле определения 1.5. #

Не следует путать отображение, монотонное в смысле определения 1.6, с монотонными функциями из курса математического анализа. Функция $f\colon \mathbb{R} \to \mathbb{R}$ будет монотонной в смысле определения 1.6 тогда и только тогда, когда она является неубывающей [I].

Для приложений особенно важны непрерывные отображения индуктивного упорядоченного множества в себя.

Определение 1.7. Элемент a множества A называют *неподвижной точкой отображения* $f\colon A \to A$, если $f(a) = a$.

Элемент a упорядоченного множества M называют ***наименьшей неподвижной точкой отображения*** $f\colon M \to M$, если он является наименьшим элементом множества всех неподвижных точек отображения f.

Теорема 1.7 (*теорема о неподвижной точке*). Любое непрерывное отображение f индуктивного упорядоченного множества $(M, \leqslant)$ в себя имеет наименьшую неподвижную точку.

◀ Обозначим через $\mathbb{O}$ наименьший элемент множества M. Полагаем $f^0(x) = x$ и $f^n(x) = f(f^{n-1}(x))$ для любого $n > 0$, т.е. $f^n(x)$ означает результат n-кратного применения f к x. Рассмотрим последовательность элементов M

$$\{f^n(\mathbb{O})\}_{n \geqslant 0} = \{\mathbb{O}, f(\mathbb{O}), \ldots, f^n(\mathbb{O}), \ldots\}. \tag{1.7}$$

Докажем, что последовательность (1.7) неубывающая. Используем метод математической индукции. Для элемента $\mathbb{O}$, как наименьшего элемента множества M, имеем $\mathbb{O} = f^0(\mathbb{O}) \leqslant \leqslant f(\mathbb{O})$. Пусть для некоторого натурального n верно соотношение $f^{n-1}(\mathbb{O}) \leqslant f^n(\mathbb{O})$. Согласно теореме 1.6, отображение f монотонно, и поэтому $f^n(\mathbb{O}) = f(f^{n-1}(\mathbb{O})) \leqslant f(f^n(\mathbb{O})) = f^{n+1}(\mathbb{O})$, т.е. соотношение верно и для номера $n+1$. Согласно методу математической индукции, $f^n(\mathbb{O}) \leqslant f^{n+1}(\mathbb{O})$ для любого $n \in \mathbb{N}_0$, т.е. последовательность (1.7) неубывающая. Следовательно, по определению индуктивного упорядоченного множества, она имеет точную верхнюю грань. Обозначим ее через a:

$$a = \sup_{n \geqslant 0} f^n(\mathbb{O}). \tag{1.8}$$

Докажем теперь, что если у неубывающей последовательности отбросить любое конечное число начальных членов, то ее точная верхняя грань не изменится.

Действительно, если a есть точная верхняя грань неубывающей последовательности $\{x_n\}_{n \geqslant 0}$, то $a \geqslant x_n$ для всякого $n \geqslant 0$. В частности, фиксируя произвольно $k > 0$, для любого $n \geqslant k$ имеем $a \geqslant x_n$, т.е. a будет верхней гранью подпоследовательности $\{x_n\}_{n \geqslant k > 0}$.

Докажем, что a является точной верхней гранью этой подпоследовательности. Пусть b — какая-то ее верхняя грань, т.е. $b \geqslant x_n$ для любого $n \geqslant k$. Так как последовательность $\{x_n\}_{n \geqslant 0}$ неубывающая, то $x_p \leqslant x_k$ для каждого $p = \overline{0, k-1}$. Поскольку $x_k \leqslant b$, то в силу транзитивности отношения порядка $x_p \leqslant b$ и тем самым $b \geqslant x_n$ для любого $n \geqslant 0$, т.е. b есть верхняя грань всей последовательности $\{x_n\}_{n \geqslant 0}$. Поскольку $a = \sup\limits_{n \geqslant 0} x_n$, то $a \leqslant b$ и $a = \sup\limits_{n \geqslant k > 0} x_n$. Следовательно, a — точная верхняя грань подпоследовательности $\{x_n\}_{n \geqslant k}$.

В силу непрерывности f получаем

$$f(a) = f\Big(\sup_{n \geqslant 0} f^n(\mathbb{O})\Big) = \sup_{n \geqslant 0} f\big(f^n(\mathbb{O})\big) = \sup_{n \geqslant 0} f^{n+1}(\mathbb{O}).$$

Но

$$\sup_{n\geqslant 0} f^{n+1}(\mathbb{O}) = \sup\{f^1(\mathbb{O}), f^2(\mathbb{O}), \ldots\} = \sup_{n\geqslant 1} f^n(\mathbb{O}) = a.$$

Таким образом, доказано, что a является неподвижной точкой отображения f.

Покажем теперь, что найденная неподвижная точка является наименьшей. Пусть для некоторого $y \in M$ $f(y) = y$. Так как $\mathbb{O} \leqslant y$, а отображение f, будучи непрерывным, монотонно, то $f(\mathbb{O}) \leqslant f(y) = y$, $f(f(\mathbb{O})) \leqslant f(f(y)) = y$ и т.д. Следовательно, для любого $n \geqslant 0$ $f^n(\mathbb{O}) \leqslant y$, т.е. элемент y есть верхняя грань последовательности $\{f^n(\mathbb{O})\}_{n\geqslant 0}$. Поскольку элемент a (как точная верхняя грань) есть наименьший элемент на множестве всех верхних граней этой последовательности, то $y \geqslant a$. Таким образом, мы доказали, что произвольная неподвижная точка отображения f не меньше элемента a, т.е. a — наименьшая неподвижная точка отображения f. ▶

Поиск неподвижной точки отображения $f\colon M \to M$ можно рассматривать как задачу решения уравнения

$$x = f(x). \tag{1.9}$$

Теорему о неподвижной точке можно трактовать таким образом: для непрерывного отображения f индуктивного упорядоченного множества в себя уравнение $x = f(x)$ имеет решение, т.е. существует такой $x_0 \in M$, что выполняется равенство $x_0 = f(x_0)$, причем множество всех решений этого уравнения имеет наименьший элемент. Этот элемент и будет наименьшей неподвижной точкой отображения f.

Отметим, что доказательство теоремы о неподвижной точке конструктивное: оно дает метод получения неподвижной точки. Для ее нахождения надо построить последовательность вида (1.7) и найти ее точную верхнюю грань.

Пример 1.20. Приведем простой пример вычисления наименьшей неподвижной точки. В качестве индуктивного упорядоченного множества M возьмем отрезок $[0, 1]$ с естественным

числовым порядком $\leqslant$. Согласно примеру 1.19, это индуктивное упорядоченное множество.

Рассмотрим на этом множестве уравнение $x = \frac{1}{2}x + \frac{1}{4}$. Можно показать, что для индуктивного упорядоченного множества M любая монотонная функция $f\colon [0,1] \to [0,1]$, непрерывная в смысле определения из курса математического анализа, непрерывна и в смысле данного выше определения. Действительно, для любой неубывающей последовательности $\{x_n\}$ на множестве $[0,1]$ справедливо равенство $\sup\{x_n\} = \lim\limits_{n\to\infty} x_n$ [I]. В силу непрерывности функции f в смысле определения из курса математического анализа имеем $f\big(\lim\limits_{n\to\infty} x_n\big) = \lim\limits_{n\to\infty} f(x_n)$. Так как функция f монотонна, то $\{f(x_n)\}_{n\geqslant 0}$ — неубывающая последовательность и $\lim\limits_{n\to\infty} f(x_n) = \sup f(x_n)$. В итоге получаем $f(\sup x_n) = f\big(\lim\limits_{n\to\infty} x_n\big) = \lim\limits_{n\to\infty} f(x_n) = \sup f(x_n)$. Следовательно, правая часть данного уравнения непрерывна.

Заметим, что если бы в правой части стояла линейная функция с отрицательным коэффициентом при x, то условие непрерывности функции в смысле приведенного выше определения было бы уже нарушено, поскольку такая функция не является монотонной в смысле определения 1.6.

Наименьшим элементом рассматриваемого множества является число 0. Вычисляем:

$$
\begin{aligned}
&f^0(0) = 0,\\
&f^1(0) = 1/4,\\
&f^2(0) = (1/2)\cdot(1/4) + 1/4 = 3/8,\\
&f^3(0) = (1/2)\cdot(3/8) + 1/4 = 7/16,\\
&\ldots\ldots\ldots\ldots\ldots\ldots\ldots
\end{aligned}
$$

получая последовательность приближений к наименьшей неподвижной точке.

Можно проверить (например, с помощью метода математической индукции), что $f^n(0) = \frac{2^n - 1}{2^{n+1}}$, $n \in \mathbb{N}$. Предел этой

последовательности равен 1/2. Таким образом, наименьшая неподвижная точка отображения f, определяемого правой частью уравнения, равна 1/2. Это единственная в данном случае неподвижная точка отображения f, что, конечно, очевидно и без теоремы 1.7 о неподвижной точке. Но здесь мы нарочно рассмотрели простой пример, показав, как можно решать подобные уравнения, основываясь на доказательстве теоремы. Мы не будем пока приводить более сложные примеры, так как интересные приложения теоремы о неподвижной точке имеются в теории графов и теории автоматов, и вернемся к этой теореме при решении задачи о путях в ориентированных графах.

1.9. Мощность множества

Некоторые сведения о мощности множества приведены в [I]. Здесь мы рассмотрим это понятие более подробно.

Множество A равномощно множеству B, если существует *биекция* $f\colon A \to B$.

Из того, что существует биекция $f\colon A \to B$, следует, что соответствие f^{-1} есть биекция B на A (см. **1.4**). Поэтому если A равномощно B, то и B равномощно A, и мы можем говорить, что ***множества*** A и B ***равномощны***.

Факт равномощности множеств A и B будем записывать так: $A \sim B$.

Из определения равномощности и свойств биекции также следует, что для любого множества A имеет место $A \sim A$ (тождественное отображение есть биекция множества A на себя); а для любых множеств A, B, C из $A \sim B$ и $B \sim C$ следует $A \sim C$ (*композиция* биекций есть биекция).

Таким образом, отношение равномощности множеств есть *отношение эквивалентности**, заданное на „множестве всех

*Зачастую в литературе по теории множеств равномощные множества и называют „эквивалентными множествами“. Однако следует различать общее понятие отношения эквивалентности и его частный случай — эквивалентность, или равномощность, множеств.

множеств“ (на самом деле на множестве всех подмножеств некоторого *универсального множества*).

Если мы обозначим через $|A|$ *класс эквивалентности* множества A по отношению $\sim$, то утверждение о равномощности множеств A и B можно записать так: $|A| = |B|$.

Этот класс эквивалентности $|A|$ называют ***мощностью множества*** A.

Конечные множества $A = \{a_1, \ldots, a_n\}$ и $B = \{b_1, \ldots, b_m\}$ равномощны тогда и только тогда, когда множества A и B состоят из одного и того же числа элементов, т.е. $n = m$. Отсюда, в частности, следует, что конечное множество не является равномощным никакому своему собственному подмножеству. Это свойство конечных множеств можно сформулировать так.

Теорема 1.8. Если A — конечное множество и $f\colon A \to A$ — инъекция, то она является сюръекцией и, следовательно, биекцией. #

В силу приведенных выше соображений мощностью конечного множества $A = \{a_1, \ldots, a_n\}$ можно считать натуральное число n, так как, задавая такое число, мы задаем и класс всех (попарно равномощных) множеств вида $\{a_1, \ldots, a_n\}$. Обратно, каждый такой класс однозначно определяет натуральное число n как число элементов в каждом множестве данного класса. Естественно считается, что мощность пустого множества равна нулю.

Перейдем теперь к исследованию мощности бесконечных множеств. Таковы хорошо известные нам числовые множества $\mathbb{N}$, $\mathbb{Z}$, $\mathbb{Q}$, $\mathbb{R}$ и $\mathbb{C}$.

Любое множество, равномощное множеству всех натуральных чисел, называют *счетным*. Мощность счетного множества обозначают $\aleph_0$ (читается „алеф нуль“).

Любую биекцию $\nu\colon \mathbb{N} \to M$ называют ***нумерацией*** счетного множества M; если элемент M есть $\nu(n)$ для некоторого $n \in \mathbb{N}$, то этот элемент M обозначаем через a_n, называя на-

туральное число n номером элемента a_n относительно данной нумерации ν.

Таким образом, элементы счетного множества можно перенумеровать, записав их в виде *последовательности* $a_1, \ldots, a_n, \ldots$ или $\{a_n\}_{n\in\mathbb{N}}$. Другими словами, счетное множество есть область значений некоторой функции натурального аргумента.

Пример 1.21. **а.** Множество всех нечетных натуральных чисел счетно. Нумерацию ν можно задать так: $\nu(n) = 2n - 1$, $n \in \mathbb{N}$.

б. Множество всех натуральных чисел, делящихся на заданное число $k \geqslant 2$, счетно. Нумерацию ν можно задать так: $\nu(n) = kn$, $n \in \mathbb{N}$. В частности, при $k = 2$ получаем, что множество всех четных чисел счетно. Этот и предыдущий примеры показывают, что бесконечное (счетное) множество может иметь собственное равномощное ему подмножество.

в. Множество $\mathbb{Z}$ всех целых чисел счетно. Нумерацию можно установить так:

$$\nu(n) = \begin{cases} \frac{n}{2} - 1, & n = 2k \text{ (четно)}; \\ -\frac{n+1}{2}, & n = 2k-1 \text{ (не четно)}. \end{cases} \quad \#$$

Рассмотрим свойства счетных множеств.

Теорема 1.9. Любое бесконечное множество содержит счетное подмножество.

◀ Пусть M_0 — бесконечное множество. Значит, оно не пусто и существует элемент $a_1 \in M_0$. Положим $M_1 = M_0 \setminus \{a_1\}$. Множество M_1 не пусто, так как в противном случае имело бы место равенство $M_0 = \{a_1\}$, что противоречит предположению о бесконечности M_0. Выберем элемент $a_2 \in M_1$ и положим $M_2 = M_1 \setminus \{a_2\} = M_0 \setminus \{a_1, a_2\}$. Множество M_2 также не пусто, поскольку иначе мы бы имели $M_0 = \{a_1, a_2\}$ и множество M_0 было бы конечным.

Методом математической индукции можно показать, что для любого $n \geqslant 1$ множество $M_n = M_0 \setminus \{a_1, \ldots, a_n\}$, где $a_1 \in M_0$, ..., $a_n \in M_{n-1}$, не пусто. Тогда существует элемент $a_{n+1} \in M_n$ и $a_{n+1} \notin M_{n+1} = M_n \setminus \{a_{n+1}\}$. Таким образом, все элементы a_n, $n \in \mathbb{N}$, попарно различны и множество $\{a_n\colon n \in \mathbb{N}\}$ счетно, а его нумерация может быть задана так: $\nu(n) = a_n$. ▶

Теорема 1.10. В любом бесконечном множестве можно выделить два не пересекающихся между собой счетных подмножества.

◀ Разобьем множество натуральных чисел на два подмножества: $\mathbb{N}_1 = \{n\colon n = 2k-1, k \in \mathbb{N}\}$ (множество нечетных чисел), и $\mathbb{N}_2 = \{n\colon n = 2k, k \in \mathbb{N}\}$ (множество четных чисел). Оба эти множества счетны (см. пример 1.21).

Согласно теореме 1.9, бесконечное множество содержит некоторое счетное подмножество A. Пусть установлена некоторая его нумерация. Разобьем это подмножество на два подмножества: всех элементов с четными и всех элементов с нечетными номерами. По построению эти подмножества не пересекаются и являются счетными, поскольку счетны множества четных и нечетных чисел. ▶

Теорема 1.11. Любое подмножество счетного множества конечно или счетно.

◀ Пустое подмножество конечно по определению. Пусть M — счетное множество, а B — его некоторое непустое подмножество. Поскольку множество M счетно, можно считать, что задана некоторая его нумерация. Следовательно, каждый элемент подмножества B имеет свой номер. Запишем номера элементов множества B в порядке возрастания: i_1, ..., i_n, ... Если среди них есть наибольший номер i_p, то подмножество B конечно. В противном случае получим счетное подмножество $\{a_{i_1}, a_{i_2}, \ldots, a_{i_n}, \ldots\}$, нумерация которого установлена так: $\nu(n) = a_{i_n}$. ▶

Если множество конечно или счетно, его называют ***не более чем счетным***. *Семейство* $(A_i)_{i\in I}$ *множеств* называют ***не более чем счетным***, если множество (индексов) I не более чем счетно.

Теорема 1.12. Объединение любого не более чем счетного семейства счетных множеств счетно.

◀ Пусть $(A_i)_{i\in I}$ — конечное или счетное семейство счетных множеств. Рассмотрим сначала случай, когда множества A_i попарно не пересекаются.

В этом случае нумерация объединения конечного семейства счетных множеств может быть проведена по схеме, изображенной на рис. 1.14, а нумерация объединения счетного семейства счетных множеств — по схеме, приведенной на рис. 1.15.

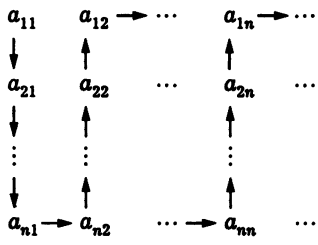


Рис. 1.14

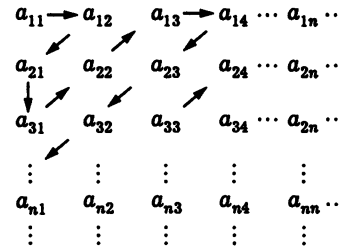


Рис. 1.15

Пусть теперь $(A_n)_{n\in\mathbb{N}}$ — произвольное счетное семейство счетных множеств, т.е. множества A_i могут пересекаться. В этом случае, применяя указанные на рис. 1.14 и 1.15 схемы нумерации к конечному или счетному объединению счетных множеств, следует пропускать каждый раз элементы, которые уже получили номера. ▶

Полезно отметить также и следующий факт.

Теорема 1.13. Объединение конечного и счетного множеств счетно. #

Теорема 1.14. Пусть M — бесконечное множество, а N — его не более чем счетное подмножество. Если множество $M \setminus N$ бесконечно, то оно равномощно множеству M.

◀ По теореме 1.10 в множестве $M \setminus N$, поскольку оно бесконечно, можно выбрать счетное подмножество N'. Ясно, что $N \cap N' = \varnothing$. Множество $N \cup N'$ является счетным как объединение двух счетных множеств или объединение счетного и конечного множеств. Поэтому существует биекция $f\colon N \cup N' \to N'$. Поскольку $(M \setminus (N \cup N')) \cup (N \cup N') = M$, $M \setminus (N \cup N') \cup N' = M \setminus N$, то требуемую биекцию M на $M \setminus N$ строим так: на подмножестве $M \setminus (N \cup N')$, общем для $M \setminus N$ и M, эта биекция совпадает с тождественным отображением; на подмножестве $N \cup N'$ эта биекция есть биекция f. ▶

Следствие 1.1. Если M — бесконечное множество, а N — не более чем счетное множество, то $M \sim M \cup N$.

Существуют бесконечные множества, не являющиеся счетными. Это вытекает из следующих рассуждений.

Рассмотрим множество всех последовательностей нулей и единиц, т.е. последовательностей вида $\{\alpha_1, \alpha_2, \ldots, \alpha_n, \ldots\}$, где $\alpha_i \in \{0, 1\}$ для каждого $i \geqslant 1$.

Обозначим множество таких „двоичных“ последовательностей через $\{0, 1\}^\omega$.

Теорема 1.15 (*теорема Кантора*). Множество $\{0, 1\}^\omega$ не есть счетное множество.

◀ Пусть множество $\{0, 1\}^\omega$ счетное. Тогда существует биекция $\varphi\colon \mathbb{N} \to \{0, 1\}^\omega$. Выпишем все последовательности $\varphi(n)$:

$$\begin{aligned}
&\varphi(1) = \{\alpha_{11}, \alpha_{12}, \ldots, \alpha_{1n}, \ldots\},\\
&\varphi(2) = \{\alpha_{21}, \alpha_{22}, \ldots, \alpha_{2n}, \ldots\},\\
&\quad\ldots\ldots\ldots\ldots\ldots\\
&\varphi(n) = \{\alpha_{n1}, \alpha_{n2}, \ldots, \alpha_{nn}, \ldots\},\\
&\quad\ldots\ldots\ldots\ldots\ldots
\end{aligned}$$

Построим последовательность $\beta = \{\beta_1, \ldots, \beta_n, \ldots\}$: положим $\beta_i = 1$, если $\alpha_{ii} = 0$, и $\beta_i = 0$, если $\alpha_{ii} = 1$. Ясно, что эта последовательность не совпадает ни с одной из последовательностей $\varphi(n)$, а это противоречит допущению, что любая последовательность из $\{0,1\}^\omega$ есть $\varphi(k)$ для некоторого k.

Итак, $\mathbb{N}$ не равномощно $\{0,1\}^\omega$.

В то же время $\{0,1\}^\omega$ содержит подмножество последовательностей, в каждой из которых только один член отличен от нуля. Это подмножество равномощно множеству всех одноэлементных подмножеств $\mathbb{N}$ и, следовательно, самому $\mathbb{N}$. Следовательно, множество $\{0,1\}^\omega$ бесконечно, но не равномощно счетному множеству и потому не является счетным. ▶

Теорема 1.16. Множество $2^{\mathbb{N}}$ всех подмножеств множества натуральных чисел и множество $\{0,1\}^\omega$ равномощны.

◀ Определим отображение φ множества $2^{\mathbb{N}}$ на множество $\{0,1\}^\omega$ следующим образом: если $X \subseteq \mathbb{N}$, то $\varphi(X)_i = 1$ при $i \in X$ и $\varphi(X)_i = 0$ при $i \notin X$.

Тем самым подмножеству X ставится в соответствие последовательность $\varphi(X)$, i-й член которой равен единице тогда и только тогда, когда число i есть элемент множества X. Докажем, что φ — биекция $2^{\mathbb{N}}$ на $\{0,1\}^\omega$.

Покажем, что отображение φ — инъекция. Пусть X и Y — различные подмножества $\mathbb{N}$. Тогда найдется число $i \in X \setminus Y$ или число $j \in Y \setminus X$. В первом случае в последовательности $\varphi(X)$ i-й член будет равен единице, а в последовательности $\varphi(Y)$ — нулю. Таким образом, $\varphi(X) \neq \varphi(Y)$. Во втором случае $\varphi(Y)_j = 1$, $\varphi(X)_j = 0$ и опять $\varphi(X) \neq \varphi(Y)$. Следовательно, отображение φ — инъекция.

Покажем, что φ — сюръекция. Возьмем произвольную последовательность $\alpha \in \{0,1\}^\omega$. Образуем множество $X_\alpha = \{i\colon \alpha_i = 1\}$. Ясно, что $\varphi(X_\alpha) = \alpha$. Таким образом, для любой последовательности $\alpha \in \{0,1\}^\omega$ существует подмножество $X_\alpha \in 2^{\mathbb{N}}$, такое, что $\varphi(X_\alpha) = \alpha$. Следовательно, φ — сюръекция.

Таким образом, мы показали, что φ — биекция, а множества $2^{\mathbf{N}}$ и $\{0,1\}^{\omega}$ равномощны. ▶

Мощность множества $2^{\mathbf{N}}$ обозначают c и называют ***мощностью континуума***, а любое множество, эквивалентное множеству $2^{\mathbf{N}}$, называют ***множеством мощности континуума*** или ***континуальным множеством.***

Теорема 1.17. Множество действительных чисел отрезка $[0,1]$ равномощно множеству всех последовательностей нулей и единиц $\{0,1\}^{\omega}$.

◀ Каждое действительное число из отрезка $[0,1]$ представим в виде бесконечной дроби в двоичной системе счисления. Число 1 представим в виде периодической дроби, содержащей бесконечное число единиц — $0,1(1)$. Конечные рациональные дроби представим как бесконечные, дополнив справа бесконечным числом нулей. Таким образом, каждое число из $[0,1]$ представлено в виде последовательности нулей и единиц. Кроме этого, выбросим счетное множество всех периодических дробей вида $0,\alpha_0\alpha_1\ldots\alpha_k0(1)$, поскольку каждая такая дробь представляет то же самое число, что и дробь $0,\alpha_0\alpha_1\ldots\alpha_k1(0)$, где $\alpha_i\in\{0,1\}$ для всякого $i=\overline{1,k}$. Легко видеть, что полученное таким образом множество двоичных дробей равномощно множеству $\{0,1\}^{\omega}$. ▶

Следствие 1.2. $[0,1]\sim 2^{\mathbf{N}}$.

◀ Выше была доказана равномощность множеств $(0,1)^{\omega}$ и $2^{\mathbf{N}}$. Тогда имеем $[0,1]\sim\{0,1\}^{\omega}\sim 2^{\mathbf{N}}$. ▶

Теорема 1.18. Следующие множества равномощны:

1) множество действительных чисел отрезка $[0,1]$;
2) множество действительных чисел интервала $(0,1)$;
3) множество действительных чисел отрезка $[a,b]$;
4) множество действительных чисел интервала (a,b);
5) множество действительных чисел (числовая ось) $\mathbb{R}$;

6) множество всех подмножеств множества натуральных чисел $2^{\mathbf{N}}$.

◀ Покажем равномощность множеств $[0,1]$ и $(0,1)$. Из множества действительных чисел отрезка $[0,1]$ выделим двухэлементное подмножество $\{0,1\}$. Разностью этих множеств будет множество действительных чисел интервала $(0,1)$, и, согласно теореме 1.14, $[0,1] \sim (0,1)$.

Отображение $y = (b-a)x + a$ задает биекцию множества $[0,1]$ на множество $[a,b]$. Следовательно, эти множества равномощны. Заметим, что аналогично доказывается равномощность $(0,1)$ и (a,b).

Покажем, что $(0,1) \sim \mathbb{R}$. Биекцию можно установить, например, с помощью функции $y = \frac{1}{\pi}\operatorname{arctg} x + \frac{1}{2}$.

Поскольку равномощность $[0,1]$ и $2^{\mathbf{N}}$ ранее доказана, имеем

$$[0,1] \sim (0,1) \sim [a,b] \sim (a,b) \sim \mathbb{R} \sim 2^{\mathbf{N}}. \qquad \blacktriangleright$$

Замечание 1.7. Заметим, что если в условиях теоремы 1.14 множество M несчетно, а N — его счетное подмножество, то множество $M \setminus N$ бесконечно, ибо иначе получилось бы, что множество $M = (M \setminus N) \cup N$ счетно как объединение конечного и счетного множеств.

Таким образом, можно утверждать, что для любого несчетного множества M и его не более чем счетного подмножества N имеет место равенство $|M \setminus N| = |M|$. #

До сих пор речь шла о равенстве мощностей. Однако мощности разных множеств можно в определенном смысле сравнивать, говоря о большей или меньшей мощности.

Считают, что мощность множества A не превышает мощность множества B ($|A| \leqslant |B|$), если A равномощно некоторому подмножеству множества B. Можно показать, что из соотношений $|A| \leqslant |B|$ и $|B| \leqslant |A|$ следует, что $|A| = |B|$.

Мощность множества A считается строго меньшей мощности множества B ($|A| < |B|$), если множества A и B неравно-

мощны и существует собственное подмножество C множества B, равномощное множеству A, т.е. $(A \not\sim B)$ и $(\exists C \subset B)(A \sim C)$.

Можно доказать, что из $|A| \leqslant |B|$ и $|B| \leqslant |C|$ следует $|A| \leqslant |C|$. Таким образом, на множестве всех мощностей (т.е. на множестве всех классов эквивалентности по отношению $\sim$) установлено *отношение порядка*.

Из определения сразу следует, что мощность любого конечного множества строго меньше мощности $\aleph_0$, а из доказательства теоремы 1.15 вытекает, что $\aleph_0 < c$. Кроме того, согласно теореме 1.9, мощность счетного множества $\aleph_0$ является *наименьшей* на множестве всех бесконечных мощностей (т.е. мощностей бесконечных множеств). Можно сказать, что всякое бесконечное множество не менее чем счетно.

Без доказательства приведем две важные теоремы.

Теорема 1.19 (*теорема Кантора — Бернштейна*). Для любых двух множеств A и B имеет место в точности одно из следующих трех условий: либо $|A| < |B|$, либо $|B| < |A|$, либо $|B| = |A|$. #

Таким образом, любые два множества сравнимы по мощности. Другими словами, „шкала мощностей“ *линейно упорядочена*.

Теорема 1.20. Для любого множества A верно неравенство $|2^A| > |A|$. #

В силу теоремы 1.20 нет *наибольшей* мощности, так как для любого множества A существует множество большей мощности — его булеан. Заметим, что для счетного множества A теорема 1.20 сводится к теореме Кантора 1.15.

Теорема 1.21 (*теорема о квадрате*). Для любого бесконечного множества M его *декартов квадрат* $M \times M$ равномощен самому множеству M.

◀ Доказательство проведем для частных случаев счетного и континуального множеств.

Сначала обратимся к счетному множеству. Для доказательства утверждения достаточно показать, что $\mathbb{N} \times \mathbb{N} \sim \mathbb{N}$, т.е. задать на $\mathbb{N} \times \mathbb{N}$ некоторую нумерацию. Рассмотрим множество $A_i = \{(i, j): j \in \mathbb{N}\}$ упорядоченных пар. Это множество счетно по построению. Легко видеть, что справедливо равенство

$$\mathbb{N} \times \mathbb{N} = \bigcup_{i \in \mathbb{N}} A_i,$$

откуда, согласно теореме 1.12, вытекает счетность декартова квадрата $\mathbb{N} \times \mathbb{N}$ множества $\mathbb{N}$ как счетного объединения счетных множеств.

Перейдем к континуальному множеству. Докажем, что множество всех упорядоченных пар двоичных последовательностей эквивалентно множеству всех таких последовательностей, т.е. $2^{\mathbb{N}} \times 2^{\mathbb{N}} \sim 2^{\mathbb{N}}$.

Паре последовательностей (α, β) поставим в соответствие последовательность $\alpha_0, \beta_0, \alpha_1, \beta_1, \ldots, \alpha_n, \beta_n, \ldots$ Это соответствие будет взаимно однозначным, т.е. установлена биекция между $2^{\mathbb{N}} \times 2^{\mathbb{N}}$ и $2^{\mathbb{N}}$. ▶

Получается, что — как это ни удивительно — в квадрате „столько же“ точек, сколько и в каждой его стороне. Можно обобщить это утверждение для произвольной конечной *декартовой степени* множества M.

Следствие 1.3. Множество рациональных чисел $\mathbb{Q}$ счетно.

◀ Каждому рациональному числу, представленному несократимой дробью $\frac{a}{b}$, однозначно соответствует упорядоченная пара (a, b), и, напротив, любая упорядоченная пара (a, b) взаимно простых целых чисел a и b однозначно определяет несократимую дробь $\frac{a}{b}$ и, значит, рациональное число. Следовательно, множество $\mathbb{Q}$ эквивалентно некоторому бесконечному подмножеству множества $\mathbb{Z} \times \mathbb{Z}$. Поскольку множество $\mathbb{Z} \times \mathbb{Z}$ счетно, из теоремы 1.11 вытекает, что любое его бесконечное подмножество счетно. Таким образом, множество $\mathbb{Q}$ счетно. ▶

В заключение приведем сводку результатов по мощностям некоторых конечных множеств.

Теорема 1.22. Если M и N — конечные множества и $|M| = m$, а $|N| = n$, то:

1) мощность множества всех отображений M в N равна n^m;

2) мощность множества всех биекций из N на себя равна $P_n = n!$;

3) мощность множества всех инъекций из M в N $(m \leqslant n)$ равна $A_n^m = \frac{n!}{(n-m)!}$;

4) мощность множества всех подмножеств множества N равна 2^n;

5) мощность множества всех k-элементных подмножеств множества N равна $C_n^k = \frac{n!}{k!(n-k)!}$;

6) мощность прямого произведения $M \times N$ равна mn. #

Напомним [I], что в комбинаторике число P_n называют числом *перестановок* n элементов, число A_n^m — числом *размещений без повторений* из n элементов по m, число C_n^k (обозначаемое также $\binom{n}{k}$) — числом *сочетаний* из n элементов по k. Эти числа называются также *биномиальными коэффициентами*, поскольку $(a+b)^n = \sum\limits_{k=0}^{n} C_n^k a^{n-k} b^k$ (формула *бинома Ньютона*).

Дополнение 1.1. Об одном парадоксе теории множеств

Задавая с помощью *коллективизирующих свойств* множества, следует иметь в виду, что не каждое *высказывание* определяет коллективизирующее свойство. Попробуем определить множество $Y = \{X\colon X \notin X\}$ — множество всех множеств, не являющихся элементами самих себя. Это множество не пусто. Те „нормальные“ множества, с которыми мы привыкли иметь дело, например числовые, как раз не являются элемента-

ми самих себя: множество $\mathbb{R}$ всех действительных чисел не есть действительное число! Однако попытка определить множество всех множеств, которые не являются элементами самих себя, приводит к противоречию. В самом деле, пусть Y не является элементом самого себя, т.е. $Y \notin Y$. Тогда, поскольку Y есть множество всех множеств, не являющихся элементами самих себя, $Y \in Y$. В то же время, если $Y \in Y$, оно должно обладать свойством, которое указано в определении Y как коллективизирующее, т.е. должно выполняться $Y \notin Y$. Следовательно, мы доказали, что $Y \notin Y \Leftrightarrow Y \in Y$! Это противоречие показывает, что высказывание о множествах $X \notin X$ не задает коллективизирующее свойство.

Указанный парадокс, называемый ***парадоксом Рассела***, приводится иногда в такой „сказочно-шутливой" редакции: „В некоторой деревне живет брадобрей, который по долгу службы должен брить тех и только тех, кто не бреет себя сам". Брадобрей оказывается в незавидном положении: если он не будет себя брить, то тотчас окажется, что он должен себя брить, а следуя неумолимой инструкции, он немедленно должен прекратить бриться, ибо он будет брить себя сам, что запрещено.

Парадокс Рассела показывает, что интуитивное понимание множества и коллективизирующего свойства позволяет трактовать идею множества настолько широко и расплывчато, что может привести к противоречиям.

Замечание 1.8. Не следует путать высказывание, определяющее пустое множество (например, „x есть четное число, не делящееся на два"), и высказывание, не задающее коллективизирующего свойства. Первое коллективизирует, определяя пустое множество, а второе приводит к противоречию, не определяя никакого множества, в том числе и пустого. #

Обсуждение возможных путей выхода из противоречий, подобных парадоксу Рассела, не является предметом данного

учебника*. Мы же только заметим, что ввиду парадокса Рассела мы не можем мыслить конструкции, подобные множеству всех множеств, которые не являются элементами самих себя, в законченном виде, т.е. считать, что нам сразу, одновременно представлены в наличии все мыслимые множества указанного вида. Вместо этого следует представлять себе процесс (обратим внимание на это слово!) порождения новых множеств (назовем их допустимыми), исходя из определенного набора „исходных" допустимых множеств. К ним, в частности, можно отнести известные числовые множества, все конечные множества. Важно понимать также, что указанный выше процесс никак не влияет на „объем" уже имеющихся допустимых множеств: все они жестко зафиксированы и „состав" их элементов никак не меняется. Всякое уже имеющееся допустимое множество всегда „равно самому себе". Но совокупность всех допустимых множеств меняется при порождении новых допустимых множеств из уже имеющихся, и именно поэтому она не может считаться множеством, ибо состав ее элементов не зафиксирован.

Считая, что исходные допустимые множества как-то заданы, мы должны регламентировать операции, которые позволяют из уже имеющихся допустимых множеств строить новые допустимые множества.

Такими операциями являются рассмотренные в главе 1 операции над множествами, в частности образование *неупорядоченной* и *упорядоченной пары*, *булеана* и *фактор-множества*.

Дополнение 1.2. Метод характеристических функций

Доказательство сложных теоретико-множественных тождеств методом двух включений часто бывает довольно громоздким, и при построении доказательства ход рассуждений

*См.: *Архангельский А.А.*; *Шенфилд Дж.*; *Куратовский К.*, *Мостовский А.*; *Кон П.*

не всегда очевиден. Одним из методов, не требующих „угадывания“ пути доказательства, является ***метод характеристических функций***.

Характеристическая функция χ_A ***множества*** $A \subseteq U$ есть функция, отображающая *универсальное множество* U в двухэлементное множество $\{0, 1\}$:

$$\chi_A(x) = \begin{cases} 1, & x \in A; \\ 0, & x \notin A. \end{cases}$$

Из определения характеристической функции множества A вытекает справедливость тождества

$$\chi_A^2(x) = \chi_A(x). \tag{1.10}$$

Выразим характеристическую функцию *пересечения* множеств A и B через характеристические функции $\chi_A(x)$ и $\chi_B(x)$ этих множеств. Из определения пересечения следует, что искомая характеристическая функция должна принимать значение 1 для тех элементов x, которые принадлежат множествам A и B одновременно, и значение 0 в противном случае. Легко видеть, что функция

$$\chi_{A \cap B}(x) = \chi_A(x)\chi_B(x)$$

удовлетворяет этому требованию.

Можно предположить, что характеристическая функция *объединения* множеств A и B будет равна сумме характеристических функций множеств. Однако так ее определить нельзя, поскольку для элементов $x \in A \cap B$ такая сумма будет иметь значение 2. Введем „поправку“ и в результате получим искомую формулу:

$$\chi_{A \cup B}(x) = \chi_A(x) + \chi_B(x) - \chi_A(x)\chi_B(x).$$

Непосредственно из определения $\overline{A}$ — *дополнения* множества A — следует, что

$$\chi_{\overline{A}}(x) = 1 - \chi_A(x).$$

Для *разности* $A \setminus B$ характеристическая функция имеет вид

$$\chi_{A\setminus B}(x) = \chi_A(x) - \chi_A(x)\chi_B(x),$$

а для симметрической разности $A \triangle B$ —

$$\chi_{A\triangle B}(x) = \chi_A(x) + \chi_B(x) - 2\chi_A(x)\chi_B(x).$$

Отметим, что последнюю формулу можно получить, опираясь на свойство 19 (см. с. 35) и тождество (1.10), а также на характеристические функции для пересечения, объединения и разности:

$$\begin{aligned}
\chi_{A\triangle B}(x) = \chi_{(A\cup B)\setminus(A\cap B)}(x) = \\
= \chi_{A\cup B}(x) - \chi_{A\cup B}(x)\chi_{A\cap B}(x) = \\
= \chi_A(x) + \chi_B(x) - \chi_A(x)\chi_B(x) - \\
- (\chi_A(x) + \chi_B(x) - \chi_A(x)\chi_B(x))\chi_A(x)\chi_B(x) = \\
= \chi_A(x) + \chi_B(x) - \chi_A(x)\chi_B(x) - \\
- (\chi_A(x)\chi_B(x) + \chi_A(x)\chi_B(x) - \chi_A(x)\chi_B(x)) = \\
= \chi_A(x) + \chi_B(x) - 2\chi_A(x)\chi_B(x).
\end{aligned}$$

С учетом равенства (1.10) полученную формулу можно записать в виде

$$\chi_{A\triangle B}(x) = (\chi_A(x) - \chi_B(x))^2.$$

Метод характеристических функций доказательства справедливости теоретико-множественного тождества заключается в выражении характеристических функций обеих его частей через характеристические функции входящих в него множеств. Тождество верно тогда и только тогда, когда характеристические функции левой и правой частей совпадают.

Пример 1.22. Используя метод характеристических функций, выясним, справедливо ли тождество

$$(A \triangle B) \cap C = (A \cap C) \triangle (B \cap C).$$

С одной стороны,

$$\chi_{(A\triangle B)\cap C}(x) = \chi_{A\triangle B}\chi_C(x) =$$
$$= (\chi_A(x) + \chi_B(x) - 2\chi_A(x)\chi_B(x))\,\chi_C(x) =$$
$$= \chi_A(x)\chi_C(x) + \chi_B(x)\chi_C(x) - 2\chi_A(x)\chi_B(x)\chi_C(x).$$

С другой стороны,

$$\chi_{(A\cap C)\triangle(B\cap C)}(x) =$$
$$= \chi_{A\cap C}(x) + \chi_{B\cap C}(x) - 2\chi_{A\cap C}(x)\chi_{B\cap C}(x) =$$
$$= \chi_A(x)\chi_C(x) + \chi_B(x)\chi_C(x) - 2\chi_A(x)\chi_C(x)\chi_B(x)\chi_C(x) =$$
$$= \chi_A(x)\chi_C(x) + \chi_B(x)\chi_C(x) - 2\chi_A(x)\chi_B(x)\chi_C(x).$$

Характеристические функции левой и правой частей тождества совпадают. Следовательно, тождество верно.

Пример 1.23. Выясним, является ли тождеством следующее выражение:

$$A \setminus (B \setminus C) = (A \setminus B) \setminus C.$$

С одной стороны,

$$\chi_{A\setminus(B\setminus C)}(x) = \chi_A(x) - \chi_A(x)\chi_{B\setminus C}(x) =$$
$$= \chi_A(x) - \chi_A(x)(\chi_B(x) - \chi_B(x)\chi_C(x)) =$$
$$= \chi_A(x) - \chi_A(x)\chi_B(x) + \chi_A(x)\chi_B(x)\chi_C(x).$$

С другой стороны,

$$\chi_{(A\setminus B)\setminus C}(x) = \chi_{A\setminus B}(x) - \chi_{A\setminus B}(x)\chi_C(x) =$$
$$= \chi_A(x) - \chi_A(x)\chi_B(x) - (\chi_A(x) - \chi_A(x)\chi_B(x))\chi_C(x) =$$
$$= \chi_A(x) - \chi_A(x)\chi_B(x) - \chi_A(x)\chi_C(x) + \chi_A(x)\chi_B(x)\chi_C(x).$$

Легко видеть, что получены разные характеристические функции. Например, при $x \in A$, $x \notin B$ и $x \in C$ $\chi_{A\setminus(B\setminus C)}(x) = 1$, а $\chi_{(A\setminus B)\setminus C}(x) = 0$. Таким образом, доказано, что $A \setminus (B \setminus C) \neq (A \setminus B) \setminus C$. #

Отметим, что метод характеристических функций не является универсальным. Так, его нельзя использовать при доказательстве тождеств, содержащих декартово произведение множеств, в частности, тождеств для соответствий (бинарных отношений).

Вопросы и задачи

1.1. Найти $\bigcap\limits_{n=1}^{\infty} X_n$, если: а) $X_n = \left[-\frac{1}{n}, \frac{1}{n}\right]$; б) $X_n = \left[0, \frac{1}{n}\right]$; в) $X_n = \left(0, \frac{1}{n}\right)$.

1.2. Используя методы двух включений и характеристических функций, доказать свойства 1–18 (см. с. 35).

1.3. Доказать тождества:

а) $A \times (B \cap C) = (A \times B) \cap (A \times C)$;

б) $(A \cap B) \times (C \cap D) = (A \times C) \cap (B \times D)$.

Проиллюстрировать графически, приняв в качестве множеств A, B, C и D отрезки числовой прямой.

1.4. Доказать, что если $(A \subseteq X)$ и $(B \subseteq Y)$, то $(A \times B) \subseteq (X \times Y)$.

1.5. Показать, что $\overline{(A \times B)} \neq \overline{A} \times \overline{B}$. Вывести соответствующее тождество.

1.6. Используя ранее доказанные тождества, показать справедливость тождества $(A \setminus B) \times C = (A \times C) \setminus (B \times C)$.

1.7. Доказать, что для любой функции f и любых множеств A и B имеют место соотношения:

а) $f(A \cup B) = f(A) \cup f(B)$; б) $f(A \cap B) \subseteq f(A) \cap f(B)$;

в) $f(A) \setminus f(B) \subseteq f(A \setminus B)$.

При каких условиях включения б) и в) превращаются в равенство?

1.8. Доказать, что для любой функции f и любых множеств A и B имеет место равенство:

а) $f^{-1}(A \cap B) = f^{-1}(A) \cap f^{-1}(B)$;

б) $f^{-1}(A \cup B) = f^{-1}(A) \cup f^{-1}(B)$;

в) $f^{-1}(\overline{A}) = \overline{f^{-1}(A)}$;

г) $f^{-1}(A \setminus B) = f^{-1}(A) \setminus f^{-1}(B)$.

1.9. Построить графики и графы следующих бинарных отношений, заданных на множестве $X = \{1, 2, 3, 4, 5, 6\}$:

а) $x_1 \varphi x_2$, если $x_1 < x_2 + 1$;

б) $x_1 \tau x_2$, если $x_1 \leqslant x_2$;

в) $x_1 \rho x_2$, если $|x_1 - x_2| \geqslant 3$;

г) $\{(a, b)\colon a + b$ — четное$\}$.

1.10. Определить, какими свойствами (рефлексивность, иррефлексивность, симметричность, антисимметричность, транзитивность) обладают следующие бинарные отношения:

а) $\varphi = \{(a, a), (a, b), (c, a), (b, d), (a, d), (b, c)\}$ на множестве $M = \{a, b, c, d\}$;

б) ρ_n, такое, что $m \rho_n k$, если $m - k$ делится на n, где m, $k \in \mathbb{Z}$, а $n \in \mathbb{Z}$ и фиксировано;

в) φ, такое, что $x \varphi y$, если $x - y \leqslant 2$, $x \in \mathbb{R}$, $y \in \mathbb{R}$.

1.11. Пусть $\rho = \{(x, y)\colon x < y$ и $y + x < 1{,}5\}$ — бинарное отношение на множестве $X = [0, 1]$. Построить графики отношений ρ и ρ^2. Исследовать свойства отношений ρ и ρ^2.

1.12. Найти $D(\rho)$, $R(\rho)$, ρ^{-1}, $\rho \circ \rho$, $\rho^{-1} \circ \rho$, $\rho \circ \rho^{-1}$ для бинарных отношений:

а) $\rho = \{(x, y)\colon x, y \in [0, 1], x + y \leqslant 1\}$;

б) $\rho = \{(x, y)\colon x, y \in [0, 1], 2x \geqslant 3y\}$.

1.13. Доказать, что для любого бинарного отношения $\rho \subseteq \subseteq A \times A$ имеют место равенства:

а) $D(\rho^{-1}) = R(\rho)$;

б) $R(\rho^{-1}) = D(\rho)$;

в) $D(\rho_1 \circ \rho_2) = \rho_1^{-1}(R(\rho_1) \cap D(\rho_2))$;

г) $R(\rho_1 \circ \rho_2) = \rho_2(R(\rho_1) \cap D(\rho_2))$.

1.14. Доказать, что для любых бинарных отношений ρ_1, ρ_2, $\rho_3 \in A \times A$ имеют место равенства:

а) $\rho_1 \cap \rho_1 = \rho_1 \cup \rho_1 = \rho_1$;

б) $\rho_1 \circ (\rho_2 \circ \rho_3) = (\rho_1 \circ \rho_2) \circ \rho_3$;

в) $\rho_1 \circ \mathrm{id}_A = \mathrm{id}_A \circ \rho_1 = \rho_1$;

г) $(\rho_1 \cap \rho_2)^{-1} = \rho_1^{-1} \cap \rho_2^{-1}$;

д) $(\rho_1 \cup \rho_2)^{-1} = \rho_1^{-1} \cup \rho_2^{-1}$;

е) $(\overline{\rho})^{-1} = \overline{(\rho^{-1})}$;

ж) $(\rho_1 \circ \rho_2)^{-1} = \rho_2^{-1} \circ \rho_1^{-1}$.

1.15. Доказать, что для бинарных отношений τ и ρ_i, $i \in I$, справедливы равенства:

а) $\tau \circ \left(\bigcup\limits_{i \in I} \rho_i\right) = \bigcup\limits_{i \in I} (\tau \circ \rho_i)$;

б) $\tau \circ \left(\bigcap\limits_{i \in I} \rho_i\right) \subseteq \bigcap\limits_{i \in I} (\tau \circ \rho_i)$.

1.16. Доказать, что для бинарного отношения ρ на A имеет место равенство $\rho \circ \rho^{-1} = \rho^{-1} \circ \rho = \mathrm{id}_A$, если и только если ρ — биекция A на A.

1.17. Найти необходимые и достаточные условия односторонней обратимости бинарного отношения ρ на множестве A, т.е. условия того, что $\rho \circ \rho^{-1} = \mathrm{id}_A$ (или $\rho^{-1} \circ \rho = \mathrm{id}_A$).

1.18. Построить бинарное отношение, которое является:

а) рефлексивным, симметричным, но нетранзитивным;

б) рефлексивным, антисимметричным, но нетранзитивным;

в) рефлексивным, транзитивным, но несимметричным;

г) антисимметричным, транзитивным, но нерефлексивным.

1.19. Доказать, что для любых рефлексивных отношений ρ и σ на произвольном множестве A $\rho \cup \sigma \subseteq \rho \circ \sigma$.

1.20. Пусть A и B — конечные множества, содержащие m и n элементов. Сколько существует различных соответствий из A в B? Сколько можно задать функций из A в B, а среди последних — инъекций? При каких m и n существуют биекции и сколько их?

1.21. Пусть в $\mathbb{R}^3$ задана плоскость $ax + by + cz = 0$. Точки с радиус-векторами $\boldsymbol{r}_1$ и $\boldsymbol{r}_2$ связаны отношением τ, если

$((\boldsymbol{r}_1 - \boldsymbol{r}_2), \boldsymbol{n}) = 0$, где $\boldsymbol{n}$ — нормаль к плоскости, а $(\cdot, \cdot)$ — скалярное произведение векторов. Показать, что τ — отношение эквивалентности. На какие классы эквивалентности разбивается $\mathbb{R}^3$? Указать фактор-множество множества $\mathbb{R}^3$ по данному отношению эквивалентности.

1.22. Пусть A — конечное множество. Какое отношение эквивалентности на нем дает наибольшее число классов эквивалентности? Сколько классов эквивалентности при этом будет? Сколькими способами можно задать отношение эквивалентности, разбивающее A на два класса?

1.23. Доказать, что число различных отношений эквивалентности на n-элементном множестве удовлетворяет формуле

$$p_{n+1} = \sum_{i=0}^{n} C_n^i p_i, \quad p_0 = 1,$$

где p_i — число различных отношений эквивалентности на i-элементном множестве.

1.24. Доказать, что композиция $\rho_1 \circ \rho_2$ двух эквивалентностей ρ_1 и ρ_2 является эквивалентностью тогда и только тогда, когда $\rho_1 \circ \rho_2 = \rho_2 \circ \rho_1$.

1.25. Описать наименьшее по включению отношение эквивалентности, содержащее данные эквивалентности ρ и σ на A. Каким будет это отношение, если $\rho \circ \sigma = \sigma \circ \rho$?

1.26. Пусть бинарное отношение υ определено на множестве положительных рациональных чисел следующим образом: $(a/b)\, \upsilon\, (c/d)$, если $ad \leqslant bc$. Показать, что υ является отношением линейного порядка.

1.27. Пусть A — произвольное множество и ρ, σ — бинарные отношения на множестве $2^A \times 2^A$: $(P, Q)\, \sigma\, (X, Y)$, если $P \subseteq X$ и $Q \subseteq Y$, а $(P, Q)\, \rho\, (X, Y)$, если $(P \triangle Q) \subseteq (X \triangle Y)$. Являются ли ρ и σ отношениями порядка?

1.28. Пусть M — множество квадратных матриц типа 2×2, элементами которых являются целые числа. Выяснить, является ли бинарное отношение τ, заданное на множестве M, отношением порядка или отношением линейного порядка:

а) $A \tau B$, если $a_{ij} \leqslant b_{ij}$, $i, j = 1, 2$, $A, B \in M$;

б) $A \tau B$, если $a_{ij} \leqslant b_{ij}$, $i, j = 1, 2$ и хотя бы для одной пары элементов неравенство строгое, $AB \in M$.

1.29. Пусть F — множество функций, непрерывных на отрезке $[a, b]$. Проверить, является ли заданное отношение отношением указанного вида:

а) $f(x) \tau g(x)$, если $\int\limits_a^b f(x)\,dx = \int\limits_a^b g(x)\,dx$; отношение эквивалентности;

б) $f(x) \tau g(x)$, если $\int\limits_a^b f(x)\,dx \leqslant \int\limits_a^b g(x)\,dx$; отношение порядка и отношение предпорядка.

1.30. Пусть ρ_1 и ρ_2 — линейные порядки на множестве A. Когда $\rho_1 \circ \rho_2$ — линейный порядок?

Указание: докажите, что если $\rho_1 \neq \rho_2$, то $\rho_1 \circ \rho_2$ не является линейным порядком.

1.31. Всегда ли транзитивна композиция транзитивных бинарных отношений?

Указание: постройте пример конечного упорядоченного множества, в котором композиция исходного и двойственного порядка не является транзитивным отношением.

1.32. Пусть A и B — конечные множества. Используя метод математической индукции, доказать, что $|A \times B| = |A||B|$.

Пусть $A_1, \ldots, A_n$ — конечные множества. Доказать, что $|A_1 \times \ldots \times A_n| = |A_1| \ldots |A_n|$.

1.33. Какую мощность имеет множество простых чисел?

1.34. Пусть A — множество всех многочленов степени не выше n, имеющих коэффициенты заданного вида. Определить мощность этого множества, если: а) все коэффициенты многочленов рациональные; б) свободный член действительный, а все остальные коэффициенты рациональные.

1.35. Доказать счетность следующих множеств:

а) множества всех многочленов с рациональными коэффициентами;

б) множества всех попарно непересекающихся открытых шаров в $\mathbb{R}^n$;

в) множество всех цифр 8, расположенных на плоскости так, что ни одна пара цифр не имеет общих точек, кроме, может быть, точек касания.

1.36. Определить мощность множество всех точек плоскости, у которых:

а) обе координаты рациональные;

б) первая координата рациональная, а вторая — иррациональная.

1.37. Доказать, что следующие множества имеют мощность континуума:

а) множество $\mathbb{N}^\omega$ всех бесконечных последовательностей натуральных чисел;

б) множество $\mathbb{N}^\infty$ всех последовательностей натуральных чисел;

в) множество A^∞ всех (конечных или бесконечных) последовательностей элементов из конечного множества A.

1.38. Доказать, не используя теорему 1.20, что множество $2^{[0,1]}$ имеет мощность бóльшую, чем мощность континуума.

Указание: установите биекцию множества $2^{[0,1]}$ на множество всех функций из $[0,1]$ в $(0,1)$ (характеристических функций подмножеств отрезка $[0,1]$), а затем обобщите „диагональную“ конструкцию из доказательства теоремы 1.15.

2. АЛГЕБРЫ: ГРУППЫ И КОЛЬЦА

Предметом рассмотрения в абстрактной алгебре являются произвольные множества с заданными на них операциями. При этом природа множеств и операций может существенно отличаться от привычных числовых множеств и известных операций над числами. Мы уже сталкивались с операциями над множествами и *бинарными отношениями* (см. 1), которые оказались в чем-то похожими на операции над числами, но в то же время проявились и их существенные отличия.

В дискретной математике разрабатываются алгоритмы и вычислительные методы, позволяющие манипулировать сложно организованными нечисловыми структурами. Проблема работы с такими объектами возникла в связи с развитием современных информационных технологий и переходом от собственно вычислений (т.е. операций над числами) к обработке сложных структур данных. Так, проблемы программирования и машинного перевода привели к задачам работы с языковыми структурами, проблемы автоматизации проектирования — к задачам обработки графических объектов.

Современная дискретная математика проникнута алгебраическим духом, поскольку оказалось, что именно на алгебраической базе наиболее удобно разрабатывать общие подходы к работе с объектами различной природы.

2.1. Операции. Понятие алгебраической структуры

Множество векторов в пространстве с операцией сложения векторов, операцией векторного умножения, множество квадратных матриц с операциями сложения или умножения, множество функций с операцией сложения — вот примеры некото-

рых множеств с операциями, рассматривающихся в различных разделах математики. Выясним, что общего есть в свойствах операций на этих множествах и в чем их различие.

Определение 2.1. Пусть A — произвольное непустое множество и n — натуральное число. Любое *отображение*

$$\omega\colon A^n \to A$$

называют ***n-арной*** (или ***n-местной***) ***операцией*** на множестве A.

Таким образом, согласно приведенному определению, n-арная операция ω каждому *кортежу* $(a_1, \ldots, a_n) \in A^n$ однозначно сопоставляет элемент $b \in A$. *Компоненты кортежа* называют при этом ***аргументами операции*** ω, а b — ***результатом применения операции*** ω к аргументам $a_1, \ldots, a_n$.

Для n-арной операции используют обозначение

$$b = \omega(a_1, \ldots, a_n)$$

или

$$b = a_1 \ldots a_n \omega.$$

Обычно, если $n = 2$, пишут $a_1 \omega a_2$.

При $n = 1$ и $n = 2$ говорят соответственно об ***унарной операции*** и ***бинарной операции***.

Специально вводят понятие ***нульарной операции*** (т.е. для $n = 0$). Под нульарной операцией на множестве A понимают произвольный фиксированный элемент множества A. Нульарные операции позволяют фиксировать элементы множества A, обладающие некоторыми специальными свойствами. Примером выполнения нульарной операции является, например, фиксирование нуля в множестве целых чисел с операцией сложения. Примером унарной операции служит *дополнение* заданного *множества* до *универсального множества*.

Наиболее важными в алгебре и, следовательно, наиболее исследованными являются бинарные операции. Примерами

таких операций могут служить сложение и умножение чисел, сложение и умножение матриц, сложение векторов линейного пространства.

Рассмотрим бинарную операцию на множестве A, обозначив ее звездочкой $(*)$.

Эту операцию называют:

1) ***ассоциативной***, если $(x*y)*z = x*(y*z)$ для любых x, y, $z \in A$;

2) ***коммутативной***, если $x*y = y*x$ для любых x, $y \in A$;

3) ***идемпотентной***, если $x*x = x$ для любого $x \in A$.

Ассоциативность операции $*$ позволяет для любых элементов a_1, a_2, ..., $a_n \in A$ однозначно трактовать результат выражения $a_1 * a_2 * \ldots * a_n$, так как

$$a_1 * a_2 * \ldots * a_n = a_1 * (a_2 * \ldots * a_n) = \\ = (a_1 * a_2) * (a_3 * \ldots * a_n) = (a_1 * a_2 * \ldots * a_{n-1}) * a_n.$$

Операция сложения, заданная на множестве натуральных чисел, является ассоциативной и коммутативной. Операция умножения матриц ассоциативна, но не коммутативна. Идемпотентными являются операции объединения и пересечения множеств.

Элемент $\mathbf{0}$ множества A называют ***левым*** (***правым***) ***нулем*** относительно данной операции $*$, если $\mathbf{0}*x = \mathbf{0}$ $(x*\mathbf{0} = \mathbf{0})$ для любого $x \in A$.

Если $\mathbf{0}'$ — левый нуль, а $\mathbf{0}''$ — правый нуль, то они совпадают. Действительно, если $\mathbf{0}'$ и $\mathbf{0}''$ существуют, то они совпадают, так как $\mathbf{0}' = \mathbf{0}' * \mathbf{0}'' = \mathbf{0}''$, и в этом случае говорят просто о ***нуле относительно операции.*** Из приведенных равенств следует, что нуль единственный и для него одновременно выполнены оба равенства $\mathbf{0}*x = \mathbf{0}$ и $x*\mathbf{0} = \mathbf{0}$.

Пример 2.1. а. На множестве целых чисел нулем относительно операции умножения будет число 0.

б. На множестве квадратных матриц вида $\begin{pmatrix} a & 0 \\ b & 1 \end{pmatrix}$, где элементы a и b — действительные числа, любая матрица вида $\begin{pmatrix} 0 & 0 \\ d & 1 \end{pmatrix}$ будет правым нулем относительно операции умножения, поскольку

$$\begin{pmatrix} a & 0 \\ b & 1 \end{pmatrix} \begin{pmatrix} 0 & 0 \\ d & 1 \end{pmatrix} = \begin{pmatrix} 0 & 0 \\ d & 1 \end{pmatrix}.$$

Однако левого нуля в этом множестве нет, так как иначе он совпадал бы с правым нулем и был бы единственным. Но правых нулей имеется больше одного. #

Элемент $\mathbf{1}$ множества A называют ***левым*** (***правым***) ***нейтральным элементом*** относительно операции $*$, если $\mathbf{1} * x = x$ ($x * \mathbf{1} = x$) для любого элемента $x \in A$. Для левого $\mathbf{1}'$ и правого $\mathbf{1}''$ нейтральных элементов, если они оба существуют, выполнены равенства $\mathbf{1}' = \mathbf{1}' * \mathbf{1}'' = \mathbf{1}''$, согласно которым они совпадают. В этом случае элемент $\mathbf{1}$, который является и левым нейтральным, и правым нейтральным, единственный, и его называют просто ***нейтральным элементом.***

Пример 2.2. Нейтральным элементом относительно операции умножения на множестве натуральных чисел является число 1. На множестве целых чисел нейтральным элементом относительно операции сложения будет число 0.

На множестве квадратных матриц вида $\begin{pmatrix} a & 0 \\ b & 0 \end{pmatrix}$, где элементы a и b — действительные числа, любая матрица вида $\begin{pmatrix} 1 & 0 \\ d & 0 \end{pmatrix}$ будет правым нейтральным элементом по операции умножения, ибо

$$\begin{pmatrix} a & 0 \\ b & 0 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ d & 0 \end{pmatrix} = \begin{pmatrix} a & 0 \\ b & 0 \end{pmatrix}.$$

Поскольку правых нейтральных элементов несколько, то левого нейтрального элемента по этой операции нет, так как иначе

существовал бы единственный нейтральный элемент (левый и правый). #

Следует заметить, что не для всякой бинарной операции нули и нейтральные элементы (левые и правые, в частности), существуют.

Рассмотрим некоторые примеры бинарных операций на множествах.

Пример 2.3. **а.** Пусть U — *универсальное множество*. Теоретико-множественные операции $\cup$, $\cap$ на множестве 2^U являются идемпотентными, ассоциативными и коммутативными, причем *пустое множество* является нулем относительно *пересечения* и нейтральным элементом относительно *объединения* ($\varnothing \cap A = A \cap \varnothing = \varnothing$, $\varnothing \cup A = A \cup \varnothing = A$), тогда как универсальное множество есть нуль относительно объединения и нейтральный элемент относительно пересечения ($U \cap A = A \cap U = A$, $U \cup A = A \cup U = U$).

Операция $\setminus$ (*разность множеств*) не является ассоциативной, так как $A \setminus (B \setminus C) \neq (A \setminus B) \setminus C$.

б. На множестве всех *бинарных отношений* на множестве A операция *композиции отношений* является ассоциативной, но не коммутативной, а *диагональ* множества A будет нейтральным элементом относительно этой операции (см. **1.4**).

в. Пусть X — произвольное множество, содержащее не менее двух элементов. На множестве всех отображений из X в X с операцией композиции отображений постоянное отображение φ_a, переводящее любой элемент $x \in X$ в фиксированный элемент $a \in X$, будет правым нулем, но не будет нулем относительно композиции отображений.

Действительно, для любого отображения $f\colon X \to X$ и любого $x \in X$ имеем $f \circ \varphi_a(x) = \varphi_a(f(x)) = a = \varphi_a(x)$, т.е. $f \circ \varphi_a = \varphi_a$, что и означает, что φ_a — правый нуль относительно операции композиции на множестве отображений из X в X. Однако для любого $x \in X$ $\varphi_a \circ f(x) = f(\varphi_a(x)) = f(a)$, т.е. $\varphi_a \circ f = \varphi_{f(a)}$ — отображение, которое любой элемент X переводит в элемент

$f(a)$. Поскольку $f(a)$ в общем случае не равно a, то $\varphi_a \circ f \neq \varphi_a$. Значит, φ_a не является левым нулем относительно операции композиции. #

Рассмотренные выше примеры множеств с операциями приводят к следующим определениям.

Определение 2.2. ***Алгебра*** (***универсальная алгебра***, ***Ω-алгебра***) считается заданной, если заданы некоторое множество A, называемое ***носителем*** данной ***алгебры***, и некоторое множество операций Ω на A, называемое ***сигнатурой*** данной ***алгебры***. Алгебру, носитель которой есть конечное множество, называют ***конечной алгеброй***.

Поскольку алгебра задается ее носителем и сигнатурой, мы будем в записи обозначать алгебру как *упорядоченную пару* множеств $\mathcal{A} = (A, \Omega)$, полагая, что первая компонента этой пары есть носитель, а вторая — сигнатура. Подчеркнем, что алгебра — это не просто носитель и не просто сигнатура, а „синтез“ этих двух объектов.

Замечание 2.1. Операции, включенные в сигнатуру, заданы как некоторые специальные отображения. Однако при этом не оговариваются свойства, которыми обладают операции на носителе. Например, они могут быть ассоциативными, коммутативными и т.д. При задании алгебр свойства операций обычно указывают дополнительно.

Если существует нейтральный элемент относительно операции, то его можно задать как нульарную операцию на носителе и включить в сигнатуру, а можно не включать и описать как свойство соответствующей операции.

Таким образом, одну и ту же алгебру можно задавать по-разному. Ниже приведены примеры различных описаний конкретных алгебр.

В ряде случаев указание носителя алгебры предполагает и задание определенной сигнатуры. В этом случае для упрощения

пишут не $\mathcal{A} = (A, \Omega)$, а просто $\mathcal{A} = A$ и говорят „элемент алгебры $\mathcal{A}$“, имея в виду элемент носителя этой алгебры.

Пример 2.4. **а.** Рассмотрим алгебру

$$\mathcal{A}_1 = \left(2^M, \{\cup, \cap, \setminus, \Delta, \bar{}, \varnothing, M\}\right).$$

Ее носителем является множество всех подмножеств произвольно фиксированного множества M, а сигнатура состоит из следующих операций над множествами: объединения, пересечения, разности, *симметрической разности*, *дополнения*, пустого множества и множества M (последние два элемента сигнатуры определяют нульарные операции).

б. Для любого множества M можно определить алгебру

$$\mathcal{A}_2 = \left(2^{M\times M}, \{\cup, \circ, {}^{-1}\}\right),$$

носителем которой является множество всех подмножеств множества упорядоченных пар на M, т.е. множество всех *бинарных отношений на множестве* M, а сигнатура состоит из операций объединения, *композиции бинарных отношений* и взятия *обратного отношения*.

в. На множестве $\mathbb{R}$ действительных чисел можно, например, определить такую алгебру:

$$\mathcal{A}_3 = (\mathbb{R}, \{+, \cdot, 0, 1\}),$$

сигнатура которой состоит из операций сложения, умножения, а также двух нульарных операций, обозначающих два особых числа 0 и 1. Обратим внимание на то, что числа 0 и 1 в данном случае являются соответственно нулем и нейтральным элементом относительно умножения, а число 0 также играет роль нейтрального элемента относительно сложения. Мы применили понятие нульарной операции, чтобы в обозначении алгебры отразить элементы со специальными свойствами.

г. Все предыдущие примеры алгебр были алгебрами с конечной сигнатурой, т.е. с сигнатурой, состоящей из конечного

числа операций. Однако несложно построить пример алгебры с бесконечной сигнатурой. Например, алгебра

$$\mathcal{A}_4 = (\mathbb{R},\ \{\uparrow^n\colon n \geqslant 2\})$$

на множестве действительных чисел $\mathbb{R}$ с операцией $\uparrow^n$ возведения в натуральную степень $n \geqslant 2$ имеет *счетную* сигнатуру. Далее будут приведены примеры алгебр и с несчетными сигнатурами. #

Определяя алгебру, следует помнить, что результат применения любой операции обязательно должен принадлежать тому же множеству, что и ее аргументы. Например, пару из множества V_3 всех свободных векторов в пространстве и операции скалярного умножения векторов нельзя рассматривать как алгебру, так как скалярное произведение двух векторов не есть вектор. Заменив скалярное умножение векторным, получим алгебру.

Кроме того, нельзя забывать, что n-арная операция, как и всякое отображение, должна быть определена для любого кортежа длины n элементов множества. Поэтому не является алгеброй множество всех матриц с операциями сложения и умножения матриц, так как эти операции определены не для любой упорядоченной пары матриц. Если же при тех же операциях ограничиться множеством квадратных матриц фиксированного порядка n, то получим алгебру.

Точно так же множество действительных чисел $\mathbb{R}$ с операцией : деления действительных чисел не является алгеброй, поскольку результат деления не определен при нулевом делителе. Пара же $(\mathbb{R}\setminus\{0\},\ \{:\})$ есть алгебра.

Договоримся, определяя конкретную алгебру, записывать ее сигнатуру без фигурных скобок, перечисляя после обозначения носителя все операции. Так, в примере 2.4.а первая алгебра может быть записана как $\mathcal{A}_1 = (2^M, \cup, \cap, \setminus, \Delta, \overline{\ }, \varnothing, M)$.

Для алгебры $\mathcal{A} = (A,\ \Omega)$ обозначим через $\Omega^{(n)}$ подмножество сигнатуры Ω, состоящее из всех n-арных операций. Тогда $\Omega =$

$= \bigcup\limits_{n \geqslant 0} \Omega^{(n)}$. Так, для алгебры $\mathcal{A}_1$ в примере 2.4.а будем иметь:

$$\Omega^{(0)} = \{\varnothing, M\},$$
$$\Omega^{(1)} = \{\overline{\ }\},$$
$$\Omega^{(2)} = \{\cup, \cap, \setminus, \Delta\},$$
$$\Omega^{(n)} = \varnothing \text{ при } n > 2.$$

Определение 2.3. Две алгебры $\mathcal{A}_1 = (A_1, \Omega_1)$ и $\mathcal{A}_2 = (A_2, \Omega_2)$ называют ***однотипными***, если существует такая *биекция* Ω_1 на Ω_2, при которой n-арная операция из Ω_1 для любого n переходит в n-арную из Ω_2.

Пример 2.5. Алгебра $\left(2^M, \cup, \cap, \varnothing, M\right)$, заданная на множестве всех подмножеств множества M, и алгебра $\mathcal{A}_3 = (\mathbb{R}, +, \cdot, 0, 1)$ (см. пример 2.4.в), заданная на множестве действительных чисел, однотипны. Биекцию (*взаимно однозначное соответствие*) между их сигнатурами, которая сохраняла бы арность операций, можно определить и так: $\cup \mapsto +$, $\cap \mapsto \cdot$, $\varnothing \mapsto 0$, $M \mapsto 1$. Указанный способ задания биекции не единственный. Например, ее можно определить так: $\cup \mapsto \cdot$, $\cap \mapsto +$, $\varnothing \mapsto 1$, $M \mapsto 0$.

Алгебра $\left(2^M, \cup, \cap, \varnothing, M\right)$ и алгебра $\mathcal{A}_1$ в примере 2.4.а не являются однотипными, так как их сигнатуры состоят из разного числа операций и между ними установить взаимно однозначное соответствие нельзя.

Не являются однотипными и алгебры $\left(2^M, \overline{\ }\right)$ и $(\mathbb{N}, +)$, ибо в первой алгебре единственная операция ее сигнатуры является унарной, а во второй — бинарной. #

Нередко сигнатуры однотипных алгебр и элементы этих сигнатур — операции — обозначают одинаково. Так, мы пишем $(\mathbb{R}, +, \cdot, 0, 1)$ и $(\mathbb{Q}, +, \cdot, 0, 1)$, хотя первая алгебра задана на множестве всех действительных чисел, а вторая — на множестве рациональных чисел, и, например, сложение в первой алгебре, строго говоря, не есть та же самая операция, что сложение во второй алгебре. В общем случае мы часто будем

говорить о различных (но однотипных) Ω-алгебрах, заданных на разных носителях, понимая, что Ω есть общее для всех этих алгебр обозначение их сигнатур.

2.2. Группоиды, полугруппы, группы

Рассмотрим *алгебры*, *сигнатуры* которых состоят из одной *бинарной операции*. Эту операцию будем обозначать точкой $(\cdot)$ и условно называть в этом случае умножением.

Группоидом называют любую алгебру $\mathcal{G} = (G, \cdot)$, сигнатура которой состоит из одной бинарной операции. В группоиде на бинарную операцию нет никаких ограничений.

Группоид $(G, \cdot)$ называют ***полугруппой***, если его операция *ассоциативна*, т.е. для любых элементов a, b, c *носителя* G выполняется равенство $a \cdot (b \cdot c) = (a \cdot b) \cdot c$.

Пример 2.6. а. Множество V_3 свободных векторов вместе с операцией векторного умножения является группоидом, но не полугруппой, так как векторное умножение не ассоциативно.

б. Множество натуральных чисел вместе с операцией возведения в степень также будет только группоидом, так как $(a^b)^c \neq a^{(b^c)}$.

в. Множество 2^A всех подмножеств множества A вместе с операцией $\setminus$ (*разность множеств*) тоже только группоид, поскольку указанная операция не ассоциативна.

г. Множество натуральных чисел $\mathbb{N}$ вместе с операцией сложения будет полугруппой. #

Группоид $\mathcal{G} = (G, \cdot)$ называют ***моноидом***, если его операция ассоциативна и относительно операции существует *нейтральный элемент*. Его называют ***нейтральным элементом моноида*** $\mathcal{G}$ или ***единицей моноида*** и обозначают $\mathbf{1}$.

Таким образом, моноид $\mathcal{G} = (G, \cdot)$ есть полугруппа, в которой для любого a имеют место равенства $a \cdot \mathbf{1} = \mathbf{1} \cdot a = a$, где $\mathbf{1}$ — нейтральный элемент (единица) моноида.

Поскольку нейтральный элемент относительно любой бинарной операции является единственным (см. **2.1**), мы можем рассматривать моноид как алгебру $(G, \cdot, \mathbf{1})$, сигнатура которой состоит из двух операций: бинарной операции $\cdot$ (умножение) и *нульарной операции* $\mathbf{1}$ (нейтрального элемента). Введение $\mathbf{1}$ в сигнатуру моноида удобно тем, что зачастую при рассмотрении конкретных примеров моноидов целесообразно явно указать нейтральный элемент относительно его операции. Например, алгебра $\left(2^{A\times A}, \circ, \mathrm{id}_A\right)$ есть моноид всех *бинарных отношений* на множестве A с операцией *композиции бинарных отношений*, в котором нейтральным элементом является *диагональ* множества A.

Среди полугрупп выделяют полугруппы с коммутативной операцией — ***коммутативные полугруппы.***

Пример 2.7. а. Множество всех бинарных отношений на произвольном множестве A с операцией композиции отношений будет моноидом, нейтральным элементом которого служит диагональ множества A, поскольку для любых бинарных отношений ρ, τ и σ на множестве A имеют место равенства (см. **1.4**) $\rho\circ(\tau\circ\sigma) = (\rho\circ\tau)\circ\sigma$ и $\mathrm{id}_A\circ\rho = \rho\circ\mathrm{id}_A = \rho$.

б. Множество всех отображений некоторого множества A в себя по операции композиции отображений есть моноид.

Напомним, что композиция отображений снова есть отображение и операция композиции имеет нейтральный элемент: *тождественное отображение* A на себя. Поскольку любое отображение множества A в себя можно рассматривать как бинарное отношение на этом множестве, а композицию отображений — как частный случай композиции отношений, требуемые свойства операции композиции отображений выполняются (см. пример 2.7.а). При этом тождественному отображению соответствует диагональ id_A множества A. Этот моноид называют часто ***симметрическим моноидом*** или ***симметрической полугруппой*** множества A.

в. Алгебра $(\mathbb{N}_0, +)$, где носитель — множество $\mathbb{N}_0$ неотрицательных целых чисел, а сигнатура состоит из одной операции сложения, есть коммутативный моноид, в котором нейтральный элемент — это число 0. Действительно, сумма двух натуральных чисел есть натуральное число, операция сложения ассоциативна, коммутативна и для любого натурального числа n имеет место равенство $n+0=n$.

Обратим внимание на то, что свойства нейтральных элементов и нулей ассоциируются со свойствами чисел 1 и 0 относительно операций умножения и сложения чисел.

г. Алгебра $(\mathbb{Z}, \cdot)$, у которой носителем является множество целых чисел, а сигнатура состоит из одной операции умножения, есть коммутативный моноид. Нейтральным элементом этого моноида является число 1.

д. Пусть A — конечное множество, а A^n — множество кортежей длины n. На множестве всех кортежей $A^+ = \bigcup\limits_{n\geqslant 1} A^n$ определим операцию ***соединения (конкатенации) кортежей*** следующим образом:

$$(a_1, \ldots, a_m)\cdot(b_1, \ldots, b_k) = (a_1, \ldots, a_m, b_1, \ldots, b_k).$$

Можно видеть, что введенная операция ассоциативна, но не имеет нейтрального элемента. Таким образом, построена полугруппа, но не моноид.

Чтобы превратить эту полугруппу в моноид, расширим носитель полугруппы, введя понятие ***нулевой декартовой степени*** A^0 произвольного множества A. Под A^0 понимают одноэлементное множество $\{\lambda\}$, единственный элемент которого называют ***пустым кортежем***. Такое определение множества A^0 объясняется следующим: мощность положительной декартовой степени A^n конечного множества равна $|A|^n$. При $n=0$ должно быть $|A^0| = |A|^0 = 1$, откуда заключаем, что A^0 — одноэлементное множество.

Обозначив $A^* = A^0 \cup A^+$, по определению для любого $x \in A^*$ полагаем $x\cdot\lambda = \lambda\cdot x = x$. В результате получим алгебру

$(A^*, \cdot)$, являющуюся моноидом, с нейтральным элементом λ. Его называют ***свободным моноидом***, порожденным множеством A. #

Полугруппу, операция которой коммутативна и *идемпотентна*, называют ***полурешеткой***.

Пример 2.8. а. Алгебры $(2^A, \cup)$, $(2^A, \cap)$ (для произвольного фиксированного множества A) являются полурешетками, поскольку операции $\cup$ и $\cap$ ассоциативны, коммутативны и идемпотентны.

б. Алгебра $(\mathbb{N}, \text{НОК})$, где НОК — операция вычисления наименьшего общего кратного двух чисел, является полурешеткой. Покажем, что указанная операция ассоциативна. Рассмотрим произвольные натуральные числа m, n и l. Каждое из этих чисел можно разложить на произведение простых чисел и представить в виде

$$m = p_1^{\alpha_1} \dots p_k^{\alpha_k}, \quad n = p_1^{\beta_1} \dots p_k^{\beta_k}, \quad l = p_1^{\gamma_1} \dots p_k^{\gamma_k},$$

где набор простых чисел p_1, ..., p_k выбран одинаковым для всех трех чисел, а некоторые из показателей α_i, β_i и γ_i, $i = \overline{1,k}$, могут быть равными нулю. Тогда для чисел m и n имеем

$$\text{НОК}(n,m) = p_1^{\max(\alpha_1,\beta_1)} \dots p_k^{\max(\alpha_k,\beta_k)}.$$

Таким образом, ассоциативность операции НОК сводится к ассоциативности операции max вычисления наибольшего из двух натуральных чисел. Ассоциативность последней вытекает из очевидного тождества $\max(a, \max(b,c)) = \max(\max(a,b), c)$, верного для любых чисел a, b и c.

Поскольку $\text{НОК}(n,m) = \text{НОК}(m,n)$, операция НОК коммутативна, а так как для любого натурального числа справедливо равенство $\text{НОК}(n,n) = n$, то операция идемпотентна.

в. Алгебра $(\mathbb{N}, \text{НОД})$, где НОД — операция вычисления наибольшего общего делителя двух целых чисел, также является полурешеткой. #

Группоид $\mathcal{G} = (G, \cdot)$ называют ***группой***, если операция ассоциативна, существует нейтральный элемент (единица) $\mathbf{1}$ относительно умножения и для каждого $x \in G$ существует такой элемент $x' \in G$, называемый ***обратным*** к x, что $x \cdot x' = x' \cdot x = \mathbf{1}$.

Таким образом, группа — это алгебра $\mathcal{G} = (G, \cdot)$, в которой для всех a, b, $c \in G$ выполняется равенство $a \cdot (b \cdot c) = (a \cdot b) \cdot c$, существует единственный элемент $\mathbf{1} \in G$, такой, что $a \cdot \mathbf{1} = \mathbf{1} \cdot a = a$ для любого $a \in G$, и для каждого $a \in G$ существует такой элемент a', что $a \cdot a' = a' \cdot a = \mathbf{1}$. Короче говоря, группа — это моноид, в котором для каждого элемента существует обратный элемент.

Отметим, что задать группу как алгебру можно несколькими способами в зависимости от состава операций, включенных в сигнатуру.

Во-первых, в сигнатуру может быть включена единственная бинарная операция. В этом случае пишут $\mathcal{G} = (G, \cdot)$, а все свойства операции описывают дополнительно.

Во-вторых, в сигнатуру может быть включена нульарная операция — нейтральный элемент группы. В этом случае пишут $\mathcal{G} = (G, \cdot, \mathbf{1})$ и дополнительно указывают существование обратного элемента относительно бинарной операции для всех элементов носителя.

Третий способ задания группы как алгебры вытекает из следующей теоремы.

Теорема 2.1. В любой группе $\mathcal{G} = (G, \cdot)$ для каждого $a \in G$ элемент, обратный к a, единственный.

◀ Пусть в группе $(G, \cdot)$ с единицей $\mathbf{1}$ для некоторого a существуют два элемента a' и a'', обратных к a. Тогда $a' = a' \cdot \mathbf{1}$ в силу свойства единицы. Так как $\mathbf{1} = a \cdot a''$, то $a' = a' \cdot (a \cdot a'')$. Используя ассоциативность и учитывая, что a' — элемент, обратный к a, получим

$$a' \cdot (a \cdot a'') = (a' \cdot a) \cdot a'' = \mathbf{1} \cdot a'' = a''. \qquad \blacktriangleright$$

Единственность для каждого элемента a обратного элемента a' группы $\mathcal{G}$ позволяет обозначать его как a^{-1} и операцию $^{-1}\colon a \mapsto a^{-1}$ вычисления (или взятия) обратного элемента ввести в сигнатуру группы. Таким образом, группу можно рассматривать и как алгебру $\mathcal{G} = \bigl(G, \cdot, {}^{-1}, \mathbf{1}\bigr)$, сигнатура которой состоит из бинарной операции умножения, унарной операции взятия обратного элемента и нульарной операции — единицы группы (нейтрального элемента).

В дальнейшем в зависимости от контекста будем использовать все указанные варианты задания группы.

Среди групп также выделяют те, бинарная операция в которых коммутативна, — ***коммутативные*** (***абелевы****) ***группы***. В коммутативных полугруппах и группах бинарную операцию часто обозначают знаком $+$ и называют ***сложением***.

Уместно здесь рассмотреть вопрос о двух формах записи бинарной операции группы. В ***аддитивной записи*** операции ее обозначают знаком $+$, нейтральный элемент — знаком $\mathbf{0}$, а элемент, обратный к a относительно операции $+$, записывают в виде $-a$, называя его при этом ***противоположным*** к a.

В ***мультипликативной записи*** операцию обозначают знаком $\cdot$, нейтральный элемент — знаком $\mathbf{1}$, а элемент, обратный к a, записывают в виде a^{-1}. В этом случае бинарную операцию группы часто называют ***умножением*** (также ***умножением группы*** или ***групповым умножением***), а элемент $a \cdot b$, как правило записываемый в виде ab, — ***произведением*** элементов a и b.

В алгебраической литературе сложилась такая традиция, что аддитивная запись используется преимущественно для коммутативных групп. Поскольку одним из самых простых, распространенных и вместе с тем важных примеров коммутативной группы служит аддитивная группа целых чисел, то обозначения и термины для произвольной аддитивно записываемой коммутативной группы „скопированы“ с терминов для группы

*Н. Абель (1802–1829) — норвежский математик.

$(\mathbb{Z}, +, 0)$. Аналогично мультипликативная запись произвольной группы „позаимствована“ у мультипликативных групп рациональных и вещественных чисел.

Пример 2.9. а. Алгебра $(\mathbb{Z}, +)$ — коммутативная группа, поскольку на множестве целых чисел операция сложения ассоциативна и коммутативна, число 0 есть нейтральный элемент, и для каждого целого числа n существует обратный по сложению элемент, а именно число $-n$, противоположное n. Рассматриваемую группу называют ***аддитивной группой целых чисел.***

б. Множество всех *биекций* некоторого множества A на себя с операцией композиции отображений есть группа.

Это следует из того, что композиция двух биекций есть биекция, операция композиции ассоциативна, ее нейтральный элемент — тождественное отображение id_A — есть биекция, для всякой биекции $f\colon A \to A$ отображение f^{-1}, обратное биекции f, определено, является биекцией и выполнены равенства $f \circ f^{-1} = f^{-1} \circ f = \mathrm{id}_A$.

Эту группу называют ***симметрической группой множества*** A, а в том случае, когда множество A конечно, — ***группой подстановок*** множества A. Если множество A состоит из n элементов, группу подстановок этого множества называют также ***симметрической группой степени*** n или *группой подстановок n-й степени* и обозначают S_n (см. пример 2.10).

в. Алгебры $(\mathbb{Q}\setminus\{0\}, \cdot)$ и $(\mathbb{R}\setminus\{0\}, \cdot)$ есть коммутативные группы. Их называют ***мультипликативной группой рациональных чисел*** и ***мультипликативной группой действительных чисел*** соответственно. В каждой из них число 1 есть нейтральный элемент (единица) группы, а обратный к числу x по операции умножения элемент x^{-1} есть число $x^{-1} = 1/x$.

г. Для произвольно фиксированного множества A рассмотрим алгебру $(2^A, \Delta)$, где Δ — операция вычисления *симметрической разности множеств.* Операция Δ ассоциативна и

коммутативна (см. 1.1). Для любого $X \subseteq A$ имеем $X \triangle \varnothing = X$. Кроме того, $X = Y$ тогда и только тогда, когда $X \triangle Y = \varnothing$. Поэтому алгебра $\left(2^A, \triangle\right)$ является абелевой группой, в которой каждый элемент обратен сам себе, а нейтральный элемент — пустое множество.

д. Рассмотрим алгебру $\mathbb{Z}_k^+ = (\{0, 1, \ldots, k-1\}, \oplus_k)$, в которой операция $\oplus_k$ (***сложения по модулю k***) определяется так: для любых двух m и n число $m \oplus_k n$, называемое ***суммой*** чисел m и n ***по модулю*** k, равно остатку от деления арифметической суммы $m + n$ на k. Можно проверить, что эта алгебра является коммутативной группой. Ее называют ***аддитивной группой вычетов по модулю k***. Нейтральным элементом служит число 0, а обратным к числу n будет $k - n$, поскольку $n \oplus_k (k - n) = 0$.

е. Множество всех невырожденных (т.е. имеющих ненулевой определитель) числовых квадратных матриц порядка n с операцией умножения матриц является группой. Действительно, произведение двух невырожденных матриц снова есть невырожденная матрица [III]; единичная матрица порядка n невырожденная, и матрица, обратная к невырожденной, также является невырожденной. Эту группу будем обозначать $\mathcal{M}_n$. #

Из рассмотренных четырех видов алгебр — группоида, полугруппы, моноида и группы — последняя обладает наиболее интересными свойствами. Изучим более подробно операцию вычисления обратного элемента.

Теорема 2.2. Пусть $\mathcal{G} = (G, \cdot)$ — группа. Для любых элементов $a, b \in G$ верны тождества

$$(a \cdot b)^{-1} = b^{-1} \cdot a^{-1}, \quad (a^{-1})^{-1} = a.$$

◀ В силу ассоциативности умножения группы имеем

$$(a \cdot b) \cdot (b^{-1} \cdot a^{-1}) = ((a \cdot b) \cdot b^{-1}) \cdot a^{-1}.$$

Используя еще раз ассоциативность, определение элемента, обратного к данному, и свойства единицы, получим

$$((a\cdot b)\cdot b^{-1})\cdot a^{-1} = a\cdot(b\cdot b^{-1})\cdot a^{-1} = a\cdot a^{-1} = \mathbf{1}.$$

Итак, $(a\cdot b)\cdot(b^{-1}\cdot a^{-1}) = \mathbf{1}$. Точно так же доказывается, что $(b^{-1}\cdot a^{-1})(a\cdot b) = \mathbf{1}$. Поэтому элемент $b^{-1}\cdot a^{-1}$ является обратным к элементу $a\cdot b$. Согласно теореме 2.1, обратный элемент единственный, и поэтому $(a\cdot b)^{-1} = b^{-1}\cdot a^{-1}$. Второе из доказываемых равенств следует непосредственно из определения элемента, обратного к данному. Действительно, определение элемента a^{-1}, обратного к a, равенством $a^{-1}\cdot a = a\cdot a^{-1} = \mathbf{1}$ можно рассматривать как определение $(a^{-1})^{-1}$ — обратного элемента к a^{-1}, которым является, согласно этим равенствам, элемент a. В силу теоремы 2.1 он единственный, т.е. $a = (a^{-1})^{-1}$. ▶

Таким образом, мы установили, что элемент, обратный к произведению $a\cdot b$, равен $b^{-1}\cdot a^{-1}$, а элемент, обратный к элементу, обратному к a, равен a.

Теорема 2.3. В любой группе $\mathcal{G} = (G, \cdot, \mathbf{1})$ справедливы ***левый*** и ***правый законы сокращения***: если $a\cdot x = a\cdot y$, то $x = y$, и если $x\cdot a = y\cdot a$, то $x = y$.

◀ Пусть $a\cdot x = a\cdot y$. Умножая обе части этого равенства слева на элемент a^{-1}, получаем

$$a^{-1}\cdot(a\cdot x) = a^{-1}\cdot(a\cdot y).$$

В силу ассоциативности групповой операции последнее равенство можно записать так:

$$(a^{-1}\cdot a)\cdot x = (a^{-1}\cdot a)\cdot y.$$

Поскольку $a^{-1}\cdot a = \mathbf{1}$, то $\mathbf{1}\cdot x = \mathbf{1}\cdot y$, откуда $x = y$. Тем самым доказан левый закон сокращения. Аналогично доказывается и правый закон. ▶

Пусть $\mathcal{G} = (G, \cdot, \mathbf{1})$ — группа, a, b — фиксированные элементы G. Рассмотрим задачу решения уравнений

$$a \cdot x = b, \tag{2.1}$$

$$x \cdot a = b \tag{2.2}$$

в группе $\mathcal{G}$, т.е. поиска всех таких элементов $x \in G$, для которых уравнение (2.1) (или (2.2)) превращается в тождество.

Теорема 2.4. В любой группе $\mathcal{G}$ уравнения вида (2.1) и (2.2) имеют решения, и притом единственные.

◀ Покажем, что $x = a^{-1} \cdot b$ есть решение (2.1). Действительно, $a \cdot (a^{-1} \cdot b) = (a \cdot a^{-1} \cdot b) = b$.

Докажем единственность решения. Пусть для фиксированных a и b и некоторого x выполнено равенство $a \cdot x = b$. В группе для любого a существует и однозначно определен элемент a^{-1}, обратный к a. Умножив на него обе части равенства, получим $a^{-1} \cdot (a \cdot x) = a^{-1} \cdot b$. В силу ассоциативности преобразуем последнее равенство к виду $(a^{-1} \cdot a) \cdot x = a^{-1} \cdot b$. Поскольку $a^{-1} \cdot a = \mathbf{1}$, то $\mathbf{1} \cdot x = a^{-1} \cdot b$, откуда $x = a^{-1} \cdot b$. Это решение единственное в силу единственности обратного элемента.

Аналогично из $x \cdot a = b$ получаем $x = b \cdot a^{-1}$, и это решение также единственное. ▶

Замечание. При использовании аддитивной записи операции для коммутативной группы $\mathcal{G} = (G, +, \mathbf{0})$ оба написанных выше уравнения сводятся к одному:

$$a + x = b,$$

а его решение есть $x = b + (-a)$. Правую часть этого равенства в коммутативной группе называют ***разностью*** элементов b и a и обозначают $b - a$. Саму же операцию, сопоставляющую упорядоченной паре (a, b) разность $b - a$, называют операцией ***вычитания***. С учетом введенных обозначений решение уравнения в коммутативной группе можно записать так: $x = b - a$.

В случае коммутативной группы при употреблении для бинарной операции мультипликативной записи решения обоих уравнений имеют вид $x = b \cdot a^{-1}$. Выражение $b \cdot a^{-1}$ в коммутативной группе называют ***частным от деления*** b на a и обозначают $\frac{b}{a}$ (или b/a), а саму операцию называют операцией ***деления***. Решение уравнения в этом случае записывают в виде $x = \frac{b}{a}$ (или $x = b/a$).

Пример 2.10. Рассмотрим группу подстановок n-й степени S_n всех биекций n-элементного множества $\{1, 2, \ldots, n\}$. Произвольную биекцию σ из S_n обычно записывают в виде

$$\begin{pmatrix} 1 & 2 & \ldots & n \\ \alpha_1 & \alpha_2 & \ldots & \alpha_n \end{pmatrix},$$

обозначая тем самым, что образ 1 (при отображении σ) есть α_1, образ 2 есть α_2, ..., образ n есть α_n. Биекцию множества $\{1, \ldots, n\}$ на себя называют *подстановкой* этого множества. Подстановку, которая отображает α_1 в α_2, α_2 в α_3, ..., α_{k-1} в α_k, а α_k в α_1, где $1 \leqslant \alpha_1, \alpha_2, \ldots, \alpha_k \leqslant n$ и все α_j попарно различны, а все элементы, отличные от $\alpha_1, \ldots, \alpha_k$, отображаются сами в себя, называют ***циклом*** длины k и записывают ее в виде $(\alpha_1\ \alpha_2\ \ldots\ \alpha_k)$. Например, подстановку из группы S_4

$$\begin{pmatrix} 1 & 2 & 3 & 4 \\ 3 & 2 & 4 & 1 \end{pmatrix}$$

можно записать в виде (1 3 4).

Цикл длины 2 называют *транспозицией*. Транспозиция представляет такое отображение множества $\{1, \ldots, n\}$ в себя, при котором два элемента меняются местами, а остальные остаются на своих местах. Так, полная запись транспозиции (3 4) в S_4 будет иметь вид

$$\begin{pmatrix} 1 & 2 & 3 & 4 \\ 1 & 2 & 4 & 3 \end{pmatrix}.$$

Подстановка, обратная подстановке

$$\begin{pmatrix} 1 & 2 & \dots & n \\ \alpha_1 & \alpha_2 & \dots & \alpha_n \end{pmatrix},$$

есть подстановка, которая отображает α_1 в 1, α_2 в 2, ... α_n в n. Отметим, что при записи обратной подстановки элементы первой строки тем не менее записываются в обычном порядке: $1, \dots, n$.

В группе S_3 решим следующее уравнение:

$$\begin{pmatrix} 1 & 2 & 3 \\ 3 & 1 & 2 \end{pmatrix} \circ X \circ \begin{pmatrix} 1 & 2 & 3 \\ 2 & 3 & 1 \end{pmatrix} = \begin{pmatrix} 1 & 2 & 3 \\ 3 & 2 & 1 \end{pmatrix}.$$

Умножив обе части уравнения слева на

$$\begin{pmatrix} 1 & 2 & 3 \\ 3 & 1 & 2 \end{pmatrix}^{-1} = \begin{pmatrix} 1 & 2 & 3 \\ 2 & 3 & 1 \end{pmatrix},$$

получим

$$X \circ \begin{pmatrix} 1 & 2 & 3 \\ 2 & 3 & 1 \end{pmatrix} = \begin{pmatrix} 1 & 2 & 3 \\ 2 & 1 & 3 \end{pmatrix}.$$

Далее, умножив полученное уравнение справа на

$$\begin{pmatrix} 1 & 2 & 3 \\ 2 & 3 & 1 \end{pmatrix}^{-1} = \begin{pmatrix} 1 & 2 & 3 \\ 3 & 1 & 2 \end{pmatrix}$$

окончательно получим

$$X = \begin{pmatrix} 1 & 2 & 3 \\ 1 & 3 & 2 \end{pmatrix} = (2\,3). \quad \#$$

В полугруппе в общем случае законы сокращения и разрешимость уравнений типа (2.1) и (2.2) могут не иметь места. Например, в полугруппе квадратных матриц фиксированного порядка с операцией умножения матриц из матричного равенства $AX = AY$, вообще говоря, не следует, что $X = Y$. Это

можно утверждать лишь при дополнительном предположении, что $\det A \neq 0$. Можно доказать, что в свободном моноиде, порожденном некоторым конечным множеством, оба закона сокращения справедливы, но никаких обратных элементов не существует.

В полугруппе можно умножать любой элемент a сам на себя, причем в силу ассоциативности операции полугруппы элемент $\underbrace{a \cdot a \cdot \ldots \cdot a}_{n \text{ раз}}$ определен однозначно. Этот элемент называют n-й ***степенью элемента*** a и обозначают a^n. При этом $a^1 = a$, $a^n = a \cdot a^{n-1}$, $n = 2, 3, \ldots$

В моноиде вводят также нулевую степень элемента, полагая $a^0 = \mathbf{1}$.

Если $(A, \cdot, \mathbf{1})$ — группа, то можно ввести и отрицательные степени элемента согласно равенству $a^{-n} = (a^{-1})^n$, $n = 1, 2, \ldots$

Без доказательства сформулируем утверждения о свойствах степеней.

Теорема 2.5. Для любой полугруппы $a^m \cdot a^n = a^{m+n}$, $(a^m)^n = a^{mn}$ $(m, n \in \mathbb{N})$.

Теорема 2.6. Для любой группы $a^{-n} = (a^n)^{-1}$ $(n \in \mathbb{N})$, $a^m \cdot a^n = a^{m+n}$, $(a^m)^n = a^{mn}$ $(m, n \in \mathbb{Z})$.

Определение 2.4. Полугруппу (в частности, группу) $(A, \cdot)$ называют ***циклической***, если существует такой элемент a, что любой элемент x полугруппы является некоторой (целой) степенью элемента a. Элемент a называют ***образующим элементом полугруппы*** (***группы***).

Пример 2.11. **а.** Полугруппа $(\mathbb{N}_0, +, 0)$ циклическая, с образующим элементом 1. При аддитивной записи бинарной операции возведение элемента a в положительную степень n есть сумма n этих элементов, и это записывают $n \cdot a$ (или na, без знака умножения).

б. Группа $(\mathbb{Z}, +, 0)$ также циклическая. Для нее образующими элементами могут быть 1 и -1. Рассмотрим элемент 1.

Тогда $0 \cdot 1 = 0$, $n \cdot 1 = \underbrace{1 + \ldots + 1}_{n \text{ раз}} = n$ $(n > 0)$ и $(-1) \cdot 1 = -1$, $(-n) \cdot 1 = n \cdot (-1) = \underbrace{(-1) + \ldots + (-1)}_{n \text{ раз}} = -n$ $(n > 0)$.

Если в качестве образующего взять элемент -1, то $0 \cdot (-1) = 0$, отрицательные целые числа получаются как положительные „степени“ -1, а положительные — как отрицательные „степени“ -1. Например, $(-2) \cdot (-1) = 2$, $4 \cdot (-1) = -4$.

в. Группа $(\mathbb{Z}_3, \oplus_3, 0)$ вычетов по модулю 3 циклическая, причем любой ее ненулевой элемент является образующим.

Действительно, для 1 имеем $1 \oplus_3 1 = 2$, $1 \oplus_3 1 \oplus_3 1 = 0$, а для 2 получим $2^2 = 2 \oplus_3 2 = 1$, $2 \oplus_3 2 \oplus_3 2 = 0$. #

Изучим подробнее строение конечных циклических групп, используя мультипликативную запись бинарной операции. Напомним, что *конечная алгебра* (***конечная группа***, в частности) — это алгебра, носитель которой — конечное множество.

Порядком конечной группы называют количество элементов в этой группе.

Так, например, аддитивная группа вычетов по модулю k имеет порядок k. Симметрическая группа степени n, т.е. группа подстановок S_n, имеет порядок $n!$. Мультипликативная группа вычетов по модулю p, где p — простое число, имеет порядок $p - 1$.

Порядок элемента a циклической группы — это наименьшее положительное n, такое, что $a^n = \mathbf{1}$.

Теорема 2.7. Порядок образующего элемента конечной циклической группы равен порядку самой группы.

◀ Пусть $\mathcal{G} = (G, \cdot, \mathbf{1})$ — конечная циклическая группа с образующим элементом a и $n > 0$ — порядок этого элемента.

Тогда все степени $a^0 = \mathbf{1}, a^1 = a, \ldots, a^{n-1}$ попарно различны. Действительно, если $a^k = a^l$, $0 < l < k < n$, то $a^{k-l} = a^{k+(-l)} = a^k a^{-l} = a^l a^{-l} = a^{l-l} = \mathbf{1}$. Поскольку $k - l < n$, получено противоречие с выбором n как порядка элемента a (ибо найдена

степень, меньшая n, при возведении в которую элемента a получится единица).

Осталось доказать, что любая степень элемента a принадлежит множеству $\{\mathbf{1}, a, \ldots, a^{n-1}\}$. Для любого целого m существуют также целые n, k, такие, что $m = kn + q$, где q — целое и $0 \leqslant q < n$. Тогда $a^m = a^{kn+q} = a^{kn} \cdot a^q = (a^n)^k \cdot a^q = \mathbf{1} \cdot a^q = a^q \in \{\mathbf{1}, a, \ldots, a^{n-1}\}$. Поскольку каждый элемент группы G есть некоторая степень элемента a, то $G = \{\mathbf{1}, a, \ldots, a^{n-1}\}$ и порядок группы равен n. ▶

Из доказанной теоремы следует, что в бесконечной циклической группе не существует такого $n > 0$, что для образующего элемента a группы выполняется равенство $a^n = \mathbf{1}$.

2.3. Кольца, тела, поля

Определение 2.5. ***Кольцом*** называют *алгебру*

$$\mathcal{R} = (R, +, \cdot, \mathbf{0}, \mathbf{1}),$$

сигнатура которой состоит из двух *бинарных* и двух *нульарных операций*, причем для любых $a, b, c \in R$ выполняются равенства:

1) $a + (b + c) = (a + b) + c$;

2) $a + b = b + a$;

3) $a + \mathbf{0} = a$;

4) для каждого $a \in R$ существует элемент a', такой, что $a + a' = \mathbf{0}$;

5) $a \cdot (b \cdot c) = (a \cdot b) \cdot c$;

6) $a \cdot \mathbf{1} = \mathbf{1} \cdot a = a$;

7) $a \cdot (b + c) = a \cdot b + a \cdot c$, $(b + c) \cdot a = b \cdot a + c \cdot a$.

Операцию $+$ называют ***сложением кольца***, операцию $\cdot$ — ***умножением кольца***, элемент $\mathbf{0}$ — ***нулем кольца***, элемент $\mathbf{1}$ — ***единицей кольца***.

Равенства 1–7, указанные в определении, называют ***аксиомами кольца***. Рассмотрим эти равенства с точки зрения понятия *группы* и *моноида*.

Аксиомы кольца 1–4 означают, что алгебра $(R, +, \mathbf{0})$, сигнатура которой состоит только из операций сложения кольца $+$ и нуля кольца $\mathbf{0}$, является *абелевой группой*. Эту группу называют ***аддитивной группой кольца*** $\mathcal{R}$ и говорят также, что по сложению кольцо есть коммутативная (абелева) группа.

Аксиомы кольца 5 и 6 показывают, что алгебра $(R, \cdot, \mathbf{1})$, сигнатура которой включает только умножение кольца $\cdot$ и единицу кольца $\mathbf{1}$, есть моноид. Этот моноид называют ***мультипликативным моноидом кольца*** $\mathcal{R}$ и говорят, что по умножению кольцо есть моноид.

Связь между сложением кольца и умножением кольца устанавливает аксиома 7, согласно которой операция умножения дистрибутивна относительно операции сложения.

Учитывая сказанное выше, отметим, что кольцо — это алгебра с двумя бинарными и двумя нульарными операциями $\mathcal{R} = (R, +, \cdot, \mathbf{0}, \mathbf{1})$, такая, что:

1) алгебра $(R, +, \mathbf{0})$ — коммутативная группа;

2) алгебра $(R, \cdot, \mathbf{1})$ — моноид;

3) операция $\cdot$ (умножения кольца) дистрибутивна относительно операции $+$ (сложения кольца).

Замечание 2.2. В литературе встречается иной состав аксиом кольца, относящихся к умножению. Так, могут отсутствовать аксиома 6 (в кольце нет $\mathbf{1}$) и аксиома 5 (умножение не ассоциативно). В этом случае выделяют ассоциативные кольца (к аксиомам кольца добавляют требование ассоциативности умножения) и кольца с единицей. В последнем случае добавляются требования ассоциативности умножения и существования единицы.

Определение 2.6. Кольцо называют ***коммутативным***, если его операция умножения коммутативна.

Пример 2.12. а. Алгебра $(\mathbb{Z}, +, \cdot, 0, 1)$ есть коммутативное кольцо. Отметим, что алгебра $(\mathbb{N}_0, +, \cdot, 0, 1)$ кольцом

не будет, поскольку $(\mathbb{N}_0, +)$ — коммутативный моноид, но не группа.

б. Рассмотрим алгебру $\mathbb{Z}_k = (\{0, 1, \ldots, k-1\}, \oplus_k, \odot_k, 0, 1)$ $(k \geqslant 1)$ с операцией $\oplus_k$ сложения по модулю k и $\odot_k$ (умножения по модулю k). Последняя аналогична операции сложения по модулю k: $m \odot_k n$ равно остатку от деления на k числа $m \cdot n$. Эта алгебра есть коммутативное кольцо, которое называют ***кольцом вычетов по модулю*** k.

в. Алгебра $\left(2^A, \triangle, \cap, \varnothing, A\right)$ — коммутативное кольцо, что следует из свойств *пересечения* и *симметрической разности множеств* (см. с. 35).

г. Пример некоммутативного кольца дает множество всех квадратных матриц фиксированного порядка с операциями сложения и умножения матриц. Единицей этого кольца является единичная матрица, а нулем — нулевая.

д. Пусть $\mathcal{L}$ — линейное пространство. Рассмотрим множество всех линейных операторов, действующих в этом пространстве.

Напомним, что *суммой* двух *линейных операторов* $\boldsymbol{A}$ и $\boldsymbol{B}$ называют оператор $\boldsymbol{A}+\boldsymbol{B}$, такой, что

$$(\boldsymbol{A}+\boldsymbol{B})\boldsymbol{x} = \boldsymbol{A}\boldsymbol{x} + \boldsymbol{B}\boldsymbol{x}, \quad \boldsymbol{x} \in \mathcal{L}.$$

Произведением линейных операторов $\boldsymbol{A}$ и $\boldsymbol{B}$ называют линейный оператор $\boldsymbol{AB}$, такой, что $(\boldsymbol{AB})\boldsymbol{x} = \boldsymbol{A}(\boldsymbol{B}\boldsymbol{x})$ для любого $\boldsymbol{x} \in \mathcal{L}$.

Используя свойства указанных операций над линейными операторами, можно показать, что множество всех линейных операторов, действующих в пространстве $\mathcal{L}$, вместе с операциями сложения и умножения операторов образует кольцо. Нулем этого кольца служит *нулевой оператор*, а единицей — *тождественный оператор*.

Это кольцо называют ***кольцом линейных операторов*** в линейном пространстве $\mathcal{L}$. #

Аксиомы кольца называют также ***основными тождествами кольца***. Тождество кольца — это равенство, справед-

ливость которого сохраняется при подстановке вместо фигурирующих в нем переменных любых элементов кольца. Основные тождества постулируются, и из них затем могут быть выведены как следствия другие тождества. Рассмотрим некоторые из них.

Напомним, что аддитивная группа кольца коммутативна и в ней определена операция *вычитания*.

Теорема 2.8. В любом кольце выполняются следующие тождества:

1) $\mathbf{0}\cdot a = a\cdot\mathbf{0} = \mathbf{0}$;
2) $(-a)\cdot b = -(a\cdot b) = a\cdot(-b)$;
3) $(a-b)\cdot c = a\cdot c - b\cdot c, \quad c\cdot(a-b) = c\cdot a - c\cdot b.$

◀ Докажем тождество $\mathbf{0}\cdot a = \mathbf{0}$. Запишем для произвольного a:

$$a + \mathbf{0}\cdot a = \mathbf{1}\cdot a + \mathbf{0}\cdot a = (\mathbf{1}+\mathbf{0})\cdot a = \mathbf{1}\cdot a = a.$$

Итак, $a + \mathbf{0}\cdot a = a$. Последнее равенство можно рассматривать как уравнение в аддитивной группе кольца относительно неизвестного элемента $\mathbf{0}\cdot a$. Так как в аддитивной группе любое уравнение вида $a + x = b$ имеет единственное решение $x = b - a$, то $\mathbf{0}\cdot a = a - a = \mathbf{0}$. Тождество $a\cdot\mathbf{0} = \mathbf{0}$ доказывается аналогично.

Докажем теперь тождество $-(a\cdot b) = a\cdot(-b)$. Имеем

$$a\cdot(-b) + a\cdot b = a\cdot((-b)+b) = a\cdot\mathbf{0} = \mathbf{0},$$

откуда $a\cdot(-b) = -(a\cdot b)$. Точно так же можно доказать, что $(-a)\cdot b = -(a\cdot b)$.

Докажем третью пару тождеств. Рассмотрим первое из них. С учетом доказанного выше имеем

$$a\cdot(b-c) = a\cdot(b+(-c)) = a\cdot b + a\cdot(-c) = a\cdot b - a\cdot c,$$

т.е. тождество справедливо. Второе тождество этой пары доказывается аналогично. ▶

Следствие 2.1. В любом кольце справедливо тождество $(-\mathbf{1})\cdot x = x\cdot(-\mathbf{1}) = -x$.

◀ Указанное следствие вытекает из второго тождества теоремы 2.8 при $a = \mathbf{1}$ и $b = x$. ▶

Первые два тождества из доказанных в теореме 2.8 выражают свойство, называемое ***аннулирующим свойством нуля*** в кольце. Третья же пара тождеств указанной теоремы выражает свойство дистрибутивности операции умножения кольца относительно операции вычитания. Таким образом, производя вычисления в любом кольце, можно раскрывать скобки и менять знаки так же, как и при сложении, вычитании и умножении действительных чисел.

Ненулевые элементы a и b кольца $\mathcal{R}$ называют ***делителями нуля***, если $a\cdot b = \mathbf{0}$ или $b\cdot a = \mathbf{0}$. Пример кольца с делителем нуля дает любое *кольцо вычетов по модулю* k, если k — составное число. В этом случае произведение по модулю k любых m и n, дающих при обычном перемножении число, кратное k, будет равно нулю. Например, в кольце вычетов по модулю 6 элементы 2 и 3 являются делителями нуля, поскольку $2 \odot_6 3 = 0$. Другой пример дает кольцо квадратных матриц фиксированного порядка (не меньшего двух). Например, для матриц второго порядка имеем

$$\begin{pmatrix} 0 & a \\ 0 & 0 \end{pmatrix}\begin{pmatrix} 0 & b \\ 0 & 0 \end{pmatrix} = \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix}.$$

При отличных от нуля a и b приведенные матрицы являются делителями нуля.

По умножению кольцо является только моноидом. Поставим вопрос: в каких случаях кольцо по умножению будет группой? Прежде всего заметим, что множество всех элементов кольца, в котором $\mathbf{0} \neq \mathbf{1}$, не может образовывать группы по умножению, так как нуль не может иметь обратного. Действительно, если предположить, что такой элемент $\mathbf{0}'$ существует, то, с одной

стороны, $\mathbf{0}\cdot\mathbf{0}'=\mathbf{0}'\cdot\mathbf{0}=\mathbf{1}$, а с другой — $\mathbf{0}\cdot\mathbf{0}'=\mathbf{0}'\cdot\mathbf{0}=\mathbf{0}$, откуда $\mathbf{0}=\mathbf{1}$. Это противоречит условию $\mathbf{0}\neq\mathbf{1}$. Таким образом, поставленный выше вопрос можно уточнить так: в каких случаях множество всех ненулевых элементов кольца образует группу по умножению?

Если в кольце имеются делители нуля, то подмножество всех ненулевых элементов кольца не образует группы по умножению уже хотя бы потому, что это подмножество не замкнуто относительно операции умножения, т.е. существуют ненулевые элементы, произведение которых равно нулю.

Кольцо, в котором множество всех ненулевых элементов по умножению образует группу, называют ***телом***, коммутативное тело — ***полем***, а группу ненулевых элементов тела (поля) по умножению — ***мультипликативной группой*** этого ***тела (поля)***. Согласно определению, поле есть частный случай кольца, в котором операции обладают дополнительными свойствами. Выпишем все свойства, выполнение которых требуется для операций поля. Их еще называют ***аксиомами поля***.

Поле есть алгебра $\mathcal{F}=(F,+,\cdot,\mathbf{0},\mathbf{1})$, сигнатура которой состоит из двух бинарных и двух нульарных операций, причем справедливы тождества:

1) $a+(b+c)=(a+b)+c$;

2) $a+b=b+a$;

3) $a+\mathbf{0}=a$;

4) для каждого $a\in F$ существует элемент $-a$, такой, что $a+(-a)=\mathbf{0}$;

5) $a\cdot(b\cdot c)=(a\cdot b)\cdot c$;

6) $a\cdot b=b\cdot a$;

7) $a\cdot\mathbf{1}=\mathbf{1}\cdot a=a$;

8) для каждого $a\in F$, отличного от $\mathbf{0}$, существует элемент a^{-1}, такой, что $a\cdot a^{-1}=\mathbf{1}$;

9) $a\cdot(b+c)=a\cdot b+a\cdot c$.

Пример 2.13. а. Алгебра $(\mathbb{Q},+,\cdot,0,1)$ есть поле, называемое ***полем рациональных чисел***.

б. Алгебры $(\mathbb{R}, +, \cdot, 0, 1)$ и $(\mathbb{C}, +, \cdot, 0, 1)$ есть поля, называемые ***полями действительных*** и ***комплексных чисел*** соответственно.

в. Примером тела, не являющегося полем, может служить алгебра ***кватернионов*** (см. **Д.2.1**). #

Итак, мы видим, что известным законам сложения и умножения чисел соответствуют аксиомы поля. Занимаясь числовыми расчетами, мы „работаем в полях“, а именно имеем дело преимущественно с полями рациональных и вещественных чисел, иногда „переселяемся“ в поле комплексных чисел.

2.4. Области целостности

Областью целостности называют *коммутативное кольцо без делителей нуля*. Так, кольцо целых чисел есть область целостности.

Теорема 2.9. Конечная область целостности является *полем*.

◀ Поле — это кольцо, умножение которого коммутативно, а каждый ненулевой элемент a имеет *обратный элемент* относительно *умножения*. Так как область целостности, по определению, является коммутативным кольцом, то достаточно доказать, что для конечной области целостности любой ненулевой элемент обратим, т.е. для всякого $a \neq \mathbf{0}$ существует единственный x, такой, что $a \cdot x = \mathbf{1}$.

Фиксируем произвольный элемент $a \neq \mathbf{0}$ и определяем отображение f_a множества всех ненулевых элементов в себя по формуле $f_a(x) = a \cdot x$ ($a \cdot x \neq \mathbf{0}$ в области целостности при $a \neq \mathbf{0}$ и $x \neq \mathbf{0}$). Отображение f_a является инъекцией, поскольку из равенства $a \cdot x = a \cdot y$ вытекает равенство $a \cdot (x - y) = \mathbf{0}$, откуда ввиду отсутствия делителей нуля $x - y = \mathbf{0}$ и $x = y$. Так как *носитель* по условию теоремы конечен, то, согласно теореме 1.8, f_a также и биекция. Поэтому для любого y существует единственный элемент x, такой, что $y = a \cdot x$. В частности, при $y = \mathbf{1}$

равенство $a \cdot x = 1$ выполнено для некоторого однозначно определенного x, т.е. $x = a^{-1}$. ▶

Доказательство теоремы 2.9 опирается на условие конечности кольца. Это условие действительно важно. Пример кольца целых чисел показывает, что бесконечная область целостности может и не быть полем.

Теорема 2.9 имеет интересные следствия. Рассмотрим ***кольцо $\mathbb{Z}_p$ вычетов по модулю p.***

Следствие 2.2. Кольцо $\mathbb{Z}_p$ вычетов по модулю p является полем тогда и только тогда, когда p — простое число.

◀ Пусть $\mathbb{Z}_p$ является полем. Покажем, что в этом случае число p простое. Предположим, что оно составное. Тогда найдутся такие числа k и l, $0 < k, l \leqslant p-1$, что $p = k \cdot l$. Поскольку в этом случае $k \cdot l = 0 (\mathrm{mod}\, p)$, по крайней мере числа k и l являются в кольце $\mathbb{Z}_p$ делителями нуля и $\mathbb{Z}_p$ — не поле. Следовательно, число p не может быть составным.

Пусть p — простое число. Предположим, что элементы m и n кольца $\mathbb{Z}_p$ будут делителями нуля, т.е. $m \cdot n = 0 (\mathrm{mod}\, p)$. При простом p равенство произведения $m \cdot n$ нулю по модулю p означает, что либо m делится на p, либо n делится на p, т.е. либо $m = 0 (\mathrm{mod}\, p)$, либо $n = 0 (\mathrm{mod}\, p)$. Учитывая неравенства $0 \leqslant m \leqslant p-1$ и $0 \leqslant n \leqslant p-1$, заключаем, что либо $m = 0$, либо $n = 0$. Таким образом, при простом p делителей нуля нет и кольцо $\mathbb{Z}_p$, как конечная область целостности, является полем. ▶

Мультипликативную группу ***поля $\mathbb{Z}_p$ вычетов по модулю p*** обозначают $\mathbb{Z}_p^*$ и называют ***мультипликативной группой вычетов по модулю p.***

Для произвольного p легко видеть, что ненулевые элементы m и n кольца $\mathbb{Z}_p$ будут делителями нуля тогда и только тогда, когда произведение $m \cdot n$ делится на p (т.е. $m \cdot n = 0 (\mathrm{mod}\, p)$). Например, в кольце Z_{12} делителями нуля будут элементы 2 и 6, 3 и 4, 3 и 8, 4 и 6, 4 и 9, 6 и 6, 6 и 8, 6 и 10, 8 и 9.

Замечание 2.3. Следствие 2.2 допускает интерпретацию с точки зрения теории чисел: каково бы ни было простое число p, для всякого ненулевого $m < p$ найдется единственное ненулевое $n < p$, такое, что $mn = 1 \pmod{p}$. Этот результат имеет место именно в силу того, что для каждого элемента поля $\mathbb{Z}_p$ есть обратный элемент относительно умножения. Это — один из примеров применения общей алгебры к теории чисел.

Пример 2.14. В заключение приведем „таблицу сложения“ (табл. 2.1) и „таблицу умножения“ (табл. 2.2) для поля $\mathbb{Z}_5$

Таблица 2.1

$\oplus_5$	0	1	2	3	4
0	0	1	2	3	4
1	1	2	3	4	0
2	2	3	4	0	1
3	3	4	0	1	2
4	4	0	1	2	3

Таблица 2.2

$\odot_5$	0	1	2	3	4
0	0	0	0	0	0
1	0	1	2	3	4
2	0	2	4	1	3
3	0	3	1	4	2
4	0	4	3	2	1

Таблицы, подобные приведенным выше, которые определяют операции в конечных алгебрах, носят название ***таблиц Кэли***. Из таблиц Кэли для поля вычетов по модулю 5 следует, что в этом поле выполняются слегка шокирующие при первом взгляде равенства: $4 = -1$, $2 = 3^{-1}$, $4 = 4^{-1}$ и т.п. Но ни о каких „отрицательных“ числах и ни о каких „дробях“ тут речи нет, поскольку расматриваются другие объекты — остатки при делении на 5. Просто равенство $4 = -1$ означает, что элемент 1 есть элемент, противоположный 4 в аддитивной группе вычетов по модулю 5: $4 \oplus_5 1 = 0$. Аналогично по умножению — в мультипликативной группе вычетов по модулю 5 элемент 3 есть обратный к 2 , так как $3 \odot_5 2 = 1$, а элемент 4 обратен к себе самому.

Пример 2.15. Рассмотрим пример решения *системы линейных алгебраических уравнений* в поле $\mathbb{Z}_5$. При записи урав-

нений будем опускать знак $\odot_5$ умножения там, где это не приводит к недоразумениям. Будем решать систему

$$\begin{cases} x_1 \oplus_5 2x_2 \oplus_5 3x_3 = 1, \\ 2x_1 \oplus_5 2x_2 \oplus_5 4x_3 = 3, \\ 4x_1 \oplus_5 3x_2 \oplus_5 x_3 = 0, \end{cases}$$

используя метод Гаусса [III]. Домножив первую строку на 3 и прибавив ее ко второй строке, получим

$$(3 \oplus_5 2)x_1 \oplus_5 (3 \odot_5 2 \oplus_5 2)x_2 \oplus_5 (3 \odot_5 3 \oplus_5 4)x_3 = 3 \oplus_5 3.$$

Воспользовавшись таблицами Кэли, вычислим коэффициенты при переменных. В итоге имеем

$$0 \odot_5 x_1 \oplus_5 3x_2 \oplus_5 3x_3 = 1.$$

Прибавив к третьей строке первую, получим

$$(1 \oplus_5 4)x_1 \oplus_5 (2 \oplus_5 3)x_2 \oplus_5 (3 \oplus_5 1)x_3 = 1,$$

откуда $4x_3 = 1$.

Система привелась к виду

$$\begin{cases} x_1 \oplus_5 2x_2 \oplus_5 3x_3 = 1, \\ 3x_2 \oplus_5 3x_3 = 1, \\ 4x_3 = 1. \end{cases}$$

Из последнего уравнения находим $x_3 = 4^{-1} \odot_5 1 = 4 \odot_5 1 = 4$. Подставив $x_3 = 4$ во второе уравнение, будем иметь

$$3x_2 \oplus_5 3 \odot_5 4 = 1,$$

т.е. $3x_2 = 1 \oplus_5 (-2) = -1 = 4$. Отсюда

$$x_2 = 3^{-1} \odot_5 4 = 2 \odot_5 4 = 3.$$

Из первого уравнения после подстановки найденных значений переменных получим

$$x_1 \oplus_5 2 \odot_5 3 \oplus_5 3 \odot_5 4 = 1,$$

откуда $x_1 \oplus_5 1 \oplus_5 2 = 1$ и $x_1 = -2 = 3$.

Таким образом, $x_1 = 3$, $x_2 = 3$ и $x_3 = 4$ — решение системы линейных уравнений. #

Заметим в заключение, что известная из [III] техника решения систем линейных алгебраических уравнений в полях действительных или комплексных чисел может быть без изменения перенесена на любое поле.

2.5. Модули и линейные пространства

Рассмотрим *абелеву группу* $\mathcal{G} = (G, +, \mathbf{0})$ и *кольцо* $\mathcal{R} = (R, +, \cdot, \mathbf{0}, \mathbf{1})$. Пусть каждому элементу α кольца $\mathcal{R}$ сопоставлено *отображение* ω_α *носителя* группы $\mathcal{G}$ в себя так, что для любых $\alpha, \beta \in R$ и любых $x, y \in G$ выполняются равенства:

1) $\omega_\alpha(x+y) = \omega_\alpha(x) + \omega_\alpha(y)$;
2) $\omega_{\alpha+\beta}(x) = \omega_\alpha(x) + \omega_\beta(x)$;
3) $\omega_{\alpha\cdot\beta}(x) = \omega_\alpha(\omega_\beta(x))$;
4) $\omega_{\mathbf{1}}(x) = x$.

Последнее равенство означает, что отображение $\omega_{\mathbf{1}}$, сопоставленное *единице кольца* $\mathcal{R}$, является *тождественным отображением* множества G на себя.

Тогда абелева группа $\mathcal{G}$ называется ***левым модулем над кольцом*** $\mathcal{R}$.

Если равенство 3 в определении модуля переписать так:

3′) $\omega_{\alpha\cdot\beta}(x) = \omega_\beta(\omega_\alpha(x))$,

то получим определение ***правого модуля над кольцом*** $\mathcal{R}$.

Для коммутативного кольца $\mathcal{R}$ левый и правый модули совпадают, так как $\omega_\alpha(\omega_\beta(x)) = \omega_{\beta\cdot\alpha}(x) = \omega_\beta(\omega_\alpha(x))$.

Заметим, что модуль можно рассматривать как *алгебру* с бесконечной *сигнатурой*, если множество R бесконечно:

$$\mathcal{G} = (G, +, \mathbf{0}, \{\omega_\alpha \colon \alpha \in \mathcal{R}\}).$$

Следует подчеркнуть, что модуль есть именно абелева группа с дополнительными операциями, отображениями ω_α, сопоставленными элементам кольца $\mathcal{R}$. Носитель модуля есть носитель G группы $\mathcal{G}$.

Теорема 2.10. В любом $\mathcal{R}$-модуле имеют место тождества:

1) $\omega_{\mathbf{0}}(x) = \mathbf{0}$;
2) $\omega_{-\mathbf{1}}(x) = -x$.

◀ 1) $x + \omega_{\mathbf{0}}(x) = \omega_{\mathbf{1}}(x) + \omega_{\mathbf{0}}(x) = \omega_{\mathbf{1}+\mathbf{0}}(x) = \omega_{\mathbf{1}}(x) = x$.

Решая уравнение $x + \omega_{\mathbf{0}}(x) = x$ относительно $\omega_{\mathbf{0}}(x)$, получаем $\omega_{\mathbf{0}}(x) = x - x = \mathbf{0}$.

2) $x + \omega_{-\mathbf{1}}(x) = \omega_{\mathbf{1}}(x) + \omega_{-\mathbf{1}}(x) = \omega_{\mathbf{1}+(-\mathbf{1})}(x) = \omega_{\mathbf{1}}(x) = \mathbf{0}$. Таким образом, $x + \omega_{-\mathbf{1}}(x) = \mathbf{0}$, откуда в силу определения *противоположного* элемента получаем $\omega_{-\mathbf{1}}(x) = -x$. ▶

Пример 2.16. **а.** Пусть $\mathcal{R} = (R, +, \cdot, \mathbf{0}, \mathbf{1})$ — произвольное кольцо. В качестве группы $\mathcal{G}$ возьмем *аддитивную группу* этого *кольца*, а отображение ω_α, $\alpha \in \mathbb{R}$, определим так, что $\omega_\alpha(x) = \alpha \cdot x$, $x \in \mathbb{R}$. Это отображение называют ***левым сдвигом*** на α. Тогда равенства 1–4 выполнены в силу *аксиом кольца*. Таким образом, получаем левый модуль, носитель которого — аддитивная группа кольца, а отображение ω_α есть левый сдвиг произвольного элемента кольца на заданное α.

Если теперь задать для каждого $\alpha \in R$ отображение $\widetilde{\omega}_\alpha \colon R \to R$ так, что $\widetilde{\omega}_\alpha(x) = x \cdot \alpha$ (***правый сдвиг*** на α), то получим правый модуль с тем же носителем, но сигнатура его (помимо операции сложения исходного кольца $\mathbb{R}$) будет состоять из всевозможных правых сдвигов $\widetilde{\omega}_\alpha$.

б. Пусть $\mathcal{G}$ есть аддитивная группа векторов какого-либо линейного пространства L, а $\mathcal{R}$ — *кольцо линейных операторов*

из L в L. Тогда, полагая для произвольных линейного оператора A и вектора $\boldsymbol{x}$ пространства L $\omega_A(\boldsymbol{x}) = A\boldsymbol{x}$, получаем, как нетрудно проверить, левый модуль над кольцом $\mathcal{R}$.

в. Пусть $\mathcal{R}$ — кольцо квадратных числовых матриц порядка n с обычными операциями сложения и умножения матриц (см. пример 2.12.г), а $\mathcal{G}$ — группа матриц-столбцов типа $n \times 1$ по сложению. Отображение ω_α определим по правилу $\omega_\alpha(X) = AX$, где A — квадратная матрица, а X — вектор-столбец. Легко видеть, что равенства 1–4 вытекают из свойств умножения матриц и линейных операций над матрицами [III].

В результате получим левый модуль над кольцом квадратных матриц.

Аналогично, взяв в качестве $\mathcal{G}$ аддитивную группу матриц-строк типа $1 \times n$ и определив отображение $\widetilde{\omega}_\alpha(Y) = YA$, где A — квадратная матрица порядка n, а Y — матрица-строка, получим правый модуль над кольцом квадратных матриц. #

Если рассматривается левый $\mathcal{R}$-модуль, то отображение ω_α называют ***левым умножением*** на элемент α кольца $\mathcal{R}$ и применяют обозначение $\omega_\alpha(x) = \alpha \circ x$. Для правого $\mathcal{R}$-модуля отображение ω_α называют ***правым умножением*** на элемент α кольца $\mathcal{R}$ и пишут $\omega_\alpha(x) = x \circ \alpha$.

$\mathcal{R}$-модуль, у которого кольцо $\mathcal{R}$ является полем, называют ***линейным пространством над полем*** $\mathcal{R}$. Если кольцо $\mathcal{R}$ является *полем действительных чисел* (или *полем комплексных чисел*), то мы получаем действительное (соответственно комплексное) линейное пространство.

2.6. Подгруппы и подкольца

Пусть $\mathcal{G} = (G, *)$ — произвольный *группоид* и $H \subseteq G$ — некоторое подмножество множества G. Рассмотрим свойства *бинарной операции* $*$ группоида $\mathcal{G}$ на подмножестве H.

Говорят, что ***множество*** $H \subseteq G$ ***замкнуто относительно операции*** $*$, если $x * y \in H$ для любых x, $y \in H$. В

этом случае подмножество H с операцией $*$ будет группоидом $\mathcal{H} = (H, *)$. Его называют ***подгруппоидом*** группоида $\mathcal{G}$.

Если подмножество H замкнуто относительно бинарной операции $*$ и эта *бинарная операция ассоциативна* на множестве G, то легко убедиться, что операция останется ассоциативной и при ее ограничении на подмножество H. Таким образом, если группоид $\mathcal{G}$ является *полугруппой*, то и всякий его подгруппоид будет полугруппой, называемой ***подполугруппой*** полугруппы $\mathcal{G}$.

Однако в случае, когда группоид является *моноидом* (*группой*), уже нельзя утверждать, что любой подгруппоид является также моноидом (группой). Например, в качестве исходного группоида рассмотрим *аддитивную группу целых чисел* $(\mathbb{Z}, +)$. Выделим в множестве целых чисел подмножество $\mathbb{N}$ натуральных чисел. Поскольку это подмножество замкнуто относительно операциии сложения $+$, группоид $(\mathbb{N}, +)$ будет подгруппоидом группоида $(\mathbb{Z}, +)$. Так как операция сложения чисел ассоциативна, $(\mathbb{N}, +)$ будет подполугруппой. Однако в множестве $\mathbb{N}$ отсутствует *нейтральный элемент* 0 относительно операции сложения. Следовательно, $(\mathbb{N}, +)$ даже не моноид.

Пусть $\mathcal{M} = (M, \cdot, \mathbf{1})$ — моноид. Если P есть подмножество M, замкнутое относительно бинарной операции $\cdot$ моноида $\mathcal{M}$ и содержащее нейтральный элемент (*единицу*) $\mathbf{1}$ этого моноида, то $\mathcal{P} = (P, \cdot, \mathbf{1})$ также есть моноид. Его называют ***подмоноидом*** моноида $\mathcal{M}$.

Полагая, по определению, что замкнутость подмножества $B \subseteq A$ относительно нульарной операции a на A равносильна соотношению $a \in B$, получаем, что моноид $\mathcal{P} = (P, \cdot, \mathbf{1})$ есть подмоноид моноида $\mathcal{M} = (M, \cdot, \mathbf{1})$ тогда и только тогда, когда множество P замкнуто относительно бинарной операции $\cdot$ моноида $\mathcal{M}$, а также относительно его нульарной операции $\mathbf{1}$.

Пусть $\mathcal{G} = (G, \cdot, {}^{-1}, \mathbf{1})$ — группа, а H есть подмножество G, замкнутое относительно операции $\cdot$ группы $\mathcal{G}$, содержащее нейтральный элемент (*единицу*) $\mathbf{1}$ этой группы и вместе с каждым элементом $x \in H$ содержащее *элемент* x^{-1}, *обратный*

к x, т.е. замкнутое относительно унарной операции $^{-1}$ взятия обратного, которая здесь включена в *сигнатуру* группы. Тогда $\mathcal{H} = (H, \cdot, ^{-1}, \mathbf{1})$ также есть группа, которую называют ***подгруппой*** группы $\mathcal{G}$.

Пусть ω — унарная операция на множестве G моноида $\mathcal{G}$, а $\mathcal{H}$ — некоторый его подмоноид. Естественно подмоноид $\mathcal{H}$ моноида $\mathcal{G}$ назвать замкнутым относительно унарной операции ω, если для каждого $x \in H$ имеет место $\omega(x) \in H$. Тогда группа $\mathcal{H} = (H, \cdot, ^{-1}, \mathbf{1})$ есть подгруппа группы $\mathcal{G} = (G, \cdot, ^{-1}, \mathbf{1})$ в том и только в том случае, когда множество H замкнуто относительно всех операций $\cdot$, $^{-1}$, $\mathbf{1}$ сигнатуры группы $\mathcal{G}$.

Замечание 2.4. Подмножество $H \subseteq G$, замкнутое относительно *группового умножения* $\cdot$ группы $\mathcal{G}$ и содержащее вместе с каждым элементом x обратный к нему элемент x^{-1}, будет содержать и нейтральный элемент (единицу) группы, поскольку в силу замкнутости H относительно операции умножения из $x \in H$ и $x^{-1} \in H$ следует, что $x \cdot x^{-1} = x^{-1} \cdot x = \mathbf{1} \in H$. #

Используя факт единственности нейтрального элемента (единицы) любого моноида и только что сформулированное определение, можно легко доказать, что единица моноида (группы, в частности) служит одновременно единицей любого его подмоноида (любой подгруппы). Заметим, что подмоноид, *носитель* которого содержит только единицу исходного моноида ($P = \{\mathbf{1}\}$), а также подмоноид, носитель которого совпадает с носителем исходного моноида ($P = M$), называют ***тривиальным подмоноидом*** (в частности, ***тривиальной подгруппой***). Подмоноид, не являющийся тривиальным, называют ***нетривиальным подмоноидом*** (в частности, ***нетривиальной подгруппой***). Подгруппоид (подполугруппу, подмоноид, подгруппу) $(G, *)$ называют ***собственным подгруппоидом*** (***подполугруппой***, ***подмоноидом***, ***подгруппой***) группоида (полугруппы, моноида, группы) $(K, *)$, если его носитель G есть *собственное подмножество* множества K.

Пример 2.17. Рассмотрим аддитивную полугруппу натуральных чисел вместе с нулем $(\mathbb{N}_0, +)$. Подмножество всех положительных четных чисел замкнуто относительно сложения, и поэтому на нем может быть определена подполугруппа полугруппы $(\mathbb{N}_0, +)$. Но аддитивная полугруппа натуральных чисел с нулем является также и моноидом с нейтральным элементом 0. Тогда построенная выше подполугруппа всех положительных четных чисел не будет подмоноидом моноида $(\mathbb{N}_0, +, 0)$, так как ее носитель не содержит нуля — единицы моноида $(\mathbb{N}_0, +, 0)$.

Подмножество всех натуральных чисел вместе с нулем, делящихся на заданное число $k > 1$, замкнуто относительно операции сложения; на нем может быть определен подмоноид моноида $(\mathbb{N}_0, +, 0)$.

Мультипликативная группа поля рациональных чисел, является подгруппой группы $(\mathbb{R} \setminus \{0\}, \cdot, 1)$ (*мультипликативной группы поля* действительных чисел). Но алгебра $(\mathbb{Z} \setminus \{0\}, \cdot, 1)$ не является подгруппой последней группы. Несмотря на то что множество всех отличных от нуля целых чисел замкнуто относительно операции умножения и содержит единицу, оно не содержит вместе с каждым целым числом m обратного к нему числа $\frac{1}{m}$. #

Пусть $\mathcal{G} = (G, \cdot, {}^{-1}, \mathbf{1})$ — группа. Как следует из теорем 2.5 и 2.6, произведение любых *степеней элемента* a есть снова некоторая степень элемента a, нулевая степень дает единицу группы, а обратным к элементу a^k является элемент a^{-k}. Таким образом, множество всех степеней фиксированного элемента a группы $\mathcal{G}$ является подгруппой группы $\mathcal{G}$.

Определение 2.7. Подгруппу группы $\mathcal{G}$, заданную на множестве всех степеней фиксированного элемента a, называют ***циклической подгруппой*** группы $\mathcal{G}$, ***порожденной элементом*** a.

Пример 2.18. В группе $\mathbb{Z}_{13}^*$ (*мультипликативной группе вычетов по модулю* 13) построим циклическую подгруппу, порожденную элементом 5. Имеем: $5^0 = 1$, $5^1 = 5$, $5^2 = 5 \odot_{13} 5 = 12$, $5^3 = 5 \odot_{13} 12 = 8$, $5^4 = 5 \odot_{13} 8 = 1$. Отсюда следует, что порядок этой циклической подгруппы в силу теоремы 2.7 равен 4. Она состоит из элементов: 1, 5, 8 и 12. #

Рассмотрим *кольцо* $\mathcal{R} = (R, +, \cdot, \mathbf{0}, \mathbf{1})$. Если множество Q есть подмножество множества R, замкнутое относительно операций сложения и умножения кольца $\mathcal{R}$, содержащее *нуль* и единицу кольца $\mathcal{R}$, а также вместе с каждым $x \in Q$ содержащее *противоположный* к нему элемент $-x$, то $\mathcal{Q} = (Q, +, \cdot, \mathbf{0}, \mathbf{1})$ также есть кольцо. Его называют ***подкольцом*** кольца $\mathcal{G}$.

Другими словами, кольцо $\mathcal{Q} = (Q, +, \cdot, \mathbf{0}, \mathbf{1})$ — это подкольцо кольца $\mathcal{R} = (R, +, \cdot, \mathbf{0}, \mathbf{1})$, если его аддитивная группа есть подгруппа *аддитивной группы кольца* $\mathcal{R}$, а его мультипликативный моноид — подмоноид *мультипликативного моноида кольца* $\mathcal{R}$.

Аналогично определяется понятие ***подполя*** (какого-либо *поля*). Единственное по сравнению с определением подкольца дополнительное требование состоит в том, что носитель подполя должен вместе с каждым элементом x содержать обратный к нему по умножению поля элемент x^{-1}. Это значит, что мультипликативная группа подполя должна быть подгруппой мультипликативной группы всего поля. Естественно, что точно так же обстоит дело и с понятием ***подтела***.

Пример 2.19. Кольцо целых чисел $(\mathbb{Z}, +, \cdot, 0, 1)$ есть подкольцо кольца действительных чисел $(\mathbb{R}, +, \cdot, 0, 1)$. При этом, несмотря на то что кольцо действительных чисел есть поле, кольцо целых чисел не является его подполем, поскольку в последнем для любого целого числа отсутствует обратный к нему по умножению элемент.

Поле рациональных чисел является подполем поля действительных чисел, которое, в свою очередь, есть подполе поля

комплексных чисел. Алгебра $(\mathbb{N}_0, +, \cdot, 0, 1)$ на множестве натуральных чисел вместе с нулем не является подкольцом ни одного из перечисленных выше колец, так как ее носитель не содержит ни обратных относительно сложения, ни обратных относительно умножения элементов.

2.7. Теорема Лагранжа

Пусть $\mathcal{G} = (G, \cdot, 1)$ — *группа*, а $\mathcal{H} = (H, \cdot, 1)$ — ее *подгруппа*.

Левым смежным классом подгруппы $\mathcal{H}$ ***по элементу*** $a \in G$ называют множество

$$aH = \{y\colon y = a \cdot h,\ h \in H\}.$$

Соответственно ***правый смежный класс подгруппы*** $\mathcal{H}$ ***по элементу*** $a \in G$ — это множество

$$Ha = \{y\colon y = h \cdot a,\ h \in H\}.$$

Очевидно, что в коммутативной группе $aH = Ha$.

Замечание. При использовании *аддитивной записи* групповой *операции* смежные классы записываются в виде $a + H$ (или $H + a$). #

Рассмотрим левые смежные классы. Прежде всего заметим, что если $a \in H$, то $aH = H$. Действительно, если $x \in aH$, то для некоторого $h \in H$ $x = ah$, а так как $a \in H$ и множество H замкнуто относительно *умножения группы* $\mathcal{G}$, то $x \in H$. Обратно, если $x \in H$, то $x = aa^{-1}x = ah$, где $h = a^{-1}x \in H$. Поэтому $x \in aH$. Окончательно получим $H = aH$.

Введем теперь *бинарное отношение* $\sim_H$ на множестве G следующим образом: *элементы* a и b *связаны отношением* $\sim_H$ ($a \sim_H b$), если и только если левые смежные классы подгруппы $\mathcal{H}$ по элементам a и b совпадают ($aH = bH$).

Теорема 2.11. Бинарное отношение $\sim_H$ есть *эквивалентность* на G, причем *класс эквивалентности* произвольного элемента $a \in G$ совпадает с левым смежным классом aH.

◀ Докажем, что $\sim_H$ является эквивалентностью на G. Поскольку $aH = aH$ для любого $a \in G$, т.е. $a \sim_H a$, то бинарное отношение $\sim_H$ *рефлексивно*. Если $a \sim_H b$, то $aH = bH$, следовательно, $bH = aH$ и $b \sim_H a$, т.е. бинарное отношение $\sim_H$ *симметрично*. Наконец, из того, что $a \sim_H b$ и $b \sim_H c$, следует $aH = bH$ и $bH = cH$, т.е. $aH = cH$ и $a \sim_H c$, откуда вытекает, что бинарное отношение $\sim_H$ *транзитивно*. Итак, $\sim_H$ есть эквивалентность.

Докажем, что класс эквивалентности произвольного элемента a равен aH. Воспользуемся методом двух включений.

Пусть $x \in [a]_{\sim_H}$, т.е. $x \sim_H a$, тогда $xH = aH$. Последнее означает, что любой элемент вида ah, $h \in H$, может быть представлен в виде xh_1, где $h_1 \in H$, т.е. $ah = xh_1$. Отсюда получаем $x = ahh_1^{-1}$. Поскольку $\mathcal{H}$ — подгруппа, h, $h_1 \in H$, то $h_2 = hh_1^{-1} \in H$. Следовательно, $x = ah_2 \in aH$ и $[a]_{\sim_H} \subseteq aH$.

Покажем второе включение, т.е. докажем, что $aH \subseteq [a]_{\sim_H}$. Пусть $x \in aH$, тогда $x = ah$ для некоторого $h \in H$. Отсюда получаем, что $xH = ahH$. Поскольку для всякого $h \in H$, как доказано выше, $hH = H$, справедливо равенство $xH = aH$, откуда $x \sim_H a$ и $x \in [a]_{\sim_H}$. ▶

Теорема 2.12. Всякий левый смежный класс подгруппы H *равномощен* H.

◀ Для произвольного фиксированного $a \in G$ зададим отображение φ_a: $H \to aH$ следующим образом: $\varphi_a(h) = ah$. Во-первых, отображение φ_a есть сюръекция, так как если $x \in aH$, то $x = ah$ для некоторого $h \in H$, откуда $x = \varphi_a(h)$. Во-вторых, φ_a — инъекция, поскольку из равенства $ah_1 = ah_2$ в силу законов сокращения в группе $\mathcal{G}$ следует $h_1 = h_2$. Следовательно, φ_a — биекция и $|aH| = |H|$. ▶

Из доказанных теорем о свойствах левых смежных классов, справедливых — подчеркнем это — для любой группы, вытекает простой, но очень важный результат для конечных групп.

Теорема 2.13 (*теорема Лагранжа*). *Порядок конечной группы* делится на порядок любой ее подгруппы.

◀ Согласно теореме 2.11, все левые смежные классы образуют разбиение множества G на подмножества, равномощные в силу теоремы 2.12 подгруппе H. Так как группа $\mathcal{G}$ конечна, то число элементов разбиения конечно. Обозначив это число через k, заключаем, что $|G| = k|H|$. Следовательно, порядок группы $|G|$ делится на порядок группы $|H|$. ▶

Число всех левых смежных классов подгруппы $\mathcal{H}$ конечной группы $\mathcal{G}$ называют ***левым индексом подгруппы $\mathcal{H}$ в группе $\mathcal{G}$***.

Рассмотрим некоторые следствия теоремы Лагранжа.

Следствие 2.3. Любая *группа* простого порядка является *циклической*.

◀ Возьмем в группе, порядок которой есть простое число, какую-то ее циклическую подгруппу, *образующий элемент* которой отличен от *единицы* (нейтрального элемента) группы. Тогда эта подгруппа содержит не менее двух элементов и ее порядок, согласно теореме Лагранжа, должен быть делителем порядка группы. Поскольку порядок всей группы — простое число, а порядок подгруппы не меньше 2, то он совпадет с порядком всей группы. ▶

Замечание. Группа, порядок которой не является простым числом, может быть циклической, т.е. утверждение, обратное следствию 2.3, не имеет места. Так, например, циклической является $\mathbb{Z}_4^+$ — аддитивная группа вычетов по модулю 4. Ее образующий элемент — 1. Можно доказать, например, что любая группа порядка 15 является циклической*. #

Группу называют ***неразложимой***, если она не имеет *нетривиальных подгрупп*.

*См.: *Каргополов М.И., Мерзляков Ю.И.*

Следствие 2.4. Конечная группа неразложима тогда и только тогда, когда она является циклической группой, порядок которой есть простое число.

◀ Если группа циклическая и ее порядок — простое число, то, согласно теореме Лагранжа, каждая ее подгруппа имеет порядок, равный либо единице, либо порядку всей группы, и группа неразложима.

Обратно, пусть конечная группа $\mathcal{G} = (G, \cdot, \mathbf{1})$ неразложима. Покажем, что $|G|$ — простое число. Выберем элемент $a \neq \mathbf{1}$. Тогда циклическая подгруппа с образующим элементом a совпадает с $\mathcal{G}$. Допустим, что $|G|$ — составное число, т.е. $|G| = kl$ для некоторых натуральных k и l, отличных от 1 и $|G|$. Тогда циклическая подгруппа с образующим элементом $b = a^k$ не совпадает с $\mathcal{G}$, так как $b^l = a^{kl} = \mathbf{1}$ и в этой подгруппе не более l элементов, что противоречит неразложимости группы $\mathcal{G}$. Следовательно, порядок группы $\mathcal{G}$ есть простое число. ▶

Следствие 2.5. В конечной группе $\mathcal{G}$ для любого элемента $a \in G$ имеет место равенство $a^{|G|} = 1$.

◀ Если группа $\mathcal{G}$ циклическая и элемент a — ее образующий элемент, утверждение очевидно. Если же элемент a является образующим элементом некоторой циклической подгруппы группы $\mathcal{G}$ порядка $k < |G|$, то в силу теоремы Лагранжа $|G| = kl$ для некоторого натурального l. Отсюда получаем $a^{|G|} = a^{kl} = (a^k)^l = \mathbf{1}^l = \mathbf{1}$. ▶

С помощью теоремы Лагранжа (точнее, следствия 2.5) можно доказать, что если целое число n не делится на простое число p, то $n^{p-1} - 1$ делится на p. В теории чисел это утверждение известно как ***малая теорема Ферма***.

Действительно, пусть $n = rp + k$, где r — целое, а $0 < k < p$ (остаток от деления n на p). Тогда ясно, что $n^{p-1} = k^{p-1} \pmod p$ (достаточно разложить $(rp + k)^{p-1}$ по формуле *бинома Ньютона*). Рассмотрим группу $\mathbb{Z}_p^*$ (*мультипликативную группу*

вычетов по модулю p) и в этой группе элемент k. Порядок группы $\mathbb{Z}_p^* = p-1$. Если $k=1$, то

$$n^{p-1} - 1 = (1^{p-1} - 1) \pmod p = 0 \pmod p$$

и утверждение очевидно. Согласно следствию 2.5, в группе $\mathbb{Z}_p^*$ справедливо равенство $k^{p-1} = 1$, т.е. $k^{p-1} = 1 \pmod p$, и, следовательно, $k^{p-1} - 1 = 0 \pmod p$, т.е. число k^{p-1} равно 1 по модулю p. Поэтому $n^{p-1} = k^{p-1} = 1 \pmod p$.

Малая теорема Ферма дает возможность доказывать утверждения о делимости очень больших чисел. Например, из нее следует, что при $p = 97$ число 97 является делителем $n^{96} - 1$ для любого n, не делящегося на 97. Подобного рода заключения важны при разработке алгоритмов защиты информации.

Кроме того, используя малую теорему Ферма, можно вычислять в *полях вычетов* по *модулю* p (p — простое число) элементы, обратные к заданным относительно умножения. Действительно, если $a \in \mathbb{Z}_p$, то, так как $a^{p-1} = 1$, умножая последнее равенство на a^{-1}, получим $a^{p-2} = a^{-1}$. Таким образом, для того чтобы вычислить элемент, обратный к a по умножению, достаточно возвести его в степень $p-2$ или, что равносильно, в степень, равную остатку от деления числа $p-2$ на порядок циклической подгруппы группы $\mathbb{Z}_p^*$, порожденной элементом a (см. теорему 2.7).

Пример 2.20. Рассмотрим, как вычислить элемент, обратный к a по умножению в поле $\mathbb{Z}_{17}$. Согласно полученному выше результату, для вычисления обратного к a элемента нужно найти $a^{17-2} = a^{15}$. Однако объем вычислений можно сократить, если порядок циклической подгруппы, порожденной элементом a, меньше порядка группы.

Порядок группы $\mathbb{Z}_{17}^*$ равен 16, следовательно, порядок циклической подгруппы, порожденной элементом a, может составлять, согласно теореме Лагранжа, 2, 4, 8, 16 (т.е. быть каким-то из делителей числа 16). Поэтому при поиске обратного элемента достаточно проверить следующие степени a (кроме

15-й): 1 (остаток от деления 15 на 2), 3 (остаток от деления 15 на 4) и 7 (остаток от деления 15 на 8).

Найдем элемент, обратный к 2. Очевидно, что $2^{-1} \neq 2$, так как $2 \odot_{17} 2 = 4 \neq 1$. Далее получим $2^3 = 4 \odot_{17} 2 = 8$. Поскольку $2 \odot_{17} 8 = 16 \neq 1$, то $2^3 = 8$ также не является обратным к 2. Вычислим $2^7 = 2^3 \odot_{17} 2^3 \odot_{17} 2 = 8 \odot_{17} 8 \odot_{17} 2 = 9$. Поскольку $9 \odot_{17} 2 = 1$, в итоге получаем $2^{-1} = 9$.

Найдем элемент, обратный к 14. Так как $14 \odot_{17} 14 = 9$, то $14^{-1} \neq 14$. Вычисляем $14^3 = 14 \odot_{17} 9 = 7$, но $14 \odot_{17} 7 = 13$, т.е. $14^3 \neq 14^{-1}$. Далее,

$$14^7 = 14^3 \odot_{17} 14^4 = 7 \odot_{17} 13 = 6,$$
$$14 \odot_{17} 6 = 16 = -1.$$

Мы видим, что и $14^7 \neq 14^{-1}$. Следовательно, остается вычислить $14^{-1} = 14^{15}$. Однако в этом случае вычисления можно сократить, заметив, что $14 \odot_{17} 14^7 = 14 \odot_{17} 6 = -1$. Из последнего равенства, согласно следствию 2.1, получим

$$1 = 14 \odot_{17} (-6) = 14 \odot_{17} 11,$$

откуда $14^{-1} = 11$.

Отметим, что $14^{16} = 1$, т.е. порядок циклической подгруппы, порожденной элементом 14, совпадает с порядком всей группы $\mathbb{Z}_{17}^*$, и, следовательно, эта группа является циклической, порожденной элементом 14 (хотя и не только им).

2.8. Гомоморфизмы групп и нормальные делители

Пусть заданы *группы* $\mathcal{G}_1 = (G_1, \cdot, \mathbf{1})$ и $\mathcal{G}_2 = (G_2, \cdot, \mathbf{1})$. Отображение $f\colon G_1 \to G_2$ называют гомоморфизмом группы $\mathcal{G}_1$ в группу $\mathcal{G}_2$ (***гомоморфизмом групп***), если для любых $x, y \in G_1$ выполняется равенство $f(x \cdot y) = f(x) \cdot f(y)$, т.е. *образ* произведения любых двух элементов группы $\mathcal{G}_1$ *при отображении* f равен произведению их образов в группе $\mathcal{G}_2$.

Если отображение f *сюръективно* (*биективно*), то его называют ***эпиморфизмом*** (***изоморфизмом***) ***групп***. В этом случае говорят также об эпиморфизме (изоморфизме) группы $\mathcal{G}_1$ на группу $\mathcal{G}_2$.

Замечание 2.5. Мы обозначили *операции* групп $\mathcal{G}_1$ и $\mathcal{G}_2$ одинаково, как это обычно и делается для *однотипных алгебр*, хотя, конечно, это разные операции разных групп.

Пример 2.21. Пусть $\mathcal{G}_1 = (\mathbb{Z}, +, 0)$ — *аддитивная группа целых чисел*, а $\mathcal{G}_2 = \mathbb{Z}_k^+$ — *аддитивная группа вычетов по модулю* k.

Зададим отображение f так: для всякого целого m образ $f(m)$ равен остатку от деления m на k. Можно проверить, что для любых целых m и n имеет место равенство $f(m+n) = f(m) \oplus_k f(n)$, т.е. для целых чисел остаток от деления суммы на k равен сумме по модулю k остатков от деления на k каждого слагаемого.

Следовательно, данное отображение есть гомоморфизм группы $\mathcal{G}_1$ в группу $\mathcal{G}_2$. Далее, поскольку любое целое число от 0 до $k-1$ есть остаток от деления на k какого-то числа, то отображение f является и эпиморфизмом группы $\mathcal{G}_1$ на группу $\mathcal{G}_2$.

Теорема 2.14. Пусть $\mathcal{G}_1$, $\mathcal{G}_2$ — произвольные группы. Если $f\colon \mathcal{G}_1 \to \mathcal{G}_2$ — гомоморфизм, то:

1) образом *единицы* (*нейтрального элемента*) *группы* $\mathcal{G}_1$ при отображении f является единица группы $\mathcal{G}_2$, т.е. $f(\mathbf{1}) = \mathbf{1}$;

2) для всякого элемента x группы $\mathcal{G}_1$ образом элемента x^{-1} является элемент $[f(x)]^{-1}$, обратный элементу $f(x)$, т.е. $f(x^{-1}) = [f(x)]^{-1}$.

◀ Согласно определению гомоморфизма, для произвольного $x \in G_1$ имеем $f(x) \cdot f(\mathbf{1}) = f(x \cdot \mathbf{1})$. Далее, $f(x \cdot \mathbf{1}) = f(x)$, т.е. $f(x) \cdot f(\mathbf{1}) = f(x)$. Следовательно, $f(\mathbf{1}) = (f(x))^{-1} \cdot f(x) = \mathbf{1}$, т.е. $f(\mathbf{1}) = \mathbf{1}$.

Докажем второе утверждение теоремы. Используя определение гомоморфизма и уже доказанное первое утверждение

теоремы, получаем

$$f(x^{-1}) \cdot f(x) = f(x^{-1} \cdot x) = f(1) = 1,$$

т.е. $f(x^{-1}) = [f(x)]^{-1}$. ▶

Множество $f(G_1)$ — образ носителя группы $\mathcal{G}_1$ при гомоморфизме f — замкнуто относительно умножения группы $\mathcal{G}_2$. Действительно, если g_2, $g_2' \in f(\mathcal{G}_1)$, то существуют такие $g_1, g_1' \in \mathcal{G}_1$, что $f(g_1) = g_2$ и $f(g_1') = g_2'$. Тогда

$$g_2 g_2' = f(g_1) f(g_1') = f(g_1 g_1') \in f(\mathcal{G}_1).$$

Из теоремы 2.14 следует, что $f(\mathcal{G}_1)$ содержит единицу этой группы и вместе с каждым элементом обратный к нему элемент. Это значит, что можно определить подгруппу группы $\mathcal{G}_2$, носителем которой будет множество $f(G_1)$. Эту группу называют гомоморфным образом группы $\mathcal{G}_1$ при гомоморфизме f.

Группу $\mathcal{K}$ называют просто ***гомоморфным образом группы*** $\mathcal{G}$, если существует гомоморфизм группы $\mathcal{G}$ на группу $\mathcal{K}$. Так, группа $\mathbb{Z}_k^*$ при любом $k > 1$ является гомоморфным образом аддитивной группы целых чисел (см. пример 2.21).

Обратимся к следующему примеру.

Пример 2.22. Рассмотрим мультипликативную группу $(\mathbb{C} \setminus \{0\}, \cdot, 1)$ комплексных чисел с обычной операцией умножения комплексных чисел. Легко понять, что эта группа не что иное, как *мультипликативная группа поля комплексных чисел.*

Рассмотрим также группу $\mathcal{M}_2$ невырожденных квадратных матриц второго порядка с операцией умножения матриц (см. пример 2.9.е).

Определим отображение f множества $\mathbb{C}$ комплексных чисел в множество квадратных матриц второго порядка, положив для произвольного ненулевого комплексного числа $a + bi$, что

$$f(a + bi) = \begin{pmatrix} a & b \\ -b & a \end{pmatrix}.$$

Покажем, что f — гомоморфизм групп. С одной стороны,

$$f[(a+bi)(c+di)] = f[(ac-bd)+i(ad+bc)] = \\ = \begin{pmatrix} ac-bd & ad+bc \\ -ad-bc & ac-bd \end{pmatrix}.$$

С другой стороны,

$$f(a+bi)f(c+di) = \begin{pmatrix} a & b \\ -b & a \end{pmatrix}\begin{pmatrix} c & d \\ -d & c \end{pmatrix} = \\ = \begin{pmatrix} ac-bd & ad+bc \\ -ad-bc & ac-bd \end{pmatrix}.$$

Следовательно,

$$f[(a+bi)(c+di)] = f(a+bi)f(c+di).$$

Таким образом, отображение f — гомоморфизм групп, а гомоморфный образ мультипликативной группы комплексных чисел при f — это подгруппа $\mathcal{K}$ группы матриц $\mathcal{M}_2$, состоящая из матриц вида $\begin{pmatrix} a & b \\ -b & a \end{pmatrix}$. Здесь мы учли, что любая матрица вида $\begin{pmatrix} a & b \\ -b & a \end{pmatrix}$ является образом некоторого комплексного числа (а именно $a+bi$) при отображении f. Группа $\mathcal{K}$ — *собственная подгруппа* группы $\mathcal{M}_2$. #

Сформулируем без доказательства одно важное свойство гомоморфизмов групп.

Теорема 2.15. Если f — гомоморфизм группы $\mathcal{G}$ в группу $\mathcal{K}$, а g — гомоморфизм группы $\mathcal{K}$ в группу $\mathcal{L}$, то *композиция отображений* $f \circ g$ есть гомоморфизм группы $\mathcal{G}$ в группу $\mathcal{L}$. #

Рассмотрим некоторые свойства изоморфизмов групп.

Теорема 2.16. Если $f\colon \mathcal{G}_1 \to \mathcal{G}_2$ — изоморфизм группы $\mathcal{G}_1$ на группу $\mathcal{G}_2$, то отображение f^{-1}, обратное к отображению f, есть изоморфизм группы $\mathcal{G}_2$ на группу $\mathcal{G}_1$.

◀ Пусть x и y — произвольные элементы группы $\mathcal{G}_2$, пусть также $x = f(u)$, а $y = f(v)$, где u и v — элементы группы $\mathcal{G}_1$. Тогда

$$f^{-1}(xy) = f^{-1}(f(u)f(v)) = f^{-1}(f(uv)) = uv = f^{-1}(x)f^{-1}(y),$$

т.е. отображение f^{-1} — гомоморфизм второй группы в первую. Но так как отображение, обратное к биекции, есть биекция, то f^{-1} — изоморфизм группы $\mathcal{G}_2$ на группу $\mathcal{G}_1$. ▶

Группы $\mathcal{G}$ и $\mathcal{K}$ называют ***изоморфными***, если существует изоморфизм одной из них на другую. При этом используют обозначение $\mathcal{G} \cong \mathcal{K}$.

Изоморфные группы с точки зрения их алгебраических свойств совершенно одинаковы, хотя их элементы могут иметь различную природу. Вернемся в этой связи к примеру 2.22. Легко убедиться в том, что определенное там отображение f множества комплексных чисел на множество квадратных матриц специального вида является биекцией. Следовательно, мультипликативная группа комплексных чисел и группа матриц указанного вида с операцией умножения матриц изоморфны, хотя элементы этих групп на первый взгляд не имеют между собой ничего общего.

Определение 2.8. ***Ядром гомоморфизма*** f группы $\mathcal{G}$ в группу $\mathcal{K}$ называют *прообраз* $\operatorname{Ker} f$ единицы группы $\mathcal{G}$ при гомоморфизме f: $\operatorname{Ker} f = f^{-1}(1) \subseteq G$.

Пример 2.23. Ядром гомоморфизма, рассмотренного в примере 2.21, служит множество всех целых чисел, делящихся на k.

Теорема 2.17. Ядро $\operatorname{Ker} f$ гомоморфизма $f\colon \mathcal{G} \to \mathcal{K}$ есть подгруппа группы $\mathcal{G}$.

◀ Нужно убедиться в том, что множество $\operatorname{Ker} f$ замкнуто относительно умножения группы $\mathcal{G}$, содержит единицу этой группы и вместе с каждым элементом содержит обратный к нему элемент.

Если a, $b \in \operatorname{Ker} f$, т.е. $f(a) = f(b) = \mathbf{1}$, то $f(ab) = f(a)f(b) = \mathbf{1}$ и $ab \in \operatorname{Ker} f$. Ясно, что $\mathbf{1} \in \operatorname{Ker} f$, так как $f(\mathbf{1}) = \mathbf{1}$ (см. теорему 2.14). Если $a \in \operatorname{Ker} f$, то $f(a^{-1}) = [f(a)]^{-1} = \mathbf{1}^{-1} = \mathbf{1}$, т.е. и $a^{-1} \in \operatorname{Ker} f$. ▶

Ядро гомоморфизма, приведенного в примере 2.21, представляет собой подгруппу аддитивной группы целых чисел, состоящую из всех чисел, кратных k.

Подгруппа $\mathcal{H}$ группы $\mathcal{G}$ называется ***нормальной подгруппой*** (***нормальным делителем***) группы $\mathcal{G}$, если $aH = Ha$ для любого $a \in G$.

В *коммутативной группе*, как было отмечено выше, $aH = Ha$. Следовательно, в этом случае любая подгруппа является нормальным делителем.

Пусть $\mathcal{H} = (H, \cdot, \mathbf{1})$ — подгруппа группы $\mathcal{G} = (G, \cdot, \mathbf{1})$. Для фиксированных элементов a, $b \in G$ через aHb обозначим множество всех произведений вида ahb, где $h \in H$. В силу ассоциативности групповой операции это обозначение корректно.

Теорема 2.18. Подгруппа $\mathcal{H} = (H, \cdot, \mathbf{1})$ является нормальным делителем группы $\mathcal{G} = (G, \cdot, \mathbf{1})$ тогда и только тогда, когда $aHa^{-1} \subseteq H$ для любого $a \in G$.

◀ Если $\mathcal{H}$ — нормальный делитель, то для любого $a \in G$ $aH = Ha$, т.е. для любого $h \in H$ найдется такое $h_1 \in H$, что $ah = h_1a$. Пусть элемент $x \in aHa^{-1}$, т.е. $x = aha^{-1}$ для некоторого $h \in H$. Так как $ah = h_1a$, то $x = h_1aa^{-1} = h_1 \in H$ и поэтому $aHa^{-1} \subseteq H$.

Обратно, если $aHa^{-1} \subseteq H$, то любой элемент $x = aha^{-1}$, где $h \in H$, принадлежит и множеству H, т.е. $aha^{-1} = h_1$ для некоторого $h_1 \in H$. Отсюда, умножая последнее равенство на a справа, получим $ah = h_1a$, т.е. элемент ah из левого смежного класса aH принадлежит и правому смежному классу Ha. Итак, $aH \subseteq Ha$.

Теперь возьмем для произвольного $a \in G$ обратный к a элемент a^{-1} и для него запишем включение $a^{-1}Ha \subseteq H$ (напомним,

что $(a^{-1})^{-1} = a$). Рассуждая как и выше, получим, что для некоторых h, $h_1 \in H$ имеет место равенство $a^{-1}h = h_1 a^{-1}$, т.е. $ha = ah_1$ и $Ha \subseteq aH$. Итак, $aH = Ha$ и $\mathcal{H}$ — нормальный делитель. ▶

Оказывается, существует связь между понятием нормального делителя и понятием гомоморфизма, которая продолжает и углубляет на новом уровне уже известную нам из главы 1 связь между понятиями отображения и *класса эквивалентности*.

Теорема 2.19. Ядро гомоморфизма f группы $\mathcal{G}$ в группу $\mathcal{K}$ является нормальным делителем группы $\mathcal{G}$.

◀ Для любого $y \in \operatorname{Ker} f$ и любого $a \in G$ имеем

$$f(aya^{-1}) = f(a)f(y)f(a^{-1}) = f(a) \cdot \mathbf{0} \cdot f(a^{-1}) = f(a)f(a^{-1}) = \mathbf{1}.$$

Это значит, что для любого $a \in G$ выполняется соотношение $a(\operatorname{Ker} f)a^{-1} \subseteq \operatorname{Ker} f$, а, согласно теореме 2.18, $\operatorname{Ker} f$ — нормальный делитель. ▶

Пусть $\mathcal{H} = (H, \cdot, \mathbf{1})$ — нормальный делитель группы $\mathcal{G} = (G, \cdot, \mathbf{1})$. Рассмотрим множество всех *левых смежных классов* $\{aH: a \in G\}$. Это будет не что иное, как *фактор-множество* множества G по определенному выше (см. теорему 2.11) *отношению эквивалентности* $\sim_H$.

Введем операцию умножения на множестве всех левых смежных классов следующим образом: произведением $aH \cdot bH$ классов aH и bH назовем класс abH.

Это определение корректно, так как множество $aH \cdot bH$, т.е. множество всех произведений вида $ahbh_1$ для различных h, $h_1 \in H$, в силу того что $Hb = bH$ для всякого $b \in G$, совпадает с левым смежным классом abH. Действительно, поскольку $hb = bh'$ для некоторого $h' \in H$, то $ahbh_1 = abh'h_1 \in abH$.

Теперь рассмотрим некоторый $x \in abH$, т.е. $x = abh$ для некоторого $h \in H_1$. Поскольку $bh = h'b$ для некоторого $h' \in H$, то $x = ah'b = ah'b\mathbf{1} \in aHbH$. Следовательно, $aH \cdot bH = abH$.

Можно далее легко показать, что для каждого $a \in G$ имеют место $aH \cdot H = H \cdot aH = aH$ и $aH \cdot a^{-1}H = a^{-1}H \cdot aH = H$. Тем самым определена группа, носителем которой является фактор-множество $G/\sim_H$ множества G по отношению эквивалентности $\sim_H$ с операцией умножения левых смежных классов, причем *нейтральным элементом* относительно этой операции служит носитель подгруппы $\mathcal{H}$, а обратным к левому смежному классу aH будет левый смежный класс $a^{-1}H$. Эту группу называют ***фактор-группой*** группы $\mathcal{G}$ ***по нормальному делителю*** $\mathcal{H}$ и обозначают $\mathcal{G}/\mathcal{H}$. Можно указать естественный гомоморфизм f группы $\mathcal{G}$ в фактор-группу $\mathcal{G}/\mathcal{H}$, который вводится согласно правилу: $(\forall x \in G)(f(x) = xH)$. Так как $xH \cdot yH = xyH$, то для любых $x, y \in G$ $f(xy) = xyH = xH \cdot yH = f(x)f(y)$ и f — действительно гомоморфизм. Его называют ***каноническим гомоморфизмом группы*** $\mathcal{G}$ ***в фактор-группу*** $\mathcal{G}/\mathcal{H}$.

Пример 2.24. а. Рассмотрим ***аддитивную группу*** $\mathbb{R} = (\mathbb{R}, +, 0)$ ***действительных чисел***. Эта группа коммутативна. Напомним, что в коммутативной группе любая подгруппа будет нормальным делителем. Поэтому для нее нормальным делителем является подгруппа целых чисел $\mathbb{Z} = (\mathbb{Z}, +, 0)$ (*аддитивная группа целых чисел*). (Для этих групп мы приняли такие же обозначения, как и для их носителей: $\mathbb{R}$ и $\mathbb{Z}$ соответственно.)

Выясним смысл отношения эквивалентности $\sim_{\mathbb{Z}}$, определяемого через равенство левых смежных классов*, по подгруппе $\mathbb{Z}$ в этом случае.

Равенство левых смежных классов $a + \mathbb{Z} = b + \mathbb{Z}$ означает, что для любого целого m найдется такое целое n, что $a + m = b + n$, т.е. $a - b = n - m \in \mathbb{Z}$. Обратно, если разность $a - b$ есть целое число, т.е. $a - b = n \in \mathbb{Z}$, то $a + \mathbb{Z} = (b + n) + \mathbb{Z} = b + \mathbb{Z}$. Итак, $a \sim_{\mathbb{Z}} b$ тогда и только тогда, когда $a - b \in \mathbb{Z}$, или, иначе

*Мы можем говорить в данном случае просто о смежных классах, не различая левых и правых, так как для нормального делителя эти классы равны, тем более что мы „работаем“ сейчас в коммутативной группе.

говоря, действительные числа a и b $\sim_{\mathbb{Z}}$-эквивалентны тогда и только тогда, когда их дробные части равны.

Аддитивная группа смежных классов, т.е. фактор-группа $\mathbb{R}/\mathbb{Z}$ группы $\mathbb{R}$ по нормальному делителю $\mathbb{Z}$ строится так: сумма классов $a+\mathbb{Z}$ и $b+\mathbb{Z}$ равна классу $(a+b)+\mathbb{Z}$. Вводя обозначение $a+\mathbb{Z}=[a]$, получаем $[a]+[b]=[a+b]$. При этом $[0]=\mathbb{Z}$ (т.е. единица фактор-группы — это смежный класс нуля — множество всех целых чисел), причем $-[a]=[-a]=(-a)+\mathbb{Z}$. Обратим внимание на то, что смежный класс числа x однозначно определяется его дробной частью $<x>$ (см. пример 1.14.б), т.е. $[x]=[<x>]$. Канонический гомоморфизм в данном случае задается так: $x\mapsto[x]$.

б. Рассмотрим теперь ***аддитивную группу действительных чисел по модулю 1***, т.е. группу $\mathbf{S}^1=([0,1),\oplus_1,0)$, заданную на полуинтервале $[0,1)$, сложение в которой определяется так: $x\oplus_1 y=<x+y>$ (дробная часть суммы $x+y$). Другими словами,

$$x\oplus_1 y=\begin{cases} x+y, & x+y<1;\\ x+y-1, & x+y\geqslant 1.\end{cases}$$

Докажем, что группа $\mathbf{S}^1$ изоморфна фактор-группе $\mathbb{R}/\mathbb{Z}$, т.е. $\mathbb{R}/\mathbb{Z}\cong\mathbf{S}^1$.

Зададим отображение φ множества $\{[a]\colon a\in\mathbb{R}\}$ смежных классов в полуинтервал $[0,1)$ так, что $\varphi([x])=<x>$. Поскольку $[x]=[<x>]$, то φ — биекция и, кроме того,

$$\varphi([x]+[y])=\varphi([x+y])=<x+y>=<<x>+<y>>=$$
$$=<x>\oplus_1<y>=\varphi([x])\oplus_1\varphi([y]).$$

Это значит, что φ — изоморфизм $\mathbb{R}/\mathbb{Z}$ на $\mathbf{S}^1$.

Группу $\mathbf{S}^1$ можно воспринимать как „наглядный образ“ фактор-группы $\mathbb{R}/\mathbb{Z}$. Довольно абстрактная идея фактор-группы кристаллизуется в виде группы с носителем $[0,1)$ и операцией сложения неотрицательных действительных чисел, строго

меньших единицы, с отбрасыванием в результате целой части. Здесь хорошо видна „польза“ понятия изоморфизма. То, что само по себе не очень наглядно, становится наглядным через свой изоморфный образ.

2.9. Гомоморфизмы колец

Рассмотрим очень коротко вопрос о гомоморфизмах *колец* и *полей*.

Пусть $\mathcal{R}_1 = (R_1, +, \cdot, \mathbf{0}, \mathbf{1})$ и $\mathcal{R}_2 = (R_2, +, \cdot, \mathbf{0}, \mathbf{1})$ — кольца.

Определение 2.9. *Отображение* $f\colon R_1 \to R_2$ называют ***гомоморфизмом колец*** (кольца $\mathcal{R}_1$ в кольцо $\mathcal{R}_2$), если $f(x+y) = f(x) + f(y)$, $f(x \cdot y) = f(x) \cdot f(y)$ для любых $x, y \in R_1$, т.е. *образ* суммы и произведения любых двух элементов кольца $\mathcal{R}_1$ *при отображении* f равен соответственно сумме и произведению их образов в кольце $\mathcal{R}_2$.

Если отображение f *сюръективно* (соответственно *биективно*), то его называют ***эпиморфизмом*** (соответственно ***изоморфизмом***) ***колец*** (кольца $\mathcal{R}_1$ на кольцо $\mathcal{R}_2$).

Пример 2.25. Рассмотрим $\mathcal{R}_1 = (\mathbb{Z}, +, \cdot, 0, 1)$ — *кольцо целых чисел* — и $\mathbb{Z}_k = (\mathbb{Z}_k, \oplus_k, \odot_k, 0, 1)$ — *кольцо вычетов по модулю* k. Зададим отображение $f\colon \mathbb{Z} \to \mathbb{Z}_k$ так: для всякого целого m образ $f(m)$ равен остатку от деления m на k. Ранее мы уже доказали (см. пример 2.21), что для любых целых m и n имеет место равенство $f(m+n) = f(m) \oplus_k f(n)$. Рассуждая аналогично, можно показать, что для любых целых m и n также верно равенство $f(m \cdot n) = f(m) \odot_k f(n)$. С учетом того что отображение f сюръективно, приходим к выводу, что оно является гомоморфизмом кольца целых чисел на кольцо $\mathbb{Z}_k$ вычетов по модулю k. #

Без доказательства сформулируем некоторые теоремы о гомоморфизмах и изоморфизмах колец (и полей). Все эти утверждения могут быть доказаны по аналогии с соответствующими теоремами о *гомоморфизмах* и *изоморфизмах групп*.

Теорема 2.20. Пусть $\mathcal{R}_1$ и $\mathcal{R}_2$ — произвольные кольца. Если $f\colon \mathcal{R}_1 \to \mathcal{R}_2$ — гомоморфизм, то

1) образ *нуля кольца* $\mathcal{R}_1$ при отображении f есть нуль кольца $\mathcal{R}_2$, т.е. $f(\mathbf{0}) = \mathbf{0}$;

2) образ *единицы кольца* $\mathcal{R}_1$ при отображении f есть единица кольца $\mathcal{R}_2$, т.е. $f(\mathbf{1}) = \mathbf{1}$;

3) для всякого элемента x кольца $\mathcal{R}_1$ образ элемента, *противоположного* элементу x, равен элементу, противоположному образу элемента x, т.е. $f(-x) = -f(x)$;

4) если кольца $\mathcal{R}_1$ и $\mathcal{R}_2$ являются полями, то для всякого элемента x кольца $\mathcal{R}_1$ образ элемента, *обратного* к элементу x *по умножению*, равен элементу, обратному к образу элемента x, т.е. $f(x^{-1}) = [f(x)]^{-1}$.

Теорема 2.21. Если f — гомоморфизм кольца $\mathcal{R}$ в кольцо $\mathcal{K}$, а g — гомоморфизм кольца $\mathcal{K}$ в кольцо $\mathcal{L}$, то композиция отображений $f \circ g$ есть гомоморфизм кольца $\mathcal{R}$ в кольцо $\mathcal{L}$.

Теорема 2.22. Если $f\colon \mathcal{R}_1 \to \mathcal{R}_2$ — изоморфизм кольца $\mathcal{R}_1$ на кольцо $\mathcal{R}_2$, то отображение f^{-1} есть изоморфизм кольца $\mathcal{R}_2$ на кольцо $\mathcal{R}_1$. #

Как и в случае групп, определяются понятия гомоморфного образа кольца и изоморфных колец. А именно кольцо $\mathcal{K}$ называют ***гомоморфным образом кольца*** $\mathcal{R}$, если существует гомоморфизм кольца $\mathcal{R}$ на кольцо $\mathcal{K}$. Два кольца $\mathcal{R}$ и $\mathcal{K}$ называют ***изоморфными*** и пишут $\mathcal{R} \cong \mathcal{K}$, если существует изоморфизм одного из них на другой.

Так, например, кольцо вычетов по модулю k есть гомоморфный образ кольца целых чисел при гомоморфизме, задаваемом отображением, которое каждому целому m сопоставляет остаток от деления m на k.

Рассмотрим один интересный пример изоморфизма полей.

Пример 2.26. Так же как и в примере 2.22, поставим в соответствие комплексному числу $a + bi$ матрицу $f(a + bi) = \begin{pmatrix} a & b \\ -b & a \end{pmatrix}$. Получим отображение f, которое, как уже было

доказано, является инъекцией, причем $f(0) = f(0+0\cdot i) = \mathbf{0}$, где $\mathbf{0}$ — нулевая матрица. Заметим, что, поскольку определитель матрицы указанного вида равен $a^2 + b^2$, среди всех таких матриц только нулевая будет иметь нулевой определитель.

Далее, легко проверить, что множество таких матриц *замкнуто относительно операций* сложения и умножения матриц, содержит (как уже было отмечено) нулевую и единичную матрицы, а также вместе с каждой матрицей A матрицу $-A$ и вместе с каждой ненулевой матрицей обратную к ней матрицу. Это значит, что множество матриц вида $\begin{pmatrix} a & b \\ -b & a \end{pmatrix}$, $a, b \in \mathbb{R}$, с операциями сложения и умножения матриц образует поле. Обозначим его $\mathcal{M}_2^{(a,b)}$.

Из примера 2.22 следует, что мультипликативная группа поля комплексных чисел изоморфна мультипликативной группе поля $\mathcal{M}_2^{(a,b)}$. Так как

$$f[(a+bi)+(c+di)] = f[(a+c)+(b+d)i] =$$
$$= \begin{pmatrix} a+c & b+d \\ -b-d & a+c \end{pmatrix} = \begin{pmatrix} a & b \\ -b & a \end{pmatrix} + \begin{pmatrix} c & d \\ -d & c \end{pmatrix} =$$
$$= f(a+bi) + f(c+di),$$

то и аддитивная группа поля комплексных чисел изоморфна аддитивной группе поля $\mathcal{M}_2^{(a,b)}$. Итак, мы получаем, что поле комплексных чисел изоморфно полю матриц $\mathcal{M}_2^{(a,b)}$. Этот изоморфизм лежит в основе матричного представления алгебры комплексных чисел, что имеет значение для компьютерных реализаций этой алгебры.

Дополнение 2.1. Кватернионы

Примером *тела*, не являющегося *полем*, может служить ***алгебра кватернионов*** („четырехмерных чисел“).

Сначала напомним, как строится алгебра *комплексных чисел*. Комплексное число определяется как *упорядоченная пара*

действительных чисел, причем сложение пар вводится покомпонентно:

$$(a,\ b)+(c,\ d)=(a+c,\ b+d),$$

а умножение — согласно формуле

$$(a,\ b)\cdot(c,\ d)=(ac-bd,\ ad+bc). \tag{2.3}$$

Заметим, что покомпонентное определение умножения не дало бы требуемых свойств этой операции (таких же, как у умножения действительных чисел), поскольку при покомпонентном умножении имеются *делители нуля*:

$$(a,\ 0)\cdot(0,\ b)=(a\cdot 0,\ 0\cdot b)=(0,\ 0).$$

При стандартном определении умножения согласно (2.3) можно доказать, что алгебра комплексных чисел является полем.

Кватернионы определяются как упорядоченные пары комплексных чисел. Сложение кватернионов вводится снова как покомпонентное, а умножение — следующим образом:

$$(a,\ b)\cdot(c,\ d)=\left(ac-b\overline{d},\ ad+b\overline{c}\right),$$

где $\overline{x}$ — *число, комплексно сопряженное* с x.

Таблица 2.3

$\cdot$	1	i	j	k
1	1	i	j	k
i	i	-1	k	$-j$
j	j	$-k$	-1	i
k	k	j	$-i$	-1

Можно показать, что каждый кватернион однозначно представляется в виде

$$\alpha=x+iy+jz+kh,$$

где x, y, z, h — действительные числа, а i, j и k — кватернионы, называемые ***мнимыми единицами***. Правила умножения мнимых единиц вместе с обычной единицей задаются табл. 2.3.

Можно заметить, что умножение попарно различных мнимых единиц совершается по тем же правилам, что и векторное умножение ортов $\boldsymbol{i}$, $\boldsymbol{j}$, $\boldsymbol{k}$ в векторной алгебре [III].

Сопряженный с $\alpha = x + iy + jz + kh$ ***кватернион***, по определению, равен $\overline{\alpha} = x - iy - jz - kh$. Имеют место следующие тождества:

$$\overline{\alpha\beta} = \overline{\alpha}\,\overline{\beta},$$
$$\overline{(\alpha+\beta)} = \overline{\alpha} + \overline{\beta},$$
$$\alpha\overline{\alpha} = \overline{\alpha}\alpha = a^2 + b^2 + c^2 + d^2.$$

Действительное число $\alpha\overline{\alpha}$, обозначаемое $n(\alpha)$, называется ***нормой кватерниона*** α.

Построенная таким образом алгебра кватернионов обладает следующими свойствами:

1. Умножение кватернионов *ассоциативно*, но не ***коммутативно***.

Это свойство непосредственно выводится из табл. 2.3. Например, пусть $\alpha = 0 + i1 + j0 + k0$, а $\beta = 0 + i0 + j1 + k0$. Тогда $\alpha\beta = 0 + i0 + j0 + k$, а $\beta\alpha = 0 + i0 + j0 + k(-1)$.

2. В алгебре кватернионов нет делителей нуля.

Действительно, $n(\alpha) = 0$ тогда и только тогда, когда $\alpha = 0$, т.е. нулевому кватерниону $0 + i0 + j0 + k0$, и можно показать, что $n(\alpha\beta) = n(\alpha)n(\beta)$.

Тогда, если $\alpha \neq 0$, то

$$\alpha \cdot ((1/n(\alpha)) \cdot \overline{\alpha}) = ((1/n(\alpha)) \cdot \overline{\alpha}) \cdot \alpha = 1.$$

Таким образом, по умножению все ненулевые кватернионы образуют группу и алгебра кватернионов оказывается телом, не будучи полем.

Из кватернионов можно также строить упорядоченные пары, вводя операции сложения и умножения как описано выше, а операцию сопряжения согласно формуле $\overline{(a,b)} = (a, -\overline{b})$ (a и b — кватернионы). Полученная таким образом алгебра называется ***алгеброй Кэли***, а ее элементы — ***числами Кэли*** или ***октавами***. Умножение в алгебре Кэли уже не будет даже

ассоциативным, однако ассоциативный закон имеет место в частном случае — для любых двух чисел Кэли α и β:

$$(\alpha\alpha)\beta = \alpha(\alpha\beta),$$
$$\alpha(\beta\beta) = (\alpha\beta)\beta,$$
$$\alpha(\beta\alpha) = (\alpha\beta)\alpha.$$

Кроме того, каждое ненулевое число Кэли имеет обратное. Другими словами, в алгебре Кэли любое уравнение $\alpha x = \beta$ (или $x\alpha = \beta$) имеет единственное решение. Таким образом, алгебра Кэли также является алгеброй без делителей нуля. Можно доказать, что, кроме алгебры комплексных чисел, алгебры кватернионов и алгебры Кэли, не существует „многомерных" числовых алгебр без делителей нуля.

Вопросы и задачи

2.1. Ассоциативна ли операция $\odot$ на множестве M, если:

а) $M = \mathbb{N}$, $x \odot y = 2xy$;
б) $M = \mathbb{Z}$, $x \odot y = x^2 + y^2$;
в) $M = \mathbb{R}$, $x \odot y = \sin x \sin y$;
г) $M = \mathbb{R} \setminus \{0\}$, $x \odot y = xy^{x/|x|}$;
д) $M = \mathbb{R}$, $x \odot y = x - y$?

2.2. Пусть S — полугруппа матриц вида $\begin{pmatrix} x & y \\ 0 & 0 \end{pmatrix}$, где $x, y \in \in \mathbb{R}$, с операцией умножения. Существуют ли в ней левые или правые нейтральные элементы?

2.3. На множестве M определена бинарная операция $\circ$ по правилу $x \circ y = x$. Доказать, что $(M, \circ)$ — полугруппа. Что можно сказать о нейтральных элементах этой полугруппы? В каких случаях она является группой?

2.4. Пусть M — произвольное множество. На множестве M^2 определена операция $\circ$ по правилу $(x, y) \circ (z, t) = (x, t)$.

Является ли алгебра $(M^2, \circ)$ полугруппой? Существует ли в ней нейтральный элемент?

2.5. Привести пример полугруппы с левой единицей (нейтральным элементом), не являющейся моноидом.

2.6. Какие из указанных множеств квадратных действительных матриц одного порядка образуют группу:

а) множество невырожденных матриц относительно умножения;

б) множество невырожденных матриц относительно сложения;

в) множество диагональных матриц относительно сложения;

г) множество диагональных матриц относительно умножения?

2.7. Доказать, что если в группе $\mathcal{G}$ для любого $x \in G$ выполняется тождество $x^2 = 1$, то группа $\mathcal{G}$ коммутативна.

2.8. В группе S_4 решить уравнения:

а) $\begin{pmatrix} 1 & 2 & 3 & 4 \\ 4 & 2 & 1 & 3 \end{pmatrix} X \begin{pmatrix} 1 & 2 & 3 & 4 \\ 3 & 2 & 1 & 4 \end{pmatrix} = (1\,2)$;

б) $(1\,2)(3\,4)X(1\,3) = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 4 & 2 & 1 & 3 \end{pmatrix}$.

2.9. Является ли полем множество чисел вида $x + \sqrt{2}y$, где $x, y \in \mathbb{Q}$, с обычными операциями сложения и умножения?

2.10. Доказать, что множество всех верхних треугольных матриц фиксированного порядка n является подкольцом кольца всех квадратных матриц порядка n. Верно ли это утверждение для диагональных и нижних треугольных матриц?

2.11. Построить пример кольца с одним элементом, т.е. такого, в котором $\mathbf{0} = \mathbf{1}$.

2.12. Кольцо $\mathcal{R} = (R, +, \cdot, \mathbf{0}, \mathbf{1})$ называется ***булевым***, если его умножение идемпотентно, т.е. $x \cdot x = x$ для любых $x \in R$. Доказать, что:

а) для любого элемента x булева кольца имеет место равенство $x + x = \mathbf{0}$, т.е. $-x = x$;

б) любое булево кольцо коммутативно;

в) в любом булевом кольце, имеющем более двух элементов ($|R| > 2$), существуют делители нуля.

2.13. Доказать, что $\left(2^M, \Delta, \cap, \varnothing, M\right)$ — булево кольцо (см. задачу 2.12). Доказать, что оно изоморфно $\mathbb{Z}_2$ при $|M| = 1$.

2.14. Показать, что множество остатков от деления многочленов от переменной x на $x^2 + x + 1$ с операциями сложения и умножения многочленов является кольцом. Является ли это кольцо полем?

2.15. Элемент x кольца называют ***обратимым***, если существует элемент x', такой, что $x \cdot x' = x' \cdot x = \mathbf{1}$. Элемент x кольца называют обратимым слева (справа), если существует x', такой, что $x' \cdot x = \mathbf{1}$ ($x \cdot x' = \mathbf{1}$). Элемент кольца называется односторонне обратимым, если он обратим слева или справа.

Элемент $x \neq \mathbf{0}$ кольца называется левым (правым) делителем нуля, если существует ненулевой элемент кольца y, такой, что $x \cdot y = \mathbf{0}$ ($y \cdot x = 0$); элемент, являющийся левым и правым делителем нуля одновременно, называется делителем нуля.

Доказать, что:

а) элемент конечного кольца обратим (слева, справа) тогда и только тогда, когда он не является делителем нуля (правым, левым);

б) в конечном кольце и в кольце без делителей нуля любой односторонне обратимый элемент обратим;

в) элемент кольца вычетов по модулю k обратим тогда и только тогда, когда он взаимно прост с k.

2.16. Пусть R — кольцо. Доказать, что:

а) если произведения xy и yx обратимы, то элементы x и y тоже обратимы;

б) если в R нет делителей нуля и произведение xy обратимо, то x и y обратимы;

в) если R конечно и произведение xy обратимо, то x и y обратимы.

Указание: использовать результаты задачи 2.15.

2.17. Доказать, что множество всех обратимых элементов кольца (см. задачу 2.15) образуют группу по умножению.

2.18. Решить систему уравнений

$$\begin{cases} x+2y=1, \\ y+2z=2, \\ 2x+z=1: \end{cases}$$

а) в поле $\mathbb{Z}_3$; б) в поле $\mathbb{Z}_5$.

2.19. Выяснить, разрешима ли в кольце $\mathbb{Z}_{21}$ система уравнений

$$\begin{cases} 5x+2y=1, \\ y-11x=13. \end{cases}$$

2.20. Введем группу $\mathbb{S}$ „движений“ (поворотов) окружности как группу, элементами которой являются всевозможные повороты, измеряемые в радианах, причем поворот на любой угол, кратный 2π, отождествляется с нулевым поворотом (тождественным отображением множества точек окружности в себя).

Доказать, что группа $\mathbb{S}$ изоморфна фактор-группе $\mathbb{R}/\mathbb{Z}$, которая, в свою очередь, изоморфна аддитивной группе $\mathbf{S}^1$ действительных чисел по модулю 1.

2.21. Пусть $\mathcal{M}=(G,\ +,\ \mathbf{0},\ \{\omega_\alpha\colon \alpha\in R\})$ — левый модуль над кольцом $\mathcal{R}=(R,\ +,\ \cdot,\ 0,\ 1)$. Является ли соответствие между элементами α кольца $\mathcal{R}$ и отображениями ω_α взаимно однозначным? В частности, всегда ли нулевое отображение определяется только нулем кольца $\mathcal{R}$?

Указание: рассмотреть левый модуль матриц-столбцов вида $(x\ \ 0)^{\mathrm{T}}$, $x\in R$, над кольцом верхнетреугольных матриц второго порядка и показать, что ответы на поставленные вопросы отрицательны.

3. ПОЛУКОЛЬЦА И БУЛЕВЫ АЛГЕБРЫ

Полукольцо является одной из важнейших *алгебр* в современной дискретной математике. Эта глава посвящена рассмотрению *полуколец* и *булевых алгебр*. Изучаемые здесь методы, прежде всего метод решения систем линейных уравнений в полукольцах, имеют первостепенное значение для теории *графов*, *булевых функций* и теории формальных языков.

3.1. Полукольца. Основные примеры

Определение 3.1. ***Полукольцо*** — это алгебра с двумя *бинарными* и двумя *нульарными операциями*

$$S = (S, +, \cdot, \mathbf{0}, \mathbf{1}),$$

такая, что для произвольных элементов a, b, c множества S выполняются следующие равенства, называемые ***аксиомами полукольца***:

1) $a + (b + c) = (a + b) + c$;
2) $a + b = b + a$;
3) $a + \mathbf{0} = a$;
4) $a \cdot (b \cdot c) = (a \cdot b) \cdot c$;
5) $a \cdot \mathbf{1} = \mathbf{1} \cdot a = a$;
6) $a \cdot (b + c) = a \cdot b + a \cdot c$;
7) $(b + c) \cdot a = b \cdot a + c \cdot a$;
8) $a \cdot \mathbf{0} = \mathbf{0} \cdot a = \mathbf{0}$.

Первую операцию $+$ называют ***сложением полукольца***, а вторую операцию $\cdot$ — ***умножением полукольца*** S; элементы $\mathbf{0}$ и $\mathbf{1}$ называют соответственно ***нулем*** и ***единицей полукольца*** S.

Аксиомы полукольца называют также ***основными тождествами полукольца.***

Таким образом, из определения следует, что операция сложения полукольца *ассоциативна* и *коммутативна*, а нуль полукольца является *нейтральным элементом* относительно операции сложения; операция умножения полукольца ассоциативна и единица полукольца является нейтральным элементом относительно операции умножения. Кроме этого имеет место свойство *дистрибутивности* (двусторонней) операции умножения относительно сложения полукольца. Аксиому 8 полукольца называют ***аннулирующим свойством нуля*** в полукольце.

Используя понятие *моноида*, определение 3.1 можно переформулировать так. Полукольцо $\mathcal{S} = (S, +, \cdot, \mathbf{0}, \mathbf{1})$ — это алгебра с двумя бинарными и двумя нульарными операциями, такая, что:

1) алгебра $(S, +, \mathbf{0})$ является коммутативным моноидом (его называют ***аддитивным моноидом полукольца***);

2) алгебра $(S, \cdot, \mathbf{1})$ является моноидом (его называют ***мультипликативным моноидом полукольца***);

3) имеют место свойства (двусторонней) дистрибутивности операции сложения относительно операции умножения;

4) выполняется аннулирующее свойство нуля.

Сравнивая определение полукольца с определением 2.5 *кольца*, мы видим, что кольцо есть частный случай полукольца: если кольцо по сложению является *абелевой группой*, то полукольцо — лишь коммутативный моноид.

Выделим два вида полуколец: ***коммутативное полукольцо*** с коммутативной операцией умножения и ***идемпотентное полукольцо*** с *идемпотентной* операцией сложения.

Пример 3.1. Рассмотрим алгебру

$$\mathcal{R}^+ = \big(\mathbb{R}^+ \cup \{+\infty\},\ \min,\ +,\ +\infty,\ 0\big),$$

где $\mathbb{R}^+$ — множество неотрицательных действительных чисел, min — операция взятия наименьшего из двух данных чисел,

$+$ — операция сложения действительных чисел, $+\infty$ — „плюс бесконечность“ (в том же смысле, что и в математическом анализе), 0 — число „нуль“.

Эта алгебра — полукольцо, носителем которого является полуось $\mathbb{R}^+ = \{x\colon x \geqslant 0\}$, пополненная элементом $+\infty$ (множество всех неотрицательных действительных чисел вместе с „плюс бесконечностью“).

Необычность полукольца $\mathcal{R}^+$ состоит в том, что в качестве его умножения взято сложение действительных чисел, тогда как в качестве сложения выбрана операция взятия наименьшего из двух чисел. Согласно *сигнатуре*, элемент $+\infty$ рассматривается как нуль полукольца. Действительно, $\min(x, +\infty) = x$ для любого $x \in \mathbb{R}^+ \cup \{+\infty\}$, т.е. элемент $+\infty$ есть нейтральный элемент относительно операции $\min$ (операции сложения в полукольце). Элемент $+\infty$ также обладает аннулирующим свойством относительно операции сложения чисел (операции умножения в полукольце): $x + (+\infty) = +\infty$. Следовательно, выполняются аксиомы 3 и 8 полукольца.

Остальные аксиомы полукольца также выполнены.

Легко убедиться, что алгебра $\bigl(\mathbb{R}^+ \cup \{+\infty\},\ \min,\ +\infty\bigr)$ — коммутативный моноид и алгебра $\bigl(\mathbb{R}^+ \cup \{+\infty\},\ +,\ 0\bigr)$ — также коммутативный моноид. Проверим свойства дистрибутивности, которые в данном случае запишутся так:

$$a + \min(b,\ c) = \min(a + b,\ a + c).$$

Имеем

$$a + \min(b,\ c) = \begin{cases} a + b, & b \leqslant c; \\ a + c, & b > c. \end{cases}$$

В то же время

$$\min(a + b,\ a + c) = \begin{cases} a + b, & b \leqslant c; \\ a + c, & b > c. \end{cases}$$

Таким образом,

$$a + \min(b,\ c) = \min(a + b,\ a + c).$$

Заметим, что в рассматриваемом полукольце умножение $+$ коммутативно, а сложение $\min$ идемпотентно. Следовательно, $\mathcal{R}^+$ — идемпотентное коммутативное полукольцо.

Пример 3.2. Рассмотрим алгебру $\mathcal{B} = (\{0,1\}, +, \cdot, 0, 1)$, в которой операции $+$ и $\cdot$ заданы *таблицами Кэли* (табл. 3.1 и 3.2).

Таблица 3.1

$+$	0	1
0	0	1
1	1	1

Таблица 3.2

$\cdot$	0	1
0	0	0
1	0	1

Проверка аксиом полукольца основана на этих таблицах и легко выполняется. Обратим внимание, что два элемента 0 и 1, из которых в данном случае состоит носитель полукольца, одновременно являются соответственно нулем и единицей данного полукольца. Легко видеть, что рассматриваемое полукольцо коммутативное и идемпотентное.

Интересно то, что операции полукольца $\mathcal{B}$ можно трактовать как логические связки „или“ и „и“, а элементы 0 и 1 — как „ложь“ и „истина“ соответственно.

Пример 3.3. Рассмотрим еще несколько различных алгебр, являющихся полукольцами. Выполнение аксиом полукольца для всех приведенных алгебр легко проверяется.

а. Алгебра $\mathcal{N} = (\mathbb{N}_0, +, \cdot, 0, 1)$ с носителем — множеством неотрицательных целых чисел и операциями сложения и умножения чисел есть коммутативное полукольцо. Оно не является идемпотентным.

б. Алгебра $S_A = \left(2^A, \cup, \cap, \varnothing, A\right)$ с носителем — множеством всех подмножеств некоторого множества A и операциями объединения и пересечения есть полукольцо. Оно является идемпотентным и коммутативным.

в. Алгебра $\mathcal{R}_A = \left(2^{A\times A}, \cup, \circ, \varnothing, id_A\right)$ с носителем — множеством всех *бинарных отношений* на множестве A — и операциями *объединения* и *композиции отношений* является полукольцом. Оно идемпотентное, но не коммутативное.

г. Алгебра $\mathcal{S}_{[a,b]} = ([a, b], \max, \min, a, b)$, носителем которой служит отрезок числовой прямой, с операциями взятия максимума и минимума из двух чисел есть идемпотентное и коммутативное полукольцо. #

В дальнейшем нас будут интересовать только идемпотентные полукольца, поскольку именно на их основе разрабатываются алгебраические методы анализа *ориентированных графов* и *конечных автоматов*.

На носителе идемпотентного полукольца $\mathcal{S} = (S, +, \cdot, \mathbf{0}, \mathbf{1})$ может быть введено *отношение порядка*, которое, естественно, согласуется со свойствами операций полукольца: для произвольных $x, y \in S$ положим $x \leqslant y$ тогда и только тогда, когда $x + y = y$, т.е.

$$x \leqslant y \Leftrightarrow x + y = y. \tag{3.1}$$

Проверим, что таким образом действительно определено отношение порядка. Для этого нужно показать, что введенное бинарное отношение *рефлексивно*, *антисимметрично* и *транзитивно*.

Поскольку для любого x в силу идемпотентности сложения имеет место равенство $x + x = x$, то, согласно (3.1), имеем $x \leqslant x$, т.е. введенное отношение рефлексивно.

Соотношения $x \leqslant y$ и $y \leqslant x$ означают, что $x + y = y$ и $y + x = x$. Поскольку сложение коммутативно, то из этих равенств следует, что $x = y$. Значит, рассматриваемое отношение антисимметрично.

Соотношения $x \leqslant y$ и $y \leqslant z$ означают, что $x + y = y$ и $y + z = z$. Тогда

$$x + z = x + (y + z) = (x + y) + z = y + z = z,$$

откуда следует, что $x \leqslant z$. Таким образом, введенное отношение транзитивно.

Итак, отношение $\leqslant$ на носителе произвольного идемпотентного полукольца есть отношение порядка. Будем называть его ***естественным порядком идемпотентного полукольца*** и говорить, что он задан в этом полукольце.

Мы установили очень важный факт: всякое идемпотентное полукольцо можно рассматривать как *упорядоченное множество*, причем отношение порядка определяется через сложение этого полукольца согласно (3.1). Введенное отношение порядка можно интерпретировать так: „большее при сложении поглощает меньшее“ (как, скажем, при объединении множества и некоторого его подмножества).

Поскольку для любого элемента x произвольного идемпотентного полукольца $\mathcal{S} = (S, +, \cdot, \mathbf{0}, \mathbf{1})$ имеет место $\mathbf{0} + x = x$, то для любого $x \in S$ выполняется неравенство $\mathbf{0} \leqslant x$, т.е. нуль идемпотентного полукольца есть *наименьший элемент* относительно естественного порядка идемпотентного полукольца.

Объясним, как интерпретируется естественный порядок идемпотентных полуколец, рассмотренных в приведенных выше примерах.

Пример 3.4. В полукольце $\mathcal{B}$ (см. пример 3.2) выполняется равенство $0 + 1 = 1$ и, следовательно, $0 \leqslant 1$.

В полукольце $\mathcal{R}^+$ (см. пример 3.1) $x \leqslant y$, если и только если $\min(x, y) = y$.

Обозначим через $\leqslant_{\mathbb{R}}$ *естественный числовой порядок* на множестве действительных чисел. Тогда для произвольных элементов x, y полукольца $\mathcal{R}^+$ соотношение $x \leqslant y$ означает, что $x \geqslant_{\mathbb{R}} y$, т.е. число x не меньше числа y относительно естественного числового порядка. Таким образом, порядок в полукольце $\mathcal{R}^+$ — это *двойственный порядок* для отношения $\leqslant_{\mathbb{R}}$. В полукольце есть наименьший элемент относительно введенного порядка — элемент $+\infty$, поскольку для любого элемента x имеем $\min(x, +\infty) = x$. Существует и *наибольший элемент* — единица полукольца, т.е. число 0. Не следует путать число 0 с нулем данного полукольца, а именно с элементом $+\infty$.

В полукольце $\mathcal{S}_A$ (см. пример 3.3.б) получаем в качестве отношения естественного порядка полукольца *отношение включения* $\subseteq$. Действительно, для любых двух множеств $X, Y \in 2^A$ из $X \cup Y = Y$ вытекает $X \subseteq Y$ и наоборот. Наименьшим элементом является нуль полукольца — $\varnothing$ (пустое множество), а наибольшим — единица полукольца (множество A).

В полукольце $\mathcal{R}_A$ (см. пример 3.3.в) отношение естественного порядка полукольца также совпадает с отношением включения для бинарных отношений. В этом полукольце существуют наименьший элемент — пустое отношение $\varnothing$ и наибольший элемент — *универсальное отношение*. Однако в отличие от полукольца $\mathcal{S}_A$ наибольший элемент не совпадает с единицей полукольца.

В полукольце $\mathcal{S}_{[a,b]}$ (см. пример 3.3.г) имеем естественный числовой порядок, определенный на множестве действительных чисел отрезка $[a, b]$. Наименьшим элементом является число a (нуль полукольца), а наибольшим — число b (единица полукольца).

Теорема 3.1. Если A — конечное подмножество (носителя) идемпотентного полукольца, то $\sup A$ относительно естественного порядка этого полукольца равен сумме всех элементов множества A.

◀ Пусть $A = \{a_1, \ldots, a_n\}$ и $a = a_1 + \ldots + a_n$. Для произвольного элемента a_i, $i = \overline{1, n}$, в силу коммутативности и идемпотентности сложения имеем

$$\begin{aligned} a_i + a = a_i + (a_1 + \ldots + a_i + \ldots + a_n) = \\ = a_1 + \ldots + a_i + a_i + \ldots + a_n = \\ = a_1 + \ldots + a_i + \ldots + a_n = a, \end{aligned}$$

т.е. $a_i \leqslant a$, и поэтому a есть *верхняя грань множества* A. Покажем, что это *точная верхняя грань множества*. Возьмем произвольную верхнюю грань b множества A и рассмотрим

сумму $b+a$. Так как для каждого $i=\overline{1,n}$ имеет место $a_i \leqslant b$, т.е. $a_i+b=b$, то

$$b+a=b+(a_1+a_2+\ldots+a_n)=$$
$$=(b+a_1)+(a_2+\ldots+a_n)=b+a_2+\ldots+a_n=\ldots=b.$$

Следовательно, $a \leqslant b$ и поэтому a — точная верхняя грань множества A. ▶

3.2. Замкнутые полукольца

При изучении *колец* большое внимание было уделено *полям*. Связано это прежде всего с тем, что в полях разработана техника решения систем линейных уравнений. Оказывается, можно выделить специальный класс *идемпотентных полуколец*, в которых также удается находить решения систем уравнений, рассматриваемых в качестве аналогов систем линейных алгебраических уравнений [III].

Определение 3.2. *Полукольцо* $S=(S,+,\cdot,\mathbf{0},\mathbf{1})$ называют ***замкнутым***, если:

1) оно идемпотентно;

2) любая последовательность элементов множества S имеет *точную верхнюю грань* относительно *естественного порядка* $\leqslant$ этого *идемпотентного полукольца*;

3) операция *умножения полукольца* S сохраняет *точные верхние грани последовательностей*, т.е. для любого $a \in S$ и любой последовательности $X=\{x_n\}_{n\in\mathbb{N}}$ элементов множества S

$$a\sup X=\sup aX, \quad (\sup X)a=\sup(Xa).$$

Замечание 3.1. Условие 3 определения 3.2 можно сформулировать иначе и говорить о сохранении точной верхней грани любого не более чем счетного подмножества множества S. Однако в дальнейшем такие подмножества часто будут строиться как множества элементов некоторых последовательностей.

Кроме того, из элементов не более чем счетного множества всегда можно образовать последовательность, используя некоторую *нумерацию* этого множества.

Теорема 3.2. Любое конечное идемпотентное полукольцо замкнуто.

◀ Поскольку носитель S идемпотентного полукольца

$$\mathcal{S} = (S, +, \cdot, \mathbf{0}, \mathbf{1})$$

есть конечное множество, то множество элементов любой последовательности в этом полукольце конечно. Для нахождения точной верхней грани такой последовательности нужно найти *точную верхнюю грань множества* $P = \{p_1, \ldots, p_n\}$ ее членов, т.е., согласно теореме 3.1, вычислить некоторую конечную сумму, которая всегда существует. Таким образом, в конечном идемпотентном полукольце любая последовательность имеет точную верхнюю грань.

Условия сохранения точных верхних граней имеют вид

$$a(p_1 + \ldots + p_n) = ap_1 + \ldots + ap_n,$$
$$(p_1 + \ldots + p_n)a = p_1a + \ldots + p_na$$

и выполняются в силу *аксиом полукольца*.

Таким образом, полукольцо $\mathcal{S}$ замкнуто. ▶

Мы доказали (см. **3.1**), что в любом идемпотентном полукольце сумма произвольного конечного множества элементов является его точной верхней гранью.

В силу этого в замкнутом полукольце естественно точную верхнюю грань последовательности $\{x_n\}_{n\in\mathbb{N}}$ называть ***суммой элементов последовательности***, полагая, по определению,

$$\sum_{n=1}^{\infty} x_n = \sup\{x_n \colon n \in \mathbb{N}\}. \tag{3.2}$$

Согласно условию 2 определения 3.2, всегда $\sum\limits_{n=1}^{\infty} x_n$ есть элемент множества S. Иногда, если это не приводит к недоразумению, „пределы суммирования" будем опускать и писать просто $\sum x_n$. Также будем использовать обозначение $\sum\limits_{n\in\mathbb{N}} x_n$. Подчеркивая, что в (3.2) множество элементов x_n в общем случае бесконечно, будем сумму, стоящую в левой части (3.2), называть ***бесконечной суммой***. Заметим, что в частных случаях бесконечная сумма может свестись к конечной (если множество всех элементов последовательности $\{x_n\}$ конечно).

Итак, согласно определению 3.2, замкнутое полукольцо является *индуктивным упорядоченным множеством*, в котором *наименьшим элементом* служит *нуль полукольца*, точной верхней гранью произвольной (в частности, неубывающей) последовательности $\{x_n\}_{n\in\mathbb{N}}$ является бесконечная сумма $\sum x_n$, причем операция умножения на произвольный фиксированный элемент a *непрерывна* в смысле определения 1.5, поскольку, согласно определению 3.2, сохраняет точные верхние грани. Заметим, что это свойство умножения в замкнутом полукольце можно рассматривать как аналог *дистрибутивности* (седьмой аксиомы полукольца, см. определение 3.1) для бесконечной суммы:

$$a\sum x_n = \sum ax_n \quad \text{и} \quad \Big(\sum x_n\Big)a = \sum x_n a.$$

Для бесконечной суммы также справедливы аналоги свойств идемпотентности и коммутативности.

Действительно, для последовательности $\{x_n\}_{n\in\mathbb{N}}$, такой, что $x_n = a$ для любого $n \in \mathbb{N}$, имеем $\sum x_n = \sup\{a\} = a$, т.е. бесконечная сумма, все слагаемые которой одинаковы и равны некоторому a, равна a. В этом состоит свойство идемпотентности бесконечной суммы (или, как иногда говорят, свойство бесконечной идемпотентности).

Так как точная верхняя грань последовательности любого упорядоченного множества равна, по определению, точной

верхней грани множества всех элементов этой последовательности, то при любом расположении слагаемых в бесконечной сумме мы получим один и тот же результат. Это свойство есть свойство бесконечной коммутативности, или коммутативности бесконечной суммы.

Сложнее обстоит дело с аналогом ассоциативности для бесконечной суммы. Такая ассоциативность означает, что результат не изменяется при произвольной группировке слагаемых бесконечной суммы.

Для некоторой последовательности $\{x_n\}_{n\in\mathbf{N}}$ элементов замкнутого полукольца и произвольной последовательности $\{n_k\}_{k\in\mathbf{N}}$ натуральных чисел образуем последовательность $\{s_k\}_{k\in\mathbf{N}}$, где

$$
\begin{aligned}
&s_1 = x_1 + \ldots + x_{n_1},\\
&s_2 = x_{n_1+1} + \ldots + x_{n_1+n_2},\\
&\ldots\ldots\ldots\ldots\ldots\ldots\\
&s_k = x_{n_{k-1}+1} + \ldots + x_{n_{k-1}+n_k},\\
&\ldots\ldots\ldots\ldots\ldots\ldots
\end{aligned}
$$

Тогда бесконечная ассоциативность состоит в выполнении для любой последовательности $\{x_n\}_{n\in\mathbf{N}}$ равенства

$$\sum_{n\in\mathbf{N}} x_n = \sum_{k\in\mathbf{N}} s_k. \tag{3.3}$$

Свойство бесконечной ассоциативности (3.3) удобнее не доказывать непосредственно, а вывести из более общих результатов. Один из них касается характеристики замкнутых полуколец в терминах точных верхних граней счетных подмножеств.

Теорема 3.3. Идемпотентное полукольцо $\mathcal{S} = (S, +, \cdot, \mathbf{0}, \mathbf{1})$ замкнуто тогда и только тогда, когда любое счетное подмножество $X \subseteq S$ имеет точную верхнюю грань и для любого $a \in S$ верны равенства

$$a \cdot \sup X = \sup(a \cdot X), \quad (\sup X) \cdot a = \sup(X \cdot a).$$

◀ Справедливость утверждения теоремы немедленно вытекает из определения точной верхней грани последовательности элементов упорядоченного множества как точной верхней грани множества этих элементов. ▶

Теорема 3.4. Пусть $\mathcal{S} = (S, +, \cdot, \mathbf{0}, \mathbf{1})$ — замкнутое полукольцо, X — не более чем счетное подмножество множества $\mathcal{S}$, а $(B_i)_{i\in I}$ — не более чем счетное семейство подмножеств множества X, такое, что объединение семейства совпадает с X, т.е. $\bigcup\limits_{i\in I} B_i = X$. Тогда

$$\sup X = \sup\{\sup B_i, i \in I\}. \tag{3.4}$$

◀ Поскольку множества X и I не более чем счетны, то, согласно определению 3.2, все точные верхние грани из (3.4) существуют и принадлежат S. Обозначим левую часть равенства (3.4) через a, а правую — через b.

Так как a есть точная верхняя грань множества X, то для любого $x \in X$ справедливо неравенство $x \leqslant a$ (где $\leqslant$ — естественный порядок идемпотентного полукольца $\mathcal{S}$). В частности, для любого подмножества B_i и любого его элемента y получаем $y \leqslant a$, откуда следует, что элемент a есть верхняя грань каждого подмножества B_i. Значит, этот элемент не меньше чем точная верхняя грань каждого из этих подмножеств, т.е. для любого $i \in I$ $a \geqslant \sup B_i$. Последнее означает, что a есть верхняя грань множества всех точных верхних граней $\sup B_i$, $i \in I$, т.е. $a \geqslant b$.

В то же время, поскольку объединение всех подмножеств B_i равно X, для любого $x \in X$ найдется такое $i \in I$, что $x \in B_i$. Поэтому $x \leqslant \sup B_i \leqslant b$, это означает, что элемент b является верхней гранью множества X, и тогда $b \geqslant a$, так как a есть точная верхняя грань X.

Итак, $a \geqslant b$ и $b \geqslant a$, откуда $a = b$, и равенство (3.4) доказано. ▶

Следствие 3.1. Если семейство подмножеств из теоремы 3.4 $(B_i)_{i\in I}$ конечно, т.е. $I=\{1,2,\ldots,n\}$, то

$$\sup X=\sum_{i=1}^{n}\sup B_i.$$

Следствие 3.2. Пусть $\{x_n\}_{n\in\mathbb{N}}$ — произвольная последовательность элементов замкнутого полукольца и $(N_i)_{i\in I}$ — не более чем счетное семейство подмножеств N, такое, что $\bigcup\limits_{i\in I}N_i=\mathbb{N}$. Тогда

$$\sum_{n\in\mathbb{N}}x_n=\sum_{i\in I}s_i, \tag{3.5}$$

где $s_i=\sum\limits_{m\in N_i}x_m$. #

В частности, из следствия 3.2 при условии, что множества N_i, $i\in I$, попарно не пересекаются, получаем свойство ассоциативности бесконечной суммы (3.3). Отсюда же при $N_1=\{1\}$, $N_2=\{1,2\}$ и т.д., т.е. при определении для любого натурального k множества N_k в виде $N_k=N_{k-1}\cup\{k\}$, получаем равенство

$$\sum_{n\in\mathbb{N}}x_n=\sum_{k\in\mathbb{N}}(x_1+\ldots+x_k), \tag{3.6}$$

т.е. точная верхняя грань последовательности $\{x_n\}_{n\in\mathbb{N}}$ равна точной верхней грани последовательности ***частичных сумм*** $\{s_k\}_{k\in\mathbb{N}}$, где $s_k=x_1+\ldots+x_k$.

Пусть $\{x_n\}_{n\in\mathbb{N}}$ и $\{y_m\}_{m\in\mathbb{N}}$ — произвольные последовательности в замкнутом полукольце $\mathcal{S}$. Образуем последовательность, членами которой являются все произведения x_ny_m при независимом пробегании индексами n и m множества натуральных чисел*. Эту последовательность будем записывать

*Такую последовательность называют произведением Коши исходных последовательностей [IX].

как $\{x_n y_m\}_{n,m\in\mathbf{N}}$, а ее точную верхнюю грань обозначим как $\sum\limits_{n,m\in\mathbf{N}} x_n y_m$.

Теорема 3.5. В любом замкнутом полукольце верно тождество

$$\sum_{n,m\in\mathbf{N}} x_n y_m = \Big(\sum_{n\in\mathbf{N}} x_n\Big)\Big(\sum_{m\in\mathbf{N}} y_m\Big). \tag{3.7}$$

◀ Ввиду непрерывности умножения в замкнутом полукольце правую часть тождества (3.7) можно переписать в виде

$$\Big(\sum_{n\in\mathbf{N}} x_n\Big)\Big(\sum_{m\in\mathbf{N}} y_m\Big) = \sum_{m\in\mathbf{N}}\Big(\sum_{n\in\mathbf{N}} x_n\Big) y_m.$$

Еще раз используя непрерывность умножения и внося сомножитель y_m под знак внутренней суммы, получаем

$$\sum_{m\in\mathbf{N}}\Big(\sum_{n\in\mathbf{N}} x_n\Big) y_m = \sum_{m\in\mathbf{N}}\Big(\sum_{n\in\mathbf{N}} x_n y_m\Big).$$

В правой части последнего равенства каждое слагаемое внешней суммы (по m) есть точная верхняя грань подпоследовательности $\{x_n y_m\}_{n\in\mathbf{N}}$ при фиксированном m. Поскольку все эти подпоследовательности в совокупности дают всю последовательность $\{x_n y_m\}_{n,m\in\mathbf{N}}$, то, согласно следствию 3.2, получаем

$$\sum_{m\in\mathbf{N}}\Big(\sum_{n\in\mathbf{N}} x_n y_m\Big) = \sum_{n,m\in\mathbf{N}} x_n y_m,$$

что и доказывает тождество (3.7). ▶

Следующая теорема устанавливает связь между конечной и бесконечной суммами.

Пусть $\{x_n\}_{n\in\mathbf{N}}$ и $\{y_n\}_{n\in\mathbf{N}}$ — произвольные последовательности в замкнутом полукольце $\mathcal{S}$. Образуем последовательность $\{x_n + y_n\}_{n\in\mathbf{N}}$, называемую суммой исходных последовательностей.

Теорема 3.6. В любом замкнутом полукольце точная верхняя грань суммы двух произвольных последовательностей равна сумме точных верхних граней этих последовательностей, т.е.

$$\sum(x_n + y_n) = \sum x_n + \sum y_n. \tag{3.8}$$

◀ Обозначим через X множество всех членов последовательности $\{x_n\}_{n\in\mathbb{N}}$, а через Y — множество всех членов последовательности $\{y_n\}_{n\in\mathbb{N}}$. В силу следствия 3.2

$$\sup(X \cup Y) = \sup X + \sup Y = \sum x_n + \sum y_n.$$

Осталось тогда доказать, что

$$\sum(x_n + y_n) = \sup(X \cup Y). \tag{3.9}$$

Обозначим через a левую, а через b правую часть доказываемого равенства (3.9). Для любого натурального n имеем $a \geqslant x_n + y_n$. Согласно теореме 3.1, $x_n + y_n \geqslant x_n$ и $x_n + y_n \geqslant y_n$. Следовательно, для любого n выполняются неравенства $a \geqslant x_n$ и $a \geqslant y_n$, т.е. $a \geqslant u$ для любого $u \in X \cup Y$. Таким образом, элемент a является верхней гранью множества $X \cup Y$, откуда $a \geqslant b$.

В то же время элемент b не меньше любого из элементов множества $X \cup Y$, и, стало быть, для любого натурального n имеет место $b \geqslant x_n$ и $b \geqslant y_n$, т.е. $b \geqslant \sup\{x_n, y_n\} = x_n + y_n$. Это значит, что b есть верхняя грань последовательности $\{x_n + y_n\}_{n\in\mathbb{N}}$, а потому $b \geqslant a$.

Итак, $a = b$, и равенство (3.9) доказано. ▶

Если в (3.8) $y_n = a$ для всех n, то получаем следствие.

Следствие 3.3. Для любой последовательности $\{x_n\}_{n\in\mathbb{N}}$ элементов замкнутого полукольца и любого элемента a этого полукольца выполняется равенство

$$a + \sum x_n = \sum(a + x_n). \quad \# \tag{3.10}$$

Тождество (3.10) можно рассматривать как свойство непрерывности операции сложения в замкнутом полукольце. Это свойство совершенно аналогично свойству непрерывности операции умножения, которое имеет место по определению.

Одним из важнейших понятий в замкнутых полукольцах является понятие ***итерации*** (или ***замыкания***) ***элемента*** замкнутого полукольца. Итерация x^* элемента x определяется как точная верхняя грань последовательности всех *степеней элемента* x, т.е.

$$x^* = \sum_{n=0}^{\infty} x^n,$$

где, по определению, $x^0 = 1$, а $x^n = x^{n-1}x$, $n = 1, 2, \ldots$

Пример 3.5. **а.** Из теоремы 3.2 сразу получаем, что идемпотентное полукольцо $\mathcal{B}$ (см. пример 3.2) замкнуто, причем точная верхняя грань любой последовательности элементов этого полукольца равна 1, если хотя бы один ее член равен 1, и равна 0 в противном случае. В частности, итерация любого элемента полукольца $\mathcal{B}$ равна 1. Для 1^* это очевидно, а для 0^* имеем

$$0^* = 0^0 + 0^1 + \ldots + 0^k + \ldots = 1 + 0 + \ldots + 0 + \ldots = 1.$$

б. В идемпотентном полукольце $\mathcal{R}^+$ (см. пример 3.1) любая последовательность есть последовательность неотрицательных действительных чисел. Такая последовательность ограничена снизу и, как известно из курса математического анализа, имеет точную нижнюю грань относительно естественного числового порядка [I], которая, согласно примеру 3.4, представляет собой точную верхнюю грань относительно естественного порядка идемпотентного полукольца $\mathcal{R}^+$. Напомним, что этот порядок является *двойственным* к естественному числовому порядку.

Итак, для любой последовательности x_n элементов полукольца $\mathcal{R}^+$ точная верхняя грань существует. Непрерывность операции умножения в этом полукольце также можно доказать,

опираясь на естественный числовой порядок, для которого она эквивалентна выполнению равенства

$$a + \inf X = \inf X_a, \tag{3.11}$$

где a — неотрицательное действительное число, а X (соответственно X_a) — множество элементов последовательности x_n (соответственно $x_n + a$). Равенство (3.11) следует из известных результатов математического анализа.

Таким образом, рассматриваемое идемпотентное полукольцо $\mathcal{R}^+$ является замкнутым.

Итерация x^* элемента x в полукольце $\mathcal{R}^+$ есть точная верхняя грань последовательности степеней элемента x. Поскольку в этом полукольце операция умножения определена как операция сложения действительных чисел, то $x^0 = 0$, так как число 0 есть единица полукольца $\mathcal{R}^+$. Далее, $x^2 = x + x = 2x$, ..., $x^n = nx$. Очевидно, что для каждого $n \geqslant 0$ выполняется неравенство $x^n \geqslant 0$ в смысле естественного числового порядка. Таким образом, число 0 есть наименьший в смысле естественного числового порядка член последовательности $\{x^n\}_{n\in\mathbf{N}}$ и, следовательно, $\inf\{x^n\}_{n\in\mathbf{N}} = 0$. Переходя к двойственному порядку — естественному порядку полукольца $\mathcal{R}^+$, получим, что число 0 является точной верхней гранью последовательности $\{x^n\}_{n\in\mathbf{N}}$, т.е. $x^* = 0$. Таким образом, в полукольце $\mathcal{R}^+$ итерация произвольного элемента также равна единице полукольца, т.е. числу 0.

в. Замкнутость идемпотентного полукольца $\mathcal{S}_A$ (см. пример 3.3.б) можно обосновать следующим образом. Отношение естественного порядка этого полукольца — это *отношение включения* (см. пример 3.4). Рассмотрим произвольную последовательность подмножеств B_1, ..., B_n, ... множества A. В данном полукольце бесконечная сумма есть объединение последовательности подмножеств множества A.

Докажем, что объединение $B = \bigcup\limits_{n\in\mathbf{N}} B_n$ и есть $\sup B_n$. Очевидно, что для каждого i справедливо включение $B_i \subseteq B$, т.е.

объединение есть верхняя грань последовательности $\{B_n\}_{n\in N}$. Покажем, что B будет точной верхней гранью. Рассмотрим произвольную верхнюю грань C последовательности B_n. Тогда $B_n \subseteq C$ для каждого $n \in \mathbb{N}$ (см. **1.5**) и поэтому

$$C \cup B = C \cup \Big(\bigcup_{n\in\mathbb{N}} B_n\Big) = \bigcup_{n\in\mathbb{N}} (C \cup B_n) = C,$$

т.е. $B \subseteq C$. Следовательно, $B = \bigcup\limits_{n\in\mathbb{N}} B_n = \sup B_n$.

Непрерывность умножения полукольца $\mathcal{S}_A$, в качестве которого взято пересечение множеств, эквивалентна тождеству

$$C \cap \Big(\bigcup_{n\in\mathbb{N}} B_n\Big) = \bigcup_{n\in\mathbb{N}} (C \cup B_n)$$

(дистрибутивности пересечения относительно произвольного объединения, см. **1.5**). В этом полукольце умножение — это пересечение множеств. Поэтому любая положительная степень множества X есть само X. Нулевая же степень X^0 равна единице полукольца, т.е. множеству A. Поэтому итерация X^* равна

$$X^* = X^0 \cup X \cup \ldots \cup X^n \cup \ldots = X^0 \cup X = A \cup X = A,$$

т.е. также равна единице полукольца.

г. Рассмотрим идемпотентное полукольцо $\mathcal{R}_A$ (см. пример 3.3.в) *бинарных отношений на множестве* A. Можно доказать, что точной верхней гранью любой последовательности отношений, как и в полукольце $\mathcal{S}_A$, служит объединение элементов этой последовательности (см. пример 3.4).

Аналогично доказательству дистрибутивности композиции бинарных отношений на множестве относительно конечного объединения (см. **1.4**) можно доказать, что для любых бинар-

ных отношений ρ и σ_n на множестве A справедливы тождества

$$\rho \circ \bigcup_{n \geqslant 1} \sigma_n = \bigcup_{n \geqslant 1} (\rho \circ \sigma_n),$$

$$\Big(\bigcup_{n \geqslant 1} \sigma_n\Big) \circ \rho = \bigcup_{n \geqslant 1} (\sigma \circ \rho).$$

Таким образом, идемпотентное полукольцо $\mathcal{R}_A$ замкнутое.

Итерация бинарного отношения ρ есть

$$\rho^* = \bigcup_{n \geqslant 0} \rho^n = \mathrm{id}_A \cup \bigcup_{n \geqslant 1} \rho^n,$$

где ρ^n, $n \geqslant 1$, — n-кратная композиция ρ с самим собой: $\rho^n = \underbrace{\rho \circ \ldots \circ \rho}_{n \text{ раз}}$. Как видно, в общем случае $\rho^* \neq \mathrm{id}_A$, т.е. в этом полукольце итерация элемента, вообще говоря, не равна единице полукольца. #

Выше было показано, что всякое замкнутое полукольцо является индуктивным упорядоченным множеством. Следовательно, согласно теореме 1.7, любое *непрерывное отображение* f замкнутого полукольца в себя имеет наименьшую *неподвижную точку*, т.е. в любом замкнутом полукольце всякое уравнение вида $x = f(x)$, где f — непрерывное отображение носителя этого полукольца в себя, имеет наименьшее решение.

Особенно важными для приложений являются линейные уравнения в замкнутом полукольце, имеющие вид

$$x = ax + b \tag{3.12}$$

или

$$x = xa + b. \tag{3.13}$$

В силу непрерывности операций сложения полукольца (см. тождество (3.10)) и умножения полукольца (см. определение 3.2) правые части уравнений (3.12) и (3.13) есть *непрерывные отображения*. Поэтому, согласно теореме 1.7 о *неподвижной точке*, существуют наименьшие решения этих уравнений.

Теорема 3.7. Наименьшими решениями уравнений (3.12) и (3.13) в замкнутых полукольцах являются соответственно

$$x = a^*b \tag{3.14}$$

и

$$x = ba^*, \tag{3.15}$$

где a^* — итерация элемента a.

◀ Используя формулу (1.8) для вычисления наименьшей неподвижной точки и записывая sup в случае замкнутого полукольца как бесконечную сумму, для уравнения (3.14) получаем

$$x = \sum_{n \geqslant 0}^{\infty} f^n(\mathbf{0}),$$

где $\mathbf{0}$ — нуль полукольца, а $f(x) = ax + b$.

Учитывая, что

$$\begin{aligned}
&f^0(\mathbf{0}) = \mathbf{0},\\
&f^1(\mathbf{0}) = b,\\
&f^2(\mathbf{0}) = ab + b = (a + \mathbf{1})b,\\
&\dots\dots\dots\dots\dots\dots\\
&f^n(\mathbf{0}) = (a^{n-1} + \ldots + a^2 + a + \mathbf{1})b,
\end{aligned}$$

получаем

$$\sum_{n=0}^{\infty} f^n(\mathbf{0}) = \sum_{n=1}^{\infty} (\mathbf{1} + a + \ldots + a^{n-1})b.$$

Используя непрерывность умножения, вынесем b (вправо) за знак бесконечной суммы и получим

$$\sum_{n=1}^{\infty} (\mathbf{1} + a + \ldots + a^{n-1})b = \left(\sum_{n=1}^{\infty} (\mathbf{1} + a + \ldots + a^{n-1})\right)b.$$

Сумма $1 + a + \ldots + a^{n-1}$ есть частичная сумма последовательности $\{a^n\}_{n \geqslant 0}$. Используя равенство (3.6), можем написать

$$\sum_{n=1}^{\infty}(1 + a + \ldots + a^{n-1}) = \sum_{n=0}^{\infty} a^n = a^*.$$

Окончательно получаем

$$\sum_{n=1}^{\infty} f^n(\mathbf{0}) = a^* b.$$

Анологично доказывается равенство (3.15). ▶

Формулы (3.14) и (3.15) дают именно наименьшие решения уравнений (3.12) и (3.13), а не все возможные их решения. Приведем в этой связи простой пример. В полукольце $\mathcal{B}$ (см. пример 3.2) можно определить только два уравнения: $x = x + 1$ и $x = x + 0$. Второе уравнение переписывается совсем просто: $x = x$; его решением является любой элемент полукольца — как 0, так и 1. Но по формуле (3.14) получим $x = 1^*0 = 0$, что, как и доказано выше, есть наименьшее решение уравнения.

Заметим еще, что в полукольце, в котором итерация любого элемента равна единице полукольца, формулы (3.14) и (3.15) дают один и тот же результат: $x = b$, т.е. в данном случае наименьшее решение совпадает со свободным членом уравнения.

3.3. Решение систем линейных уравнений

Полученные выше результаты (см. **3.2**) можно распространить на системы линейных уравнений вида

$$\begin{cases} x_1 = a_{11}x_1 + \ldots + a_{1n}x_n + b_1, \\ \ldots\ldots\ldots\ldots\ldots\ldots \\ x_n = a_{n1}x_1 + \ldots + a_{nn}x_n + b_n, \end{cases} \tag{3.16}$$

где все элементы a_{ij}, $1 \leqslant i \leqslant n$, $1 \leqslant j \leqslant n$, и b_i, $1 \leqslant i \leqslant n$, суть элементы некоторого *замкнутого полукольца*, и речь идет о решении системы (3.16) в этом полукольце.

Для этого введем в рассмотрение множество $\mathbb{M}_{m\times n}(\mathcal{S})$ прямоугольных матриц типа $m \times n$ с элементами из произвольного идемпотентного полукольца $\mathcal{S} = (S, +, \cdot, \mathbf{0}, \mathbf{1})$. Множество всех квадратных матриц порядка n с элементами из полукольца $\mathcal{S}$ обозначим $\mathbb{M}_n(\mathcal{S})$. Через $\mathbf{Matr}(\mathcal{S})$ обозначим множество всех матриц любого типа с элементами из $\mathcal{S}$.

Операции сложения и умножения матриц определяют точно так же, как и в числовом случае [III], — с учетом того, что сложение и умножение элементов матриц понимаются в смысле данного идемпотентного полукольца $\mathcal{S}$, а именно:

1) суммой матриц $A = (a_{ij})$ и $B = (b_{ij})$ типа $m \times n$ называют матрицу $C = (c_{ij})$ того же типа с элементами $c_{ij} = a_{ij} + b_{ij}$, $i = \overline{1, m}$, $j = \overline{1, n}$, и используют обозначение $C = A + B$;

2) произведением AB матриц $A = (a_{ij})$ типа $m \times n$ и $B = (b_{ij})$ типа $n \times p$ называют матрицу $C = (c_{ij})$ типа $m \times p$ с элементами

$$c_{ij} = \sum_{k=1}^{n} a_{ik} b_{kj}.$$

Нулевая (O) и единичная (E) матрицы любого порядка определяются с помощью единицы и нуля полукольца.

На множестве $\mathbb{M}_n(\mathcal{S})$ всех квадратных матриц фиксированного порядка n можно определить *алгебру*

$$\mathrm{M}_n(\mathcal{S}) = (\mathbb{M}_n(\mathcal{S}), +, \cdot, O, E).$$

Теорема 3.8. Алгебра $\mathrm{M}_n(\mathcal{S})$ есть идемпотентное полукольцо. Если полукольцо $\mathcal{S}$ замкнуто, то полукольцо $\mathrm{M}_n(\mathcal{S})$ тоже замкнуто.

◀ Операции суммы и произведения матриц определены таким образом, что все свойства операций сложения и умножения в полукольце сохраняются и для соответствующих операций над матрицами. Поэтому для суммы и произведения матриц из $\mathrm{M}_n(\mathcal{S})$ выполнены аксиомы полукольца, и, кроме того, операция сложения матриц идемпотентна. Следовательно, $\mathrm{M}_n(\mathcal{S})$ — идемпотентное полукольцо.

Выясним смысл *отношения порядка* в этом идемпотентном полукольце. В силу определения *естественного порядка идемпотентного полукольца* неравенство $A \leqslant B$ для матриц A и B означает, что $A + B = B$, или для всех i, j справедливо $a_{ij} + b_{ij} = b_{ij}$. Следовательно, $A \leqslant B$ тогда и только тогда, когда для всех i, j справедливо $a_{ij} \leqslant b_{ij}$.

Пусть $\mathcal{S}$ — замкнутое полукольцо. Докажем замкнутость идемпотентного полукольца $\mathrm{M}_n(\mathcal{S})$ и существование *точной верхней грани* у любой последовательности матриц в $\mathrm{M}_n(\mathcal{S})$.

Пусть $\{A_m\}_{m\in\mathbf{N}}$ — произвольная последовательность квадратных матриц $A_m = (a_{ij}^m)$ порядка n. Рассмотрим матрицу $B = \Big(\sum\limits_{m\in\mathbf{N}} a_{ij}^m\Big)$. Каждый элемент $b_{ij} = \sum\limits_{m\in\mathbf{N}} a_{ij}^m$ этой матрицы есть точная верхняя грань последовательности элементов a_{ij}^m. Эти точные верхние грани существуют, поскольку a_{ij}^m — элементы замкнутого полукольца $\mathcal{S}$. Так как сложение матриц и отношение порядка в полукольце матриц определяются поэлементно, то матрица B и будет точной верхней гранью последовательности матриц A_m.

Докажем теперь непрерывность умножения в $\mathrm{M}_n(\mathcal{S})$, т.е. что для любой последовательности $\{A_m\}_{m\in\mathbf{N}}$ матриц и произвольной матрицы B имеет место

$$B\sum A_m = \sum(BA_m).$$

Матрица $C = (c_{ij}) = \sum A_m$ есть точная верхняя грань последовательности $\{A_m\}_{m\in\mathbf{N}}$. Тогда имеем

$$B\sum A_m = BC = \Big(\sum_{k=1}^{n} b_{ik}c_{kj}\Big).$$

Элемент c_{kj} есть точная верхняя грань последовательности $\{a_{kj}^m\}_{m\in\mathbf{N}}$ элементов матриц A_m, т.е.

$$c_{ij} = \sum_{m\in\mathbf{N}} a_{ij}^m.$$

Используя непрерывность умножения в исходном полукольце $\mathcal{S}$, получаем

$$\sum_{k=1}^{n} b_{ik}c_{ki} = \Big(\sum_{k=1}^{n} b_{ik} \sum_{m\in\mathbf{N}} a_{kj}^{m}\Big) = \Big(\sum_{k=1}^{n}\sum_{m\in\mathbf{N}} b_{ik}a_{kj}^{m}\Big).$$

Следовательно,

$$B\sum_{m\in\mathbf{N}} A_m = \Big(\sum_{k=1}^{n} b_{ik} \sum_{m\in\mathbf{N}} a_{kj}^{m}\Big).$$

Используя непрерывность сложения, получаем

$$\Big(\sum_{k=1}^{n} b_{ik} \sum_{m\in\mathbf{N}} a_{kj}^{m}\Big) = \sum_{m\in\mathbf{N}}\Big(\sum_{k=1}^{n} b_{ik}a_{kj}^{m}\Big) = \sum_{m\in\mathbf{N}}(BA_m).$$

Аналогично доказывается, что $(\sum A_m)B = \sum(A_mB)$. ▶

Полукольцо $\mathrm{M}_n(\mathcal{S})$ будем называть ***полукольцом матриц над полукольцом*** $\mathcal{S}$. Доказанная теорема позволяет нам применять к замкнутым полукольцам матриц над некоторым замкнутым полукольцом $\mathcal{S}$ теорему 3.7 и решать произвольные уравнения вида

$$X = AX + B \tag{3.17}$$

или

$$X = XA + B \tag{3.18}$$

относительно неизвестной матрицы X.

Наименьшие решения этих уравнений есть

$$X = A^*B \tag{3.19}$$

и

$$X = BA^* \tag{3.20}$$

соответственно, где A^* — *итерация* матрицы A в $\mathrm{M}_n(\mathcal{S})$. Итерация A^* матрицы A играет в теории линейных уравнений в

замкнутых полукольцах такую же роль, как обратная матрица в классической линейной алгебре.

Основную роль при решении задач теории *ориентированных графов* и теории формальных языков играют ***праволинейные уравнения*** вида (3.17), поэтому мы будем, как правило, рассматривать только их. ***Леволинейное уравнение*** (3.18) может быть проанализировано аналогично.

Мы доказали существование решений матричных уравнений в матричном полукольце над замкнутым полукольцом. Теперь нам необходимо разработать технику поиска их решений и применить ее к решению систем вида (3.16).

Полагая, что ξ_j — j-й столбец матрицы X, а β_j — j-й столбец матрицы B, уравнение (3.17) можно переписать как систему уравнений относительно неизвестных столбцов матрицы X:

$$\xi_j = A\xi_j + \beta_j, \quad 0 \leqslant j \leqslant n. \tag{3.21}$$

Каждая система вида (3.21) есть матричная форма записи указанной выше системы (3.16). Поэтому наименьшее решение этой системы, как следует из (3.19), есть

$$\xi_j = A^*\beta_j. \tag{3.22}$$

Для поиска решения системы вида (3.21) можно воспользоваться ***методом последовательного исключения неизвестных***, аналогичным классическому *методу Гаусса*.

Поскольку система уравнений вида (3.16) имеет решение, мы можем подставить его в систему и работать с уравнениями как с тождествами.

Рассмотрим процедуру решения системы уравнений (3.16). Запишем первое уравнение системы так:

$$x_1 = a_{11}x_1 + (a_{12}x_2 + \ldots + a_{1n}x_n + b_1).$$

Из первого уравнения системы выразим x_1 через остальные неизвестные, воспользовавшись формулой (3.14):

$$x_1 = a_{11}^*(a_{12}x_2 + \ldots + a_{1n}x_n + b_1). \tag{3.23}$$

В формуле (3.23) выражение в скобках не содержит неизвестного x_1. Подставляя (3.23) вместо x_1 в остальные уравнения, получаем систему из $n-1$ уравнений, которая уже не содержит x_1:

$$\begin{cases} x_2 = a_{21}a_{11}^*(a_{12}x_2 + \ldots + a_{1n}x_n + b_1) + \\ \qquad\qquad + a_{22}x_2 + \ldots + a_{2n}x_n + b_2, \\ x_3 = a_{31}a_{11}^*(a_{12}x_2 + \ldots + a_{1n}x_n + b_1) + \\ \qquad\qquad + a_{32}x_2 + \ldots + a_{3n}x_n + b_3, \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ x_n = a_{n1}a_{11}^*(a_{12}x_2 + \ldots + a_{1n}x_n + b_1) + \\ \qquad\qquad + a_{n2}x_2 + \ldots + a_{nn}x_n + b_n. \end{cases} \tag{3.24}$$

Приводя подобные члены в каждом уравнении системы, получаем:

$$\begin{cases} x_2 = (a_{21}a_{11}^*a_{12} + a_{22})x_2 + \ldots \\ \qquad \ldots + (a_{21}a_{11}^*a_{1n} + a_{2n})x_n + a_{21}a_{11}^*b_1 + b_2, \\ x_3 = (a_{31}a_{11}^*a_{12} + a_{32})x_2 + \ldots \\ \qquad \ldots + (a_{31}a_{11}^*a_{1n} + a_{3n})x_n + a_{31}a_{11}^*b_1 + b_3, \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ x_n = (a_{n1}a_{11}^*a_{12} + a_{n2})x_2 + \ldots \\ \qquad \ldots + (a_{n1}a_{11}^*a_{1n} + a_{nn})x_n + a_{n1}a_{11}^*b_1 + b_n. \end{cases} \tag{3.25}$$

Первое уравнение этой системы перепишем так:

$$x_2 = (a_{21}a_{11}^*a_{12} + a_{22})x_2 + \gamma_2,$$

где

$$\gamma_2 = a_{21}a_{11}^*(a_{13}x_3 + \ldots + a_{1n}x_n + b_1) + a_{23}x_3 + \ldots + a_{2n}x_n + b_2.$$

Заметим, что γ_2 не содержит x_1 и x_2. Воспользовавшись соотношением (3.14), будем иметь

$$x_2 = \alpha_2^*\gamma_2, \tag{3.26}$$

где $\alpha_2 = a_{21}a_{11}^*a_{12} + a_{22}$ не содержит неизвестных. Используя полученное выражение, исключаем x_2 из остальных уравнений.

Действуя подобным образом, на i-м шаге ($1 \leqslant i \leqslant n$) получаем

$$x_i = \alpha_i^* \gamma_i, \tag{3.27}$$

где выражение α_i^* не содержит неизвестных, а выражение γ_i может содержать только неизвестные, начиная с $(i+1)$-го, т.е. $x_{i+1}, \ldots, x_n$.

При $i = n$ имеем

$$x_n = \alpha_n^* \gamma_n, \tag{3.28}$$

где выражения α_n^* и γ_n не содержат неизвестных. Таким образом, исходная система (3.16) преобразована к „треугольному“ виду: правая часть уравнения (3.28) не содержит неизвестных, уравнение (3.27) при $i = n-1$ в правой части содержит только одно неизвестное x_{n-1} и каждое следующее уравнение при просмотре „снизу вверх“ на одно неизвестное больше, чем предыдущее. Первое уравнение системы — уравнение (3.23) — в правой части содержит все неизвестные от x_2 до x_n. На этом завершается первый этап алгоритма, который называют *прямым ходом метода Гаусса*.

Второй этап алгоритма, называемый *обратным ходом метода Гаусса*, состоит в последовательном нахождении значения всех неизвестных $x_1, \ldots, x_{n-1}$, начиная с x_{n-1}. Подставив в выражение для x_{n-1} вместо x_n правую часть (3.28), найдем x_{n-1}. Затем определим x_{n-2}, подставив полученные значения x_n и x_{n-1} в правую часть выражения (3.27) при $i = n-2$, и так далее до тех пор, пока не найдем x_1.

Замечание 3.2. Положив $B = E$ в уравнении (3.17), получим $X = A^*E = A^*$. Таким образом, чтобы вычислить итерацию матрицы A, достаточно решить матричное уравнение (3.21) для всех $j = \overline{1, n}$ при β_j, равном j-му столбцу единичной матрицы E.

Пример 3.6. Проиллюстрируем приведенную схему решения системы из двух линейных уравнений. Имеем

$$\begin{cases} x_1 = a_{11}x_1 + a_{12}x_2 + b_1, \\ x_2 = a_{21}x_1 + a_{22}x_2 + b_2. \end{cases}$$

Из первого уравнения выразим x_1:

$$x_1 = a_{11}^*(a_{12}x_2 + b_1).$$

Подставляя это выражение во второе уравнение, получаем

$$x_2 = (a_{21}a_{11}^*a_{12} + a_{22})^*(a_{21}a_{11}^*b_1 + b_2).$$

Подставляя этот результат в написанное выше выражение для x_1, находим окончательное решение:

$$x_1 = a_{11}^*(a_{12}(a_{21}a_{11}^*a_{12} + a_{22})^*(a_{21}a_{11}^*b_1 + b_2) + b_1).$$

Особенно просто решение выглядит в случае тривиальной итерации, т.е. тогда, когда в полукольце итерация любого элемента равна единице полукольца (как в полукольцах $\mathcal{B}$, $\mathcal{R}^+$, $\mathcal{S}_A$, $\mathcal{S}_{[a,b]}$). В этом случае для системы из двух уравнений решение равно

$$x_1 = a_{12}(a_{21}b_1 + b_2) + b_1,$$
$$x_2 = a_{21}b_1 + b_2.$$

Пример 3.7. Рассмотрим в полукольце $\mathcal{S}_{[0,1]}$ (см. пример 3.3.г) систему линейных уравнений

$$\begin{cases} x_1 = 0{,}5x_1 + 0{,}4x_2 + 0{,}2, \\ x_2 = 0{,}3x_1 + 0{,}9x_2 + 0{,}6. \end{cases}$$

Решим эту систему уравнений, следуя общему алгоритму. Из первого уравнения получаем

$$x_1 = 0{,}5^*(0{,}4x_2 + 0{,}2) = 1(0{,}4x_2 + 0{,}2) = 0{,}4x_2 + 0{,}2.$$

Далее,

$$x_2 = 0{,}3(0{,}4x_2 + 0{,}2) + 0{,}9x_2 + 0{,}6 = 0{,}3x_2 + 0{,}2 + 0{,}9x_2 + 0{,}6.$$

Отсюда $x_2 = (0{,}3 + 0{,}9)x_2 + 0{,}6 = 0{,}9x_2 + 0{,}6 = 0{,}9^*0{,}6 = 0{,}6$.

Подставляя $x_2 = 0{,}6$ в полученное выражение для x_1, находим, что

$$x_1 = 0{,}4 \cdot 0{,}6 + 0{,}2 = 0{,}4 + 0{,}2 = 0{,}4. \quad \#$$

Не всякое бесконечное идемпотентное полукольцо является замкнутым. Однако можно заметить, что при решении линейных уравнений и систем требовалось вычисление точной верхней грани последовательностей специального вида, а именно нахождение итерации элементов. Поэтому помимо замкнутых полуколец интерес для приложений представляют так называемые ***полукольца с итерацией***.

Под полукольцом с итерацией в данном контексте мы будем понимать идемпотентное полукольцо, которое является подполукольцом* некоторого замкнутого полукольца и вместе с каждым элементом содержит его итерацию. Важнейшим примером такого полукольца является *полукольцо регулярных языков* (см. **7**).

Рассматривая в полукольце с итерацией произвольное линейное уравнение, т.е. уравнение вида (3.12) или (3.13), получаем следующие результаты. Во-первых, это уравнение имеет наименьшее решение, так как полукольцо с итерацией содержится в некотором замкнутом полукольце в качестве подполукольца. Во-вторых, это наименьшее решение снова окажется в этом же полукольце, поскольку носитель полукольца с итерацией замкнут относительно итерации. Таким образом, носитель полукольца $\mathcal{S}$ с итерацией замкнут относительно операции нахождения наименьшей неподвижной точки любого линейного отображения $ax + b$ (или $xa + b$), где a и b — элементы $\mathcal{S}$.

Не составляет труда распространить этот результат на произвольные матричные уравнения. Можно доказать следующее утверждение.

*Полукольцо $\mathcal{Q} = (Q, +, \cdot, \mathbf{0}, \mathbf{1})$ называют ***подполукольцом*** полукольца $\mathcal{R} = (R, +, \cdot, \mathbf{0}, \mathbf{1})$, если множество Q есть подмножество множества R, замкнутое относительно операций сложения и умножения полукольца $\mathcal{R}$, а также содержащее нуль и единицу полукольца $\mathcal{R}$.

Теорема 3.9. Если A — матрица, все элементы которой принадлежат некоторому полукольцу с итерацией, то все элементы ее *итерации* A^* также принадлежат этому полукольцу с итерацией.

3.4. Булевы алгебры

Теория *булевых алгебр* является классическим разделом дискретной математики. Булевы алгебры возникли в трудах английского математика Дж. Буля в 50-х годах XIX века как аппарат логики. При этом элементы булевой алгебры трактовались как высказывания, а операциями являлись *дизъюнкция*, *конъюнкция* и *отрицание*.

Существуют различные подходы к определению булевой алгебры. Мы определим булеву алгебру как частный случай *идемпотентного полукольца*.

Определение 3.3. Полукольцо $\mathcal{S} = (S, +, \cdot, \mathbf{0}, \mathbf{1})$ называют ***симметричным*** (или ***взаимным***), если оно идемпотентно и в нем выполнены следующие тождества:

1) $a \cdot a = a$ (*идемпотентность операции умножения полукольца*);

2) $a \cdot b = b \cdot a$ (*коммутативность операции* умножения полукольца);

3) $a + (b \cdot c) = (a + b) \cdot (a + c)$ (*дистрибутивность операции сложения полукольца* относительно умножения);

4) $\mathbf{1} + a = \mathbf{1}$ (*аннулирующее свойство единицы полукольца* относительно сложения).

Можно дать определение, равносильное определению 3.3. Идемпотентное полукольцо $\mathcal{S} = (S, +, \cdot, \mathbf{0}, \mathbf{1})$ называется симметричным (или взаимным), если алгебра $\mathcal{S}' = (S, \cdot, +, \mathbf{1}, \mathbf{0})$ также является идемпотентным полукольцом. При этом будем говорить, что идемпотентное полукольцо $\mathcal{S}'$ есть ***полукольцо, двойственное*** к идемпотентному полукольцу $\mathcal{S}$.

Из определения следует, что в двойственном идемпотентном полукольце операция сложения — это операция умножения в исходном полукольце и наоборот; *нуль* двойственного *полукольца* — это единица исходного полукольца и наоборот. Легко видеть, что полукольцо $\mathcal{S}''$, двойственное к двойственному полукольцу $\mathcal{S}'$, есть исходное полукольцо $\mathcal{S}$.

Запишем полностью все аксиомы (основные тождества) симметричного полукольца, объединяя двойственные аксиомы в пары (табл. 3.3).

Таблица 3.3

№ п/п	Аксиома	№ п/п	Аксиома
1	$a+(b+c)=(a+b)+c$	7	$a\cdot(b\cdot c)=(a\cdot b)\cdot c$
2	$a+b=b+a$	8	$a\cdot b=b\cdot a$
3	$a+a=a$	9	$a\cdot a=a$
4	$a+\mathbf{0}=a$	10	$a\cdot\mathbf{1}=a$
5	$a\cdot(b+c)=$ $=(a\cdot b)+(a\cdot c)$	11	$a+(b\cdot c)=$ $=(a+b)\cdot(a+c)$
6	$a\cdot\mathbf{0}=\mathbf{0}$	12	$a+\mathbf{1}=\mathbf{1}$

В табл. 3.1 можно увидеть, что в симметричном полукольце операции сложения и умножения, равно как и элементы $\mathbf{0}$ и $\mathbf{1}$, полностью „взаимозаменяемые“, или взаимно двойственные.

Таким образом, справедлив ***принцип двойственности*** для симметричных полуколец: любое тождество в симметричном полукольце остается верным, если в нем операцию сложения заменить операцией умножения и наоборот, а единицу заменить нулем и наоборот.

Пример 3.8. **а.** Полукольцо $\mathcal{B}$ (см. пример 3.2) симметричное.

б. Полукольцо $\mathcal{R}^+$ (см. пример 3.1) не является симметричным в силу неидемпотентности умножения полукольца, хотя в

нем единица полукольца (число 0) имеет аннулирующее свойство, поскольку $\min(0,\, x) = 0$.

в. Полукольцо $\mathcal{S}_A$ (см. пример 3.3.б) симметрично в силу известных свойств операций пересечения и объединения множеств.

г. Полукольцо $\mathcal{R}_A$ (см. пример 3.3.в) не является симметричным.

д. Полукольцо $\mathcal{S}_{[a,\,b]}$ (см. пример 3.3.г) симметрично.

Пример 3.9. Рассмотрим алгебру

$$\mathcal{D}_n = (\text{Дел}(n),\ \text{НОК},\ \text{НОД},\ 1,\ n),$$

где Дел(n) — множество всех делителей натурального числа n; НОК — операция вычисления наименьшего общего кратного; НОД — операция вычисления наибольшего общего делителя двух чисел. Эта алгебра является симметричным полукольцом.

Действительно, для любых двух натуральных чисел m и p верно представление

$$m = 2^{\alpha_1} 3^{\alpha_2} \ldots p_k^{\alpha_k} \quad \text{и} \quad p = 2^{\beta_1} 3^{\beta_2} \ldots p_k^{\beta_k},$$

где p_k — наибольший простой множитель в разложении m и p. Тогда

$$\text{НОК}(m,p) = 2^{\max(\alpha_1,\beta_1)} 3^{\max(\alpha_2,\beta_2)} \ldots p_k^{\max(\alpha_n,\beta_n)},$$

$$\text{НОД}(m,p) = 2^{\min(\alpha_1,\beta_1)} 3^{\min(\alpha_2,\beta_2)} \ldots p_k^{\min(\alpha_n,\beta_n)}.$$

Таким образом, свойства операций НОК и НОД определяются свойствами операций max и min. В силу этого рассматриваемая алгебра и является симметричным полукольцом (см. пример 3.3.г). #

Рассмотрим некоторые свойства симметричного полукольца, вытекающие из его аксиом.

Свойство 3.1. Для любых элементов симметричного полукольца выполняются равенства

$$x(x+y)=x, \quad x+xy=x.$$

◀ Имеем $x(x+y)=xx+xy=x+xy=x(1+y)=x\cdot 1=x$. ▶

Равенства, приведенные в формулировке свойства 3.1, называют ***тождествами поглощения.***

Свойство 3.2. В симметричном полукольце неравенство $x\leqslant y$ имеет место тогда и только тогда, когда $xy=x$.

◀ Вспомним, что по определению *естественного порядка* произвольного *идемпотентного полукольца* $x\leqslant y \Leftrightarrow x+y=y$.

Пусть $x\leqslant y$. Тогда $xy=x(x+y)=x$ (по свойству 3.1).

Пусть теперь $xy=x$. Тогда $x+y=xy+y=y$. Следовательно, $x\leqslant y$. ▶

Свойство 3.2 выражает связь умножения с *естественным порядком идемпотентного полукольца*: меньший сомножитель поглощает больший, т.е. порядок в двойственном полукольце S' является *порядком, двойственным* к порядку в полукольце S. Но, как известно, при переходе к двойственному порядку *наибольший* (*максимальный*) *элемент* становится *наименьшим* (*минимальным*) *элементом*, а *точная верхняя грань* — *точной нижней гранью.*

Свойство 3.3. В симметричном полукольце произведение xy есть точная нижняя грань последовательности $\{x,y\}$:

$$xy=\inf\{x,y\}.$$

Свойство 3.4. Для любого элемента x симметричного полукольца имеет место неравенство $0\leqslant x\leqslant 1$.

◀ Первое неравенство $0\leqslant x$ равносильно равенству $0+x=x$, верному для любого x. Второе неравенство $x\leqslant 1$ вытекает из четвертого тождества определения 3.3. ▶

Таким образом, в симметричном полукольце единица (1) является наибольшим элементом.

Определение 3.4. ***Булева алгебра*** — это симметричное полукольцо, в котором для каждого x существует элемент $\overline{x}$, называемый ***дополнением*** x, такой, что

$$x + \overline{x} = \mathbf{1}, \tag{3.29}$$

$$x \cdot \overline{x} = \mathbf{0}. \tag{3.30}$$

Обычно сложение в булевой алгебре называют ***булевым объединением*** и обозначают $\vee$, а умножение — ***булевым пересечением*** и обозначают $\wedge$.

Запишем аксиомы булевой алгебры в виде табл. 3.4, объединяя „двойственные пары“ (как это мы уже сделали, записывая аксиомы симметричного идемпотентного полукольца).

Таблица 3.4

№ п/п	Аксиома	№ п/п	Аксиома
1	$a \vee (b \vee c) = (a \vee b) \vee c$	8	$a \wedge (b \wedge c) = (a \wedge b) \wedge c$
2	$a \vee b = b \vee a$	9	$a \wedge b = b \wedge a$
3	$a \vee a = a$	10	$a \wedge a = a$
4	$a \vee \mathbf{0} = a$	11	$a \wedge \mathbf{1} = a$
5	$a \wedge (b \vee c) = (a \wedge b) \vee (a \wedge c)$	12	$a \vee (b \wedge c) = (a \vee b) \wedge (a \vee c)$
6	$a \wedge \mathbf{0} = \mathbf{0}$	13	$a \vee \mathbf{1} = \mathbf{1}$
7	$a \vee \overline{a} = \mathbf{1}$	14	$a \wedge \overline{a} = \mathbf{0}$

Рассмотрим некоторые важные свойства булевых алгебр, вытекающие из определения.

Свойство 3.5 (единственность дополнения). В булевой алгебре для любого x его дополнение $\overline{x}$ единственное.

◀ Пусть для элемента x найдется еще одно такое a, что $a \wedge x = \mathbf{0}$ и $a \vee x = \mathbf{1}$.

Имеем $a = a \vee \mathbf{0}$. Воспользовавшись свойством (3.29), получим $a = a \vee (x \wedge \overline{x})$. В силу дистрибутивности и с учетом свойств элемента a имеем $a = (a \vee x) \wedge (a \vee \overline{x}) = \mathbf{1} \wedge (a \vee \overline{x})$. С учетом свойств дополнения преобразуем последнее выражение следующим образом: $a = (x \vee \overline{x}) \wedge (a \vee \overline{x}) = (x \wedge a) \vee \overline{x}$. Поскольку $x \wedge a = \mathbf{0}$, то $a = \mathbf{0} \vee \overline{x} = \overline{x}$. Таким образом, элемент a совпадает с дополнением x. ▶

Свойство 3.6 („симметричность“ операции дополнения). В булевой алгебре выполняется тождество

$$\overline{\overline{x}} = x.$$

◀ Так как $\overline{\overline{x}}$ является дополнением к $\overline{x}$, то $\overline{\overline{x}} \wedge \overline{x} = \mathbf{0}$ и $\overline{\overline{x}} \vee \overline{x} = \mathbf{1}$. В то же время $\overline{x} \wedge x = \mathbf{0}$ и $\overline{x} \vee x = \mathbf{1}$. В силу единственности дополнения к элементу $\overline{x}$ имеем $\overline{\overline{x}} = x$. ▶

Свойство 3.7. В булевой алгебре верны следующие тождества:

$$\overline{x \vee y} = \overline{x} \wedge \overline{y}, \quad \overline{x \wedge y} = \overline{x} \vee \overline{y}. \tag{3.31}$$

◀ В силу свойств 3.5 и 3.6 для доказательства первого закона достаточно показать, что $(\overline{x} \wedge \overline{y}) \vee (x \vee y) = \mathbf{1}$ и $(\overline{x} \wedge \overline{y}) \wedge (x \vee y) = \mathbf{0}$.

Преобразуя выражения в левых частях, получаем

$$(\overline{x} \wedge \overline{y}) \vee (x \vee y) = (\overline{x} \vee x \vee y) \wedge (\overline{y} \vee x \vee y) = \mathbf{1} \wedge \mathbf{1} = \mathbf{1},$$
$$(\overline{x} \wedge \overline{y}) \wedge (x \vee y) = (\overline{x} \wedge \overline{y} \wedge x) \vee (\overline{x} \wedge \overline{y} \wedge y) = \mathbf{0} \vee \mathbf{0} = \mathbf{0}.$$

Первое тождество доказано. Второе тождество следует из принципа двойственности. ▶

Тождества (3.31) называют ***законами де Моргана.***

Единственность дополнения означает, что в булевой алгебре возникает унарная операция — переход от элемента к его дополнению. Эту операцию можно ввести в сигнатуру алгебры, т.е. рассматривать булеву алгебру как алгебру вида

$$\mathrm{B} = (B, \vee, \wedge, {}^{-}, \mathbf{0}, \mathbf{1})$$

с двумя бинарными, одной унарной и двумя нульарными операциями, такую, что:

1) $(B, \vee, \wedge, \mathbf{0}, \mathbf{1})$ — симметричное полукольцо;

2) $a \vee \overline{a} = \mathbf{1}$ и $a \wedge \overline{a} = \mathbf{0}$ (для любого a).

Дополнение в булевой алгебре называют ***булевым дополнением***, а все операции булевой алгебры — ***булевыми операциями***.

Рассмотрим теперь некоторые примеры булевых алгебр.

Пример 3.10. Полукольцо $\mathcal{B}$ (см. пример 3.2) является булевой алгеброй. Эта булева алгебра — важнейшая структура. Мы назовем ее ***двухэлементной булевой алгеброй*** и обозначим $\mathbb{B}$. Видно, что в $\mathbb{B}$

$$\overline{0} = 1, \quad \overline{1} = 0.$$

Пример 3.11. На множестве $\{0,1\}^n$ определим структуру булевой алгебры, положив для произвольных $\widetilde{\alpha} = (\alpha_1, \ldots, \alpha_n)$, $\widetilde{\beta} = (\beta_1, \ldots, \beta_n)$ из $\{0,1\}^n$, что

$$\begin{aligned}
&\widetilde{\alpha} \vee \widetilde{\beta} = (\alpha_1 \vee \beta_1, \ldots, \alpha_n \vee \beta_n),\\
&\widetilde{\alpha} \wedge \widetilde{\beta} = (\alpha_1 \wedge \beta_1, \ldots, \alpha_n \wedge \beta_n),\\
&\overline{\widetilde{\alpha}} = (\overline{\alpha}_1, \ldots, \overline{\alpha}_n),\\
&\mathbf{0} = (0, \ldots, 0),\\
&\mathbf{1} = (1, \ldots, 1).
\end{aligned}$$

Можно без труда показать, что все аксиомы булевой алгебры выполняются. Носитель определенной таким образом булевой алгебры называют ***булевым кубом*** размерности n, а его элементы — ***булевыми векторами*** (или ***булевыми наборами***) размерности n. Вектор $\mathbf{0}$ называют при этом ***нулевым булевым вектором*** или ***нулевым набором***, а вектор $\mathbf{1}$ — ***единичным булевым вектором*** или ***единичным набором***. Заметим, что случаи $n = 0$ и $n = 1$ включаются в эту конструкцию. При $n = 1$ получаем уже рассмотренную двухэлементную булеву алгебру $\mathbb{B}$, а при $n = 0$ — так называемую

одноэлементную булеву алгебру, в которой $\mathbf{0} = \mathbf{1}$. Но эта структура малоинтересна.

Итак, булевы операции над булевыми векторами выполняются покомпонетно — так же, как сложение векторов или умножение вектора на число в линейной алгебре. Отношение порядка здесь определено также покомпонентно, т.е. для произвольных $\widetilde{\alpha} = (\alpha_1, \ldots, \alpha_n)$, $\widetilde{\beta} = (\beta_1, \ldots, \beta_n) \in \{0, 1\}^n$ неравенство $\widetilde{\alpha} \leqslant \widetilde{\beta}$ означает, что $\alpha_i \leqslant \beta_i$, $i = \overline{1, n}$. Так, например,

$$(0, 1, 0, 0, 1) \leqslant (1, 1, 0, 0, 1),$$

а векторы $(0, 1, 0, 0, 1)$ и $(0, 1, 0, 0, 0)$ не сравнимы.

Пример 3.12. Полукольцо $\mathcal{S}_A$ (см. пример 3.3.б) — булева алгебра, в которой все булевы операции суть не что иное, как обычные теоретико-множественные операции, т.е. булево объединение есть обычное объединение множеств, булево пересечение — пересечение множеств, булево дополнение — дополнение множества.

Пример 3.13. **а.** Рассмотрим полукольцо $\mathcal{D}_6$ делителей числа 6 с операциями НОК и НОД. Из примера 3.9 следует, что это полукольцо симметричное. Нуль этого полукольца есть число 1, а единица — число 6. Убедимся, что каждый элемент полукольца имеет дополнение. Начнем с числа 1. Дополнение x должно удовлетворять равенствам $1 \vee x = 6$ и $1 \wedge x = 1$. Первое равенство означает, что $\text{НОК}(1, x) = 6$, а второе — $\text{НОД}(1, x) = 1$. Легко видеть, что $x = \overline{1} = 6$. Рассуждая аналогично, получим $\overline{2} = 3$. Следовательно, рассматриваемое полукольцо есть булева алгебра.

б. Полукольцо $\mathcal{D}_{12}$ делителей числа 12 не является булевой алгеброй, так как, например, из $2 \vee x = \text{НОК}(2, x) = 12 = \mathbf{1}$ следует, что $x = 12$, но $2 \wedge 12 = \text{НОД}(2, 12) = 2 \neq 1 = \mathbf{0}$, и элемент 2 не имеет дополнения. #

Теория булевых алгебр имеет многочисленные приложения: в математической логике, в теории вероятностей. Она позволяет, в частности, рассматривать с единой точки зрения операции

над множествами, над высказываниями, над случайными *событиями*. В этой книге мы используем изложенные здесь факты в главе **6**, посвященной *булевым функциям*.

3.5. Решетки

Напомним, что *полурешеткой* называют *полугруппу*, *операция* которой *коммутативна* и *идемпотентна*.

Таким образом, полурешетка — это *алгебра* $\mathcal{L} = (L, \vee)$, в которой для любых a, b, $c \in L$ выполнены равенства

$$\begin{aligned} &a \vee (b \vee c) = (a \vee b) \vee c, \\ &a \vee b = b \vee a, \\ &a \vee a = a. \end{aligned}$$

В каждой полурешетке может быть определено естественное *отношение порядка* аналогично тому, как это определялось для (идемпотентного) *полукольца*. По определению полагаем для произвольных a, $b \in L$

$$a \leqslant b \Leftrightarrow a \vee b = b. \tag{3.32}$$

Из определения (3.32) отношения порядка в полурешетке следует, что элемент $a \vee b$ есть *точная верхняя грань двухэлементного множества* $\{a, b\}$.

Действительно,

$$a \vee (a \vee b) = (a \vee a) \vee b = a \vee b$$

(в силу ассоциативности и идемпотентности операции $\vee$). Аналогично $b \vee (a \vee b) = a \vee b$ (с использованием и коммутативности операции $\vee$). Итак, $a \vee b$ есть *верхняя грань* множества $\{a, b\}$. Полагая, что c есть какая-то верхняя грань данного множества, вычислим $(a \vee b) \vee c$. Используя ассоциативность и то, что в силу $b \leqslant c$ выполняется равенство $b \vee c = c$, получим

$$(a \vee b) \vee c = a \vee (b \vee c) = a \vee c.$$

Но так как $a \leqslant c$, то $a \vee c = c$. Итак, $(a \vee b) \vee c = c$, т.е. $a \vee b \leqslant c$, и поэтому элемент $a \vee b$ есть точная верхняя грань множества $\{a, b\}$.

Отношение порядка, определенное в произвольной полурешетке согласно условию (3.32), будем называть ***естественным порядком*** данной ***полурешетки.***

Таким образом, в полурешетке любое двухэлементное (и, следовательно, любое конечное) подмножество имеет точную верхнюю грань (по естественному порядку полурешетки).

Можно доказать, что имеет место и обратное.

Теорема 3.10. Любое *упорядоченное множество* $\mathcal{L} = (L, \leqslant)$, в котором всякое двухэлементное подмножество имеет точную верхнюю грань, является полурешеткой, естественный порядок которой совпадает с отношением $\leqslant$.

◀ Определим операцию $\vee$ так: $a \vee b = \sup\{a, b\}$. Коммутативность и идемпотентность операции $\vee$ следует сразу из определения точной верхней грани множества. Действительно, $a \vee a = \sup\{a, a\} = \sup\{a\} = a$, и так как $\{a, b\} = \{b, a\}$, то $a \vee b = \sup\{a, b\} = \sup\{b, a\} = b \vee a$.

Докажем ассоциативность операции $\vee$. Для этого нужно показать, что для произвольных a, b, $c \in L$ имеет место равенство

$$\sup\{\sup\{a, b\}, c\} = \sup\{a, \sup\{b, c\}\}. \tag{3.33}$$

Обозначим левую часть равенства (3.33) через d_1, а правую — через d_2. Элемент d_1 является точной верхней гранью множества, элементы которого суть $\sup\{a, b\}$ и c. Поэтому $d_1 \geqslant \sup\{a, b\}$ и $d_1 \geqslant c$. Из первого неравенства следует, что $d_1 \geqslant a$ и $d_1 \geqslant b$. Тогда, поскольку $d_1 \geqslant b$ и $d_1 \geqslant c$, то d_1 есть верхняя грань множества $\{b, c\}$, т.е. $d_1 \geqslant \sup\{b, c\}$. Так как $d_1 \geqslant a$, то $d_1 \geqslant \sup\{a, \sup\{b, c\}\} = d_2$. Аналогично доказывается, что $d_2 \geqslant d_1$, т.е. $d_1 = d_2$, что и требовалось доказать.

Поскольку для любых элементов a, b упорядоченного множества $(L, \leqslant)$ согласно условию теоремы соотношение $a \leqslant b$

эквивалентно тому, что $\sup\{a, b\} = b$, то исходное отношение порядка $\leqslant$ совпадает с естественным порядком полурешетки $(L, \sup)$. ▶

Из принципа двойственности для упорядоченных множеств следует, что точная верхняя (нижняя) грань любого подмножества упорядоченного множества $(L, \leqslant)$ является одновременно точной нижней (верхней) гранью этого подмножества в смысле *двойственного порядка* $\geqslant$. Отсюда получаем утверждение, двойственное теореме 3.10.

Теорема 3.11. Любое упорядоченное множество $\mathcal{L} = (L, \leqslant)$, в котором всякое двухэлементное подмножество имеет точную нижнюю грань, является полурешеткой, причем естественный порядок этой полурешетки является порядком, двойственным к исходному порядку $\leqslant$.

◀ Доказательство дословно повторяет доказательство теоремы 3.10, но операция $\wedge$ полурешетки определяется так:

$$a \wedge b = \inf\{a, b\}. \qquad \blacktriangleright$$

Полурешетку $(L, \sup)$, определенную теоремой 3.10, называют ***верхней полурешеткой***, а полурешетку $(L, \inf)$, определяемую теоремой 3.11, — ***нижней полурешеткой***.

Замечание 3.3. Обратим внимание на то, что понятия „верхний“ и „нижний“ имеют смысл относительно фиксированного отношения порядка и при переходе к двойственному порядку „верх“ превращается в „низ“ и наоборот.

Пример 3.14. а. Отрезок $[a, b]$ числовой прямой (с естественным числовым порядком), согласно теоремам 3.10 и 3.11, является и верхней и нижней полурешеткой. Более того, поскольку на числовой прямой любое двухэлементное подмножество имеет как точную верхнюю, так и точную нижнюю грань, то любое подмножество множества действительных чисел $\mathbb{R}$ (в частности, само $\mathbb{R}$) является и верхней и нижней полурешеткой.

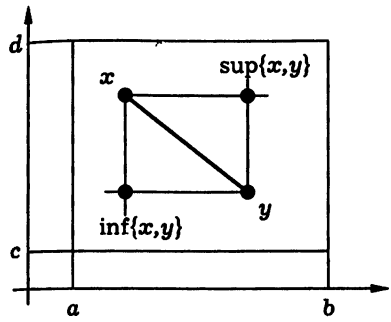


Рис. 3.1

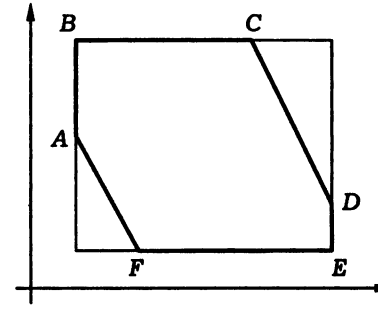


Рис. 3.2

б. Если на множестве точек плоскости (на которой задана некоторая декартова система координат и точка рассматривается как упорядоченная пара действительных чисел) определить отношение порядка так, что $(x,\ y) \leqslant (u,\ v)$ тогда и только тогда, когда $x \leqslant u$ и $y \leqslant v$ (см. пример 1.17 и рис. 1.11), то множество точек любого прямоугольника $[a, b] \times [c, d]$ (с указанным отношением порядка), согласно теоремам 3.10 и 3.11, есть и верхняя и нижняя полурешетка (рис. 3.1). Если же мы „срежем" у этого прямоугольника углы и рассмотрим, например, множество точек многоугольника $ABCDEF$ на рис. 3.2, то это множество не будет ни верхней, ни нижней полурешеткой, так как, скажем, множество $\{A, F\}$ не имеет inf, а множество $\{C, D\}$ не имеет sup. Если мы „срежем" только один угол, левый нижний или правый верхний, то получим верхнюю или нижнюю полурешетку соответственно.

в. На рис. 3.3 изображена диаграмма Хассе множества, не являющегося ни верхней, ни нижней полурешеткой. Действительно, множество $\{d, e\}$ не имеет точной верхней грани, хотя имеет точную нижнюю грань: $\inf\{d, e\} = c$. Аналогично множество $\{a, b\}$ не имеет точной нижней грани, хотя имеет точную верхнюю грань: $\sup\{a, b\} = c$. #

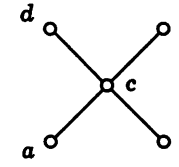


Рис. 3.3

Полурешетка, как полугруппа, может быть *моноидом*, т.е. может иметь *нейтральный элемент*. Нейтральный элемент верхней полурешетки $(L, \vee)$ называют ***нулем полурешетки*** и обозначают $\mathbf{0}$, называя при этом саму полурешетку верхней полурешеткой с нулем. Двойственным образом нейтральный элемент нижней полурешетки $(L, \wedge)$ называют ***единицей полурешетки***, используя обозначение $\mathbf{1}$ и называя эту полурешетку нижней полурешеткой с единицей.

Для нуля полурешетки $(L, \vee, \mathbf{0})$ имеем $a \vee \mathbf{0} = a$ для всякого $a \in L$, т.е. нуль полурешетки есть ее *наименьший элемент*. Двойственным образом единица полурешетки $(L, \wedge, \mathbf{1})$ есть ее *наибольший элемент*.

Пример 3.15. **а.** Прямоугольник $[a, b] \times [c, d]$ (см. пример 3.14.б и рис. 3.1) — одновременно и нижняя полурешетка с единицей, которой служит точка (b, d), и верхняя полурешетка с нулем, которым является точка (a, c). Прямоугольник со „срезанным“ правым верхним углом — нижняя полурешетка, но без единицы полурешетки.

б. Отрезок $[a, b] \subset \mathbb{R}$ (при любых его границах) есть одновременно и нижняя полурешетка с единицей, которой является число b, и верхняя полурешетка с нулем. Им служит число a.

Полуинтервалы $(a, b]$ и $[a, b)$ суть соответственно нижняя полурешетка с единицей и верхняя полурештка с нулем. Заметим, что одновременно первый полуинтервал есть верхняя полурешетка, не имеющая нуля, а второй полуинтервал — нижняя полурешетка без единицы.

Интервал (a, b) есть одновременно и нижняя и верхняя полурешетка, но ни та, ни другая не имеет нейтральных элементов.

в. *Индуктивное упорядоченное множество* в общем случае не является верхней полурешткой с нулем, хотя и имеет наименьший элемент. Дело в том, что двухэлементное подмножество этого множества, состоящее из *несравнимых элементов*, может и не иметь точной верхней грани, поскольку точную верхнюю грань имеет в этом множестве любая неубывающая

последовательность и, как можно показать, любая конечная или счетная *цепь*.

Индуктивное линейно упорядоченное множество является верхней полурешеткой с нулем. Обратное неверно. Так, множество всех неотрицательных вещественных чисел (с естественным числовым порядком) есть верхняя полурешетка с нулем, но ни одна возрастающая последовательность в этом множестве не имеет точной верхней грани. #

Рассмотренные выше примеры показывают, что существуют упорядоченные множества, являющиеся нижней и верхней полурешетками одновременно. Если такая „связка" полурешеток имеет определенные дополнительные свойства, то она называется решеткой.

Определение 3.5. ***Решетка*** — это *алгебра* $\mathcal{L} = (L, \vee, \wedge)$ с двумя *бинарными операциями*, такая, что каждая из алгебр $(L, \vee)$ и $(L, \wedge)$ является полурешеткой и выполняются следующие ***тождества поглощения***:

$$a \vee (a \wedge b) = a, \qquad a \wedge (a \vee b) = a.$$

Операции решетки $\vee$ и $\wedge$ называют ***решеточным объединением*** и ***решеточным пересечением*** соответственно. Операции решетки называют также ***решеточными операциями***.

Другими словами, решетка — это алгебра с двумя бинарными операциями, обозначаемыми $\vee$ и $\wedge$, такими, что выполняются тождества

$$\begin{aligned}
&a \vee (b \vee c) = (a \vee b) \vee c, \quad a \wedge (b \wedge c) = (a \wedge b) \wedge c,\\
&a \vee b = b \vee a, \quad a \wedge b = b \wedge a,\\
&a \vee a = a, \quad a \wedge a = a,\\
&a \vee (a \wedge b) = a, \quad a \wedge (a \vee b) = a.
\end{aligned}$$

Из определения решетки следует, что всякое *симметричное полукольцо* и, значит, любая *булева алгебра* являются решетками, т.е. *сложение* и *умножение* симметричного *полукольца*,

равно как и *булево объединение* и *булево пересечение*, есть примеры решеточных операций. Тогда и булева алгебра $\mathcal{S}_A$ всех подмножеств некоторого множества A, и полукольцо $\mathcal{D}_n$ всех делителей натурального числа n — примеры решеток.

Приведем пример решетки, не являющейся полукольцом.

Пример 3.16. Алгебра $((a, b), \max, \min)$, носитель которой — открытый интервал на числовой прямой, а операции — взятие наибольшего и наименьшего из двух чисел (относительно естественного числового порядка), есть решетка. Однако отсутствие нейтральных элементов по данным операциям не позволяет считать данную алгебру полукольцом. #

Заметим, что решеточные операции в общем случае и не дистрибутивны (хотя в только что рассмотренном примере 3.16 дистрибутивность каждой операции относительно другой имеет место). Конкретные примеры недистрибутивных решеток будут приведены ниже.

Выясним теперь связь между понятием решетки и понятием упорядоченного множества. Каждая решетка, как следует из определения, есть „связка“ двух полурешеток, но априори совсем не очевидно, что естественные порядки этих полурешеток суть взаимно двойственные порядки, т.е. для любых $a, b \in L$ неравенство $a \leqslant b$, равносильное равенству $a \vee b = b$, имеет место если и только если $a \wedge b = a$. Такую связь между решеточными операциями устанавливает следующая теорема.

Теорема 3.12. В любой решетке $\mathcal{L} = (L, \vee, \wedge)$ естественный порядок $\leqslant$ полурешетки $(L, \vee)$ есть порядок, двойственный к естественному порядку полурешетки $(L, \wedge)$, т.е. для любых a, $b \in L$ имеет место равенство

$$a \vee b = b \Leftrightarrow a \wedge b = a,$$

т.е. $a \vee b = \sup\{a, b\}$ и $a \wedge b = \inf\{a, b\}$.

◀ Согласно определению естественного порядка полурешетки, для любых a, $b \in L$ выполняется условие $a \leqslant b \Leftrightarrow a \vee b = b$. Мы

должны доказать, что двойственный порядок $\geqslant$ совпадает с естественным порядком полурешетки $(L, \wedge)$, т.е. нужно доказать, что $a \geqslant b \Leftrightarrow a \wedge b = b$ или, что равносильно, $a \leqslant b \Leftrightarrow a \wedge b = a$. Для этого достаточно показать, что $a \vee b = b \Leftrightarrow a \wedge b = a$.

Имеем: если $a \vee b = b$, то $a \wedge b = a \wedge (a \vee b)$. Согласно второму тождеству поглощения, $a \wedge (a \vee b) = a$, и поэтому $a \wedge b = a$; если же $a \wedge b = a$, то с учетом первого тождества поглощения будем иметь $a \vee b = (a \wedge b) \vee b = b$.

Итак, естественные порядки полурешеток $(L, \wedge)$ и $(L, \vee)$ взаимно двойственные. Так как в силу теоремы 3.10 $\sup\{a, b\} = a \vee b$ для любых $a, b \in L$, то, согласно принципу двойственности, $\inf\{a, b\} = a \wedge b$. ▶

Обратим внимание на использование тождеств поглощения при доказательстве теоремы 3.12. Именно они делают естественные порядки полурешеток $(L, \vee)$ и $(L, \wedge)$ взаимно двойственными.

Таким образом, любая решетка $\mathcal{L} = (L, \vee, \wedge)$ есть в то же время упорядоченное множество $(L, \leqslant)$, где отношение порядка $\leqslant$ совпадает с естественным порядком полурешетки $(L, \vee)$, которая тогда будет верхней полурешеткой. Полурешетка же $(L, \wedge)$ оказывается (относительно этого же отношения порядка) нижней полурешеткой. Первую полурешетку поэтому часто называют верхней полурешткой решетки $\mathcal{L}$, а вторую полурешетку — нижней полурешеткой той же самой решетки. Само же отношение порядка $\leqslant$ называют ***естественным порядком решетки*** $\mathcal{L}$.

Можно доказать утверждение, обратное к теореме 3.12.

Теорема 3.13. Любое упорядоченное множество $\mathcal{L} = (L, \leqslant)$, в котором всякое двухэлементное подмножество имеет точную верхнюю и точную нижнюю грани, т.е. для любых $a, b \in L$ существуют $\sup\{a, b\}$ и $\inf\{a, b\}$, является решеткой в смысле определения 3.5, в которой решеточные операции определяются так:

$$a \vee b = \sup\{a, b\}, \qquad a \wedge b = \inf\{a, b\}.$$

При этом естественный порядок решетки $(L, \sup, \inf)$ совпадает с исходным порядком $\leqslant$.

◀ То, что каждая из алгебр $(L, \sup)$ и $(L, \inf)$ является полурешеткой, равно как и то, что естественный порядок первой полурешетки совпадает с исходным порядком $\leqslant$, а естественный порядок второй полурешетки двойствен к порядку $\leqslant$, следует из теорем 3.10 и 3.11. Остается доказать тождества поглощения. Достаточно доказать одно из них, так как второе будет выполняться в силу принципа двойственности (для упорядоченных множеств).

Итак, докажем, что для любых $a, b \in L$ выполняется равенство

$$\inf\{a, \sup\{a, b\}\} = a. \tag{3.34}$$

Равенство (3.34) следует из того, что для любых элементов $c, d \in L$ из того, что $c \leqslant d$, следует, что $c = \inf\{c, d\}$. Тогда в силу неравенства $a \leqslant \sup\{a, b\}$ имеем (3.34). ▶

Важность последнего результата состоит в том, что полурешетку можно задавать как упорядоченное множество с точными верхними и точными нижними гранями для любой пары элементов. Иными словами, для того чтобы убедиться, что перед нами решетка, достаточно проверить это свойство.

Пример 3.17. а. Рассмотренный в примере 3.14.б прямоугольник $[a, b] \times [c, d]$ есть решетка (см. рис. 3.1), но если мы „срежем“ у этого прямоугольника углы и рассмотрим, например, множество точек многоугольника $ABCDEF$ на рис. 3.2, то это множество уже не будет решеткой, так как, скажем, множество $\{A, F\}$ не имеет inf, а множество $\{C, D\}$ не имеет sup.

б. Не является решеткой и конечное упорядоченное множество, *диаграмма Хассе* которого изображена на рис. 3.3. Здесь $\sup\{a, b\} = c$, но $\inf\{a, b\}$ не существует. Аналогично $\inf\{d, e\} = c$, но $\sup\{d, e\}$ не существует.

в. *Декартов квадрат* множества целых чисел $\mathbb{Z}^2$ — решетка, где упорядоченные пары целых чисел упорядочиваются аналогично точкам плоскости (см. пример 3.17.а). Относительно этого же отношения порядка решеткой будет и любой „целочисленный прямоугольник“ $[a \ldots b] \times [c \ldots d]$, где $[m \ldots n]$ обозначает множество всех целых чисел p, таких, что $m \leqslant p \leqslant n$ $(m \leqslant n)$. Этот „прямоугольник“ и графически выглядит как „решетка“, что видно на рис. 3.4, на котором изображено множество $[1 \ldots 5] \times [1 \ldots 4]$. #

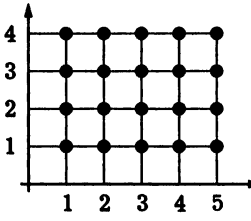


Рис. 3.4

Если верхняя полурешетка решетки $\mathcal{L} = (L, \vee, \wedge)$ есть полурешетка с нулем, то этот нуль является наименьшим элементом решетки $\mathcal{L}$. Тогда ее называют решеткой с нулем, а нуль полурешетки $(L, \vee)$ — ***нулем решетки*** $\mathcal{L}$. Двойственным образом единица нижней полурешетки $(L, \wedge)$, если она существует, является наибольшим элементом решетки $\mathcal{L}$. Эту единицу тогда называют ***единицей решетки*** $\mathcal{L}$, а саму решетку — решеткой с единицей.

Таким образом, решетка с нулем имеет нейтральный элемент по операции решеточного объединения (и ее верхняя полурешетка превращается в моноид), а решетка с единицей имеет нейтральный элемент по операции решеточного пересечения (и моноидом становится уже ее нижняя полурешетка). Кроме того, из теоремы 3.12 следует, что $a \wedge \mathbf{0} = \mathbf{0}$ для любого $a \in L$ (соответственно $a \vee \mathbf{1} = \mathbf{1}$ для любого $a \in L$), т.е. выполняются аналоги аннулирующих свойств в симметричных полукольцах. Симметричное полукольцо является примером решетки с нулем и единицей.

Пример 3.18. На рис. 3.1 прямоугольник $[a, b] \times [c, d]$ является решеткой с нулем и единицей, причем $\mathbf{0} = (a, c)$, $\mathbf{1} = (b, d)$.

Рассмотрим теперь вместо отрезка $[a, b]$ промежуток $A = \{x\colon x \leqslant b\}$ ($\leqslant$ означает здесь естественный числовой порядок). Тогда множество $A \times [c, d]$ при сохранении того же отношения порядка на $\mathbb{R}^2$, что и в примере 3.17, будет решеткой с единицей, равной (b, d). Нуля у этой решетки не будет (если рассматривать, конечно, нерасширенную числовую прямую, точнее, не включать в нее $-\infty$). Если же вместо отрезка $[a, b]$ взять также неограниченный промежуток $B = \{x\colon x \geqslant a\}$, то $B \times [c, d]$ будет решеткой с нулем, равным $(a)c$. Если опять-таки считать, что элемент $+\infty$ не принадлежит $\mathbb{R}$, то эта решетка не будет иметь единицу. #

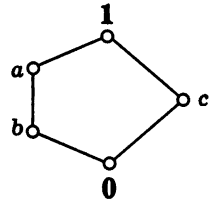


Рис. 3.5

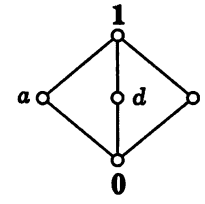


Рис. 3.6

В общем случае в решетке не имеет места дистрибутивность одной из операций относительно другой. На рис. 3.5 и 3.6 приведены диаграммы Хассе решеток, называемых соответственно ***пентагоном*** и ***диамантом***. В каждой из них ни одна из операций не является дистрибутивной относительно другой. Так, например, в первой из них $a \wedge (b \vee c) = a \wedge \mathbf{1} = a$, но $(a \wedge b) \vee (a \wedge c) = b \vee \mathbf{0} = b$. Для второй решетки $a \wedge (b \vee c) = a \wedge \mathbf{1} = a$, тогда как $(a \wedge b) \vee (a \wedge c) = \mathbf{0} \vee \mathbf{0} = \mathbf{0}$.

Определение 3.6. Решетку $\mathcal{L} = (L, \vee, \wedge)$ называют ***дистрибутивной***, если для любых $a, b, c \in L$ имеют место равенства

$$a \vee (b \wedge c) = (a \vee b) \wedge (a \vee c),$$
$$a \wedge (b \vee c) = (a \wedge b) \vee (a \wedge c).$$

Теперь мы можем дать в терминах решеток характеристику симметричных полуколец и булевых алгебр.

Теорема 3.14. Симметричное полукольцо — это дистрибутивная решетка с нулем и единицей.

Булева алгебра — это дистрибутивная решетка с нулем и единицей, в которой для каждого элемента x существует *дополнение*, т.е. такой элемент $\overline{x}$, что $x \vee \overline{x} = \mathbf{1}$ и $x \wedge \overline{x} = \mathbf{0}$.

Доказательство теоремы 3.14 мы не приводим, так как оно является непосредственным следствием определений 3.3, 3.4 и 3.6.

Теория решеток находит многочисленные применения в теории графов и теоретическом программировании. В специальной литературе на базе теории решеток развиты стройная концепция типов данных в языках программирования и теория типов данных*, хорошо изложены общая теория решеток** и ее приложения к теории графов***.

Вопросы и задачи

3.1. Проверив аксиомы, убедиться, что алгебры, приведенные в примере 3.3, являются полукольцами.

3.2. Выяснить, является ли алгебра $([0, 1], \max, \cdot)$ полукольцом.

3.3. В полукольце $\mathcal{S}_{[0,1]}$ (см. пример 3.3.г) решить систему линейных уравнений

$$\begin{cases} x_1 = 0{,}3x_1 + 0{,}5x_2 + 0{,}1x_3 + 0{,}2, \\ x_2 = 0{,}6x_1 + 0{,}8x_2 + 0{,}4x_3 + 0{,}6, \\ x_3 = 0{,}2x_1 + 0{,}4x_2 + 0{,}25x_3 + 0{,}1. \end{cases}$$

3.4. Привести примеры замкнутых полуколец из 4, 8 и 16 элементов.

*См.: *Скотт Д.* Теория решеток, типы данных и семантика

**См.: *Гретцер Г.*

***См.: *Евстигнеев В.А.*

Указание: рассмотреть алгебру $\left(2^M, \cup, \cap, \varnothing, M\right)$, где множество M конечно.

3.5. Доказать теорему 3.9.

3.6. Описать одноэлементную булеву алгебру. Доказать, что булева алгебра одноэлементна тогда и только тогда, когда в ней $\mathbf{0} = \mathbf{1}$.

3.7. Доказать, что полукольцо $\mathcal{D}_m$ является булевой алгеброй тогда и только тогда, когда m есть произведение попарно различных простых чисел.

3.8. Пусть $\mathrm{B} = (B, \vee, \wedge, {}^{-}, \mathbf{0}, \mathbf{1})$ — булева алгебра. Определим на носителе B операции $\oplus$ и $\cdot$ так:

$$x \oplus y = (x \wedge \overline{y}) \vee (\overline{x} \wedge y), \quad x \cdot y = x \wedge y.$$

Доказать, что $\mathcal{R}_{\mathbb{B}} = (B, \oplus, \cdot, \mathbf{0}, \mathbf{1})$ — булево кольцо. Доказать, что $\mathcal{R}_{\mathbb{B}} = \mathbb{Z}_2$.

3.9. Пусть $\mathcal{R} = (R, \oplus, \cdot, \mathbf{0}, \mathbf{1})$ — булево кольцо (см. задачу 2.12). Определим на его носителе R операции $\vee$, $\wedge$ и ${}^{-}$ так:

$$x \vee y = x \oplus y \oplus xy, \quad x \wedge y = x \cdot y, \quad \overline{x} = x \oplus \mathbf{1}.$$

Доказать, что $\mathrm{B}_{\mathcal{R}} = (R, \vee, \wedge, {}^{-}, \mathbf{0}, \mathbf{1})$ — булева алгебра. Показать, что $\mathrm{B}_{\mathbb{Z}_2} = \mathbb{B}$.

3.10. Показать, что для любых булевой алгебры B и булева кольца $\mathcal{R}$ (см. задачу 2.12) $\mathcal{R}_{\mathrm{B}_{\mathcal{R}}} = \mathcal{R}$ и $\mathrm{B}_{\mathcal{R}_{\mathrm{B}}} = \mathrm{B}$.

3.11. На носителе произвольного (т.е. не обязательно идемпотентного) полукольца $\mathcal{S} = (S, +, \cdot, \mathbf{0}, \mathbf{1})$ определим бинарное отношение $\preceq$ так, что $x \preceq y \Leftrightarrow (\exists z)(y = x + z)$. Является ли это бинарное отношение порядком или предпорядком? Как интерпретируется это отношение для кольца?

3.12. Пусть полукольцо $\mathcal{S} = (S, +, \cdot, \mathbf{0}, \mathbf{1})$ таково, что в его аддитивном моноиде справедлив закон сокращения, т.е. для

любых x, y, $a \in S$ из равенства $x + a = y + a$ следует равенство $x = y$, а также для любых a, $b \in S$ из того, что $a + b = 0$, следует $a = b = 0$. Доказать, что тогда бинарное отношение $\preceq$, введенное в задаче 3.11, является порядком. Привести пример такого полукольца. При каких условиях полукольцо с такими свойствами будет идемпотентным?

3.13. Доказать, что в полукольце бинарных отношений на n-элементном множестве итерация любого бинарного отношения равна сумме всех его степеней, начиная с нулевой и кончая n-й, т.е. $\rho^* = \bigcup_{k=0}^{n} \rho^k$. Доказать, что аналогичное свойство справедливо для полукольца квадратных матриц над полукольцом $\mathcal{B}$.

3.14. Доказать подробно, что метод последовательного исключения неизвестных при решении систем линейных уравнений в замкнутых полукольцах (или в полукольцах с итерацией) действительно дает наименьшее решение системы.

Указание: воспользоваться методом математической индукции и доказать, что на каждом шаге прямого хода метода Гаусса мы получаем очередную компоненту наименьшего решения; при этом использовать непрерывность (и, следовательно, монотонность) операций сложения и умножения.

3.15. Доказать, что идемпотентность решеточных операций вытекает из тождеств поглощения.

3.16. Доказать, что алгебра $(A, \vee, \wedge)$ является решеткой тогда и только тогда, когда $(A, \vee)$ и $(A, \wedge)$ — полурешетки, и равенство $a = a \wedge b$ равносильно равенству $b = a \vee b$.

3.17. Пусть алгебры $(A, \wedge, \vee_1)$ и $(A, \wedge, \vee_2)$ (с одним и тем же носителем) являются решетками. Доказать, что тогда операции $\vee_1$ и $\vee_2$ совпадают.

3.18. Нарисовать диаграммы Хассе всех решеток, состоящих не более чем из шести элементов.

4. АЛГЕБРАИЧЕСКИЕ СИСТЕМЫ

При решении многих математических задач (задач дискретной математики, в частности) используются более сложные структуры, чем „множества с операциями“. Например, изучая множество действительных чисел, мы имеем дело не только с операциями над числами, но и с определенными *отношениями* на множестве действительных чисел, такими, как, скажем, *естественный числовой порядок*. Важейшая структура — *упорядоченное множество* — вообще определяется без каких-либо операций. Таким образом, мы приходим к необходимости изучения „множеств с операциями и отношениями“, в частности „множеств с отношениями“ (примерами которых служат упорядоченное множество, *индуктивное упорядоченное множество*, множество, на котором задано некоторое *отношение предпорядка*, *эквивалентности* и т.п.). Это приводит нас к понятиям „алгебраической системы“ и „модели“.

В этой главе изучаются некоторые результаты общей теории алгебраических систем, в том числе обобщаются конструкции, которые были введены в двух предшествующих главах. Материал этой главы будет использован затем в теории графов, *булевых функций* и языков.

4.1. Модели и алгебры

Алгебраическая система — это *упорядоченная тройка* $\mathcal{A} = (A, \Omega, \Pi)$, где A — некоторое множество, называемое ***носителем*** алгебраической системы; Ω — некоторое множество операций на A, Π — некоторое множество *отношений* на A. *Упорядоченную пару* (Ω, Π) называют ***сигнатурой*** алгебраической системы.

Алгебраическую систему называют ***конечной***, если ее носитель — конечное множество.

Обозначая через $\Omega^{(n)}$ подмножество всех *n-арных операций* в Ω, получим $\Omega = \bigcup\limits_{n \geqslant 0} \Omega^{(n)}$. Точно так же $\Pi = \bigcup\limits_{n \geqslant 1} \Pi^{(n)}$, где $\Pi^{(n)}$ — подмножество всех *n-арных отношений*.

Алгебраическая система, у которой множество Π отношений пусто ($\Pi = \varnothing$), есть не что иное, как *Ω-алгебра* (см. **2**). Алгебраическую систему, у которой пусто множество операций ($\Omega = \varnothing$), называют ***моделью***.

Замечание 4.1. Докажем, что n-арная операция на множестве A есть частный случай отношения на этом множестве. Действительно, если $\omega\colon A^n \to A$ — n-арная операция, то определим $(n+1)$-арное отношение $\rho_\omega \subseteq A^{n+1}$ так, что *кортеж* $(a_1, \ldots, a_n, b) \in \rho_\omega$ тогда и только тогда, когда $b = \omega(a_1, \ldots, a_n)$. Понятно, что это отношение *функционально по* $(n+1)$-*й компоненте*. С учетом этого любая алгебраическая система может рассматриваться как модель. #

Мы рассмотрели много примеров алгебр (см. **2** и **3**). В некоторых из них были введены отношения, например *естественный порядок полукольца*, *естественный порядок булевой алгебры*, *естественный порядок решетки*. Эти отношения разумно ввести в сигнатуру и рассматривать указанные алгебры как алгебраические системы, тем более что указанные отношения определяли важнейшие свойства названных алгебр.

Пример 4.1. а. Одной из основных моделей в математике является *упорядоченное множество*. Сигнатура этой модели состоит из единственного *отношения порядка*. Важным частным случаем служит *индуктивное упорядоченное множество*.

б. Как уже было замечено, любое полукольцо, в частности замкнутое полукольцо, является алгебраической системой, сигнатура которой помимо операций полукольца содержит отношение естественного порядка полукольца.

Рассмотрим теперь некоторое *поле* $\mathcal{F} = (F, +, \cdot, \mathbf{0}, \mathbf{1})$, множество всех ненулевых элементов которого разбито на подмножества P и N. Другими словами, по определению полагаем, что для каждого $a \in F$ выполняется в точности одно из трех условий: $a \in P$, $a = \mathbf{0}$ или $a \in N$. Элементы P назовем (условно) положительными, а элементы N — отрицательными элементами данного поля. При этом, по определению, выполняются следующие условия:

1) для каждого $a \in F$ a отрицательно тогда и только тогда, когда $-a$ положительно, т.е. $-a \in P$;

2) если $a, b \in P$, то $a + b \in P$ и $a \cdot b \in P$.

Введенные условия вполне естественны: первое означает, что элемент, противоположный к отрицательному, является положительным и наоборот, а второе — что сумма и произведение положительных элементов положительны.

Введем теперь на множестве F *бинарное отношение* $<$ так, что $a < b \Leftrightarrow b - a \in P$ (читается: a „меньше“ b, по определению, если *разность* $b - a$ есть положительный элемент). Естественно, полагаем, что $a \leqslant b$ означает $a < b$ или $a = b$. Можно показать, что введенное таким образом отношение $\leqslant$ на носителе поля $\mathcal{F}$ является отношением *линейного порядка*, т.е. для любых двух элементов $a, b \in F$ или $a \leqslant b$, или $b \leqslant a$.

Поле вместе с отношением порядка, введенным указанным образом, называют ***упорядоченным полем***. Таким образом, упорядоченное поле можно рассматривать как алгебраическую систему $\mathcal{F}_O = (F, +, \cdot, \mathbf{0}, \mathbf{1}, \leqslant)$, в которой алгебра $\mathcal{F} = (F, +, \cdot, \mathbf{0}, \mathbf{1})$ является полем, а отношение порядка $\leqslant$ определено так, как сказано выше.

Пусть, кроме этого, отношение порядка в упорядоченном поле обладает следующим ***свойством непрерывности***: каковы бы ни были непустые множества $A \subset F$ и $B \subset F$, у которых для любых двух элементов $a \in A$ и $b \in B$ выполняется $a \leqslant b$, существует такой элемент α, что для всех $a \in A$ и $b \in B$ выполняется двойное неравенство $a \leqslant \alpha \leqslant b$. Тогда получаем алгебраиче-

скую систему, называемую ***непрерывным упорядоченным полем***. Важнейший пример непрерывного упорядоченного поля — *поле действительных чисел*.

Заметим, что *поле рациональных чисел*, являясь упорядоченным полем, уже не будет непрерывным. Это вытекает из того, что можно построить такие два *собственных подмножества* $A, B \subset \mathbb{Q}$, что для всех $a \in A$ и для всех $b \in B$ будет иметь место $a < b$, но нельзя найти такое рациональное число r, чтобы выполнялось $(\forall a \in A)(\forall b \in B)(a \leqslant r \leqslant b)$. Такими двумя подмножествами A и B в множестве рациональных чисел могут быть, например, $B = \{q\colon q^2 > 2\}$, $A = \mathbb{Q} \setminus B$. Дело в том, что, как можно убедиться, не существует *наибольшего* рационального числа в множестве A. В множестве же $\mathbb{R}$ наибольшее из всех чисел, квадрат которых не больше 2, существует и равно $\sqrt{2}$. #

Две алгебраические системы

$$\mathcal{A}_1 = (A_1, \Omega_1, \Pi_1) \text{ и } \mathcal{A}_2 = (A_2, \Omega_2, \Pi_2)$$

называют ***однотипными***, если между множествами операций Ω_1 и Ω_2 существует *взаимно однозначное соответствие*, отображающее $\Omega_1^{(n)}$ в $\Omega_2^{(n)}$, $n \geqslant 0$, а между множествами отношений Π_1 и Π_2 можно установить взаимно однозначное соответствие, при котором $\Pi_1^{(n)}$ соответствует $\Pi_2^{(n)}$, $n \geqslant 0$.

Таким образом, для однотипных алгебраических систем $\mathcal{A}_1 = (A_1, \Omega_1, \Pi_1)$ и $\mathcal{A}_2 = (A_2, \Omega_2, \Pi_2)$ любой n-арной операции из Ω_1 (любому n-арному отношению из Π_1) может быть однозначно сопоставлена n-арная операция из Ω_2 (n-арное отношение из Π_2).

Например, алгебраическая система $(\mathbb{Z}, -, +, \cdot, \leqslant)$ с операциями $-$ (унарный минус — переход к противоположному числу), $+$ (сложения), $\cdot$ (умножения) и отношением $\leqslant$ естественного числового порядка однотипна с алгебраической системой $(\mathbb{Q}, -, +, \cdot, \leqslant)$ с теми же операциями, а также с алгебраической системой $\left(2^M, \overline{\ }, \cup, \cap, \subseteq\right)$ (для некоторого множества M) с операциями дополнения, объединения, пересечения множеств

и *отношением включения*. В первом случае операции и отношения, заданные на разных множествах (целых и рациональных чисел), обозначены одинаковыми символами; во втором случае однотипные алгебраические системы имеют разные обозначения операций и отношений.

В то же время две модели (A, ρ) и (B, σ), в которых ρ — n-арное отношение на A, а σ — m-арное отношение на B, при $n \neq m$ не будут однотипными.

Зачастую, если это не вредит точности, соответствующие друг другу операции и отношения однотипных алгебраических систем будем обозначать одинаковыми символами.

4.2. Подсистемы

Выше (см. **2**) введены понятия *подгруппы*, *подкольца*, *подполя*, которые можно объединить в рамках общего понятия *подсистемы* произвольной *алгебраической системы*, а также понятия *подалгебры* произвольной *Ω-алгебры*.

Пусть $\mathcal{A} = (A, \Omega, \Pi)$ — произвольная алгебраическая система и $B \subseteq A$ — непустое множество.

Множество B называют ***замкнутым относительно операций*** из Ω (***Ω-замкнутым множеством***), если результат применения любой *n-арной операции* из Ω к любым элементам из B принадлежит B, т.е. для любой n-арной операции ω и любых элементов $a_1, \ldots, a_n \in B$ элемент $\omega(a_1, \ldots, a_n) \in B$.

Например, в полугруппе $(\mathbb{N}, +)$ подмножество четных чисел замкнуто относительно операции сложения, а подмножество нечетных чисел не замкнуто. В *кольце целых чисел* подмножество натуральных чисел замкнуто относительно операциий сложения и умножения, но не замкнуто относительно операции взятия противоположного элемента.

Алгебраическую систему $\mathcal{B} = (B, \Omega, \Pi|_B)$, где $B \subseteq A$, называют ***подсистемой алгебраической системы*** $\mathcal{A}$, если B Ω-замкнуто и $\Pi|_B$ есть множество *ограничений* на B всех *отношений* из Π: $\Pi|_B = \{p|_B \colon p \in \Pi\}$.

Очевидно, что алгебраические системы $\mathcal{A}$ и $\mathcal{B}$ *однотипны*, и часто вместо $\Pi|_B$ будем в таком случае писать просто Π.

Если $\mathcal{A}$ — алгебра, то любую ее подсистему называют ее ***подалгеброй*** (точнее, ***Ω-подалгеброй***).

Замечание 4.2. В определении Ω-подалгебры требуется лишь замкнутость относительно операций из Ω. Если же мы хотим, чтобы при переходе к подалгебре „наследовались“ какие-либо специальные свойства операций исходной алгебры, то это нужно специально оговаривать. Именно так мы и поступали, определяя понятия подгруппы, подкольца, подполя и т.п. Впрочем, подгруппу можно определить и через свойство замкнутости, но лишь в том случае, если в сигнатуру *группы* включить не только одну *бинарную операцию* „умножения“ (которая обладает специальными „групповыми свойствами“, см. **2.2**), но также *унарную операцию* взятия *обратного элемента* и *нульарную операцию — единицу группы*.

Аналогично, исключительно через требование замкнутости, можно определить понятие подмоноида. Следовательно, таким образом можно определить и подкольцо.

Сложнее обстоит дело с *телом* и *полем*. Мы не можем определить поле как алгебру с сигнатурой $\{+, *, -, ^{-1}, \mathbf{0}, \mathbf{1}\}$, где операция $-$ есть операция вычисления *противоположного элемента* (обратного по сложению), а операция $^{-1}$ — операция вычисления обратного элемента по умножению, так как последняя операция есть *частичное отображение* и не определена для элемента $\mathbf{0}$. Поэтому она не может быть введена в сигнатуру алгебры, по определению содержащей только всюду определенные операции.

Обратим внимание и на то, что переходя к Ω-замкнутому подмножеству, мы можем получить алгебру как обогащенную новыми свойствами операций сигнатуры Ω, так и утратившую некоторые из свойств. Например, *моноид* $(\mathbb{N}_0, +, 0)$ будет только *подмоноидом* группы $(\mathbb{Z}, +, 0)$ (но не подгруппой), а подмоноид биекций в *симметрическом моноиде* некоторого

бесконечного множества будет уже группой (это не подгруппа, а именно подмоноид, являющийся группой!). #

В следующей теореме сформулировано простое, но очень важное свойство замкнутых подмножеств.

Теорема 4.1. Непустое пересечение произвольного *семейства* Ω-замкнутых *подмножеств* Ω-замкнуто.

◀ Для простоты рассмотрим доказательство для пересечения двух Ω-замкнутых подмножеств.

Пусть в алгебре (A, Ω) Ω-замкнутые подмножества B_1 и B_2 имеют непустое пересечение. Тогда для любого $n \geqslant 0$, а также любых (не обязательно различных) $a_1, \ldots, a_n \in B_1 \cap B_2$ элемент $\omega(a_1, \ldots, a_n)$, какова бы ни была операция $\omega \in \Omega^{(n)}$, снова будет принадлежать пересечению $B_1 \cap B_2$, так как в силу замкнутости каждого из множеств B_1 и B_2 одновременно $\omega(a_1, \ldots, a_n) \in B_1$ и $\omega(a_1, \ldots, a_n) \in B_2$. ▶

Рассмотрим алгебру $\mathcal{A} = (A, \Omega)$ и подмножество $B \subseteq A$, не обязательно Ω-замкнутое.

Из теоремы 4.1 следует, что существует Ω-замкнутое подмножество, совпадающее с пересечением всех Ω-замкнутых подмножеств, содержащих B. Его называют ***замыканием*** подмножества B ***относительно операций*** из Ω (или ***Ω-замыканием*** подмножества B) и обозначают $[B]_\Omega$. Хотя бы одно Ω-замкнутое подмножество, содержащее B, обязательно найдется — весь носитель A. В том случае, когда $[B]_\Omega = A$, подмножество B называют ***системой образующих*** алгебры $\mathcal{A} = (A, \Omega)$, а ее саму называют ***алгеброй, порожденной множеством*** B. Алгебру, которая имеет конечную систему образующих, называют ***конечно порожденной***.

Замечание 4.3. Определение замыкания можно представить и в несколько иной форме, которая содержательно ассоциируется с некоторой процедурой построения множества $[B]_\Omega$ по шагам.

Определим семейство множеств $(B_i)_{i\geqslant 0}$, полагая $B_0 = B$, а

$$B_{i+1} = B_i \cup \left\{x\colon x = \omega(b_1, \ldots, b_n), n \geqslant 0, \omega \in \Omega^{(n)}, b_1, \ldots, b_n \in B_i\right\}.$$

Таким образом, множество B_1 состоит из всех элементов $B_0 = B$, и к ним добавляются все элементы, которые могут быть получены как результат применения операций сигнатуры Ω к *аргументам операции* из B_0. Множество B_2 точно так же содержит все элементы множества B_1 плюс все результаты применения операций из Ω к аргументам из B_1 и т.д. По определению, $B = B_0 \subseteq B_1 \subseteq B_2 \subseteq \ldots \subseteq B_i \subseteq B_{i+1} \subseteq \ldots$, т.е. для любого $i \geqslant 0$ имеют место включения $B_i \subseteq B_{i+1}$ и $B_i \subseteq [B]_\Omega \subseteq A$. Можно показать, что $[B]_\Omega = \bigcup\limits_{i\geqslant 0} B_i$.

Замкнутость B означает с точки зрения такого определения, что все множества B_i совпадают с множеством B. Кроме того, может оказаться, что процесс образования множеств B_i „оборвется на некотором шаге“, т.е. найдется такое i, что $B_{i+1} = B_i$. Тогда $B_i = [B]_\Omega$.

Для конечной алгебры описанную выше процедуру можно рассматривать как алгоритм построения замыкания исходного множества (при том, что каждой операции сопоставлен некий алгоритм ее вычисления). На первом шаге алгоритма в замыкание $[B]_\Omega$ помещают все элементы множества B, а затем применяют операции сигнатуры Ω к исходным и вновь получаемым элементам до тех пор, пока не перестанут появляться новые элементы.

Иначе это можно описать так:

1) все элементы множества B_0 считаются и элементами замыкания $[B]_\Omega$;

2) каковы бы ни были элементы $b_1, \ldots, b_n$, относительно которых известно, что они принадлежат $[B]_\Omega$ (т.е. какому-то множеству B_i из определенного выше семейства), к имеющимся элементам замыкания $[B]_\Omega$ добавляют все элементы $\omega(b_1, \ldots, b_n)$ для произвольной n-арной операции ω сигнатуры Ω.

Никакие другие элементы, кроме тех, что могут быть получены рассмотренным выше способом, замыканию $[B]_\Omega$ не принадлежат.

Образно говоря, B — это „детали конструктора“, а $[B]_\Omega$ — все, что можно собрать из этих деталей по некоторым заранее оговоренным „правилам сборки“ (каковыми являются операции сигнатуры Ω). #

Рассмотрим примеры построения Ω-замыкания.

Пример 4.2. а. В алгебре $(\mathbb{N}, +)$ возьмем одноэлементное множество $B = \{1\} = B_0$. Тогда $B_1 = \{1, 2\}$, $B_2 = \{1, 2, 3, 4\}$, $B_3 = \{1, 2, 3, 4, 5, 6, 7, 8\}$ и т.д. Множество B_i при $i \geqslant 1$ состоит из всех сумм вида $m + n$, где $m, n \in B_{i-1}$. Несложно заметить, что $B_i = \{1, 2, \ldots, k, \ldots, 2k\}$, где $k = 2^{i-1}$, $i \geqslant 0$. Ясно, что в данном случае $\bigcup\limits_{i \geqslant 0} B_i = \mathbb{N}$, и, таким образом, множество $\{1\}$ является системой образующих этой алгебры.

б. В *мультипликативной группе вычетов* по модулю 23 (т.е. в группе $\mathbb{Z}_{23}^*$) построим замыкание множества $B = \{3\}$, полагая, что сигнатура группы состоит из единственной операции умножения (по модулю 23).

Поскольку в этой алгебре сигнатура состоит из одной операции, а исходное множество B также одноэлементно, то замыкание B будет состоять из всех степеней элемента 3. Итак, для построения замыкания множества B в данном случае достаточно вычислять последовательно степени (по модулю 23) элемента 3. Имеем

$$3^2 = 9, \quad 3^3 = 4, \quad 3^4 = 4 \cdot 3 = 12, \quad 3^5 = 12 \cdot 3 = 13,$$
$$3^6 = 13 \cdot 3 = 16, \quad 3^7 = 16 \cdot 3 = 2, \quad 3^8 = 2 \cdot 3 = 6,$$
$$3^9 = 6 \cdot 3 = 18, \quad 3^{10} = 18 \cdot 3 = 8, \quad 3^{11} = 8 \cdot 3 = 1.$$

Так как получена единица, то „круг замкнулся“, и тем самым вычислено замыкание множества $\{3\}$. Заметим, что в этом случае множество B_1 состоит из всех степеней тройки, начиная с первой и кончая второй, множество B_2 — из всех степеней, начиная с первой и кончая четвертой, множество B_3 — из

всех степеней с первой по восьмую, а множество B_4 — из всех степеней с первой по 16-ю, но поскольку начиная с 12-й степени элементы повторяются, т.е. $3^{12} = 3$, $3^{13} = 3^2$, $3^{14} = 3^3$, $3^{15} = 3^4$ и, наконец, $3^{16} = 3^5$, то уже множество B_5 совпадет с множеством B_4, так что в данном случае $[B]_{\{\cdot\}} = B_4$.

Порожденная множеством $\{3\}$ группа совпала с *циклической подгруппой* группы $\mathbb{Z}_{23}$ с *образующим элементом* 3. Этот результат легко обобщить, доказав, что для произвольной конечной группы $\mathcal{G} = (G, \cdot)$, рассматриваемой как алгебра с сигнатурой, состоящей только из операции умножения, ее циклическая подгруппа с образующим элементом a совпадает с подгруппой, порожденной множеством $\{a\}$. #

Циклическая группа есть один из важнейших примеров конечно порожденной алгебры.

В этой связи обратим внимание на одну тонкость. Если $\mathcal{G}$ — циклическая группа с образующим элементом a, то ее, вообще говоря, нельзя рассматривать как алгебру с системой образующих $\{a\}$. Все зависит от конкретной сигнатуры группы. Действительно, если в сигнатуру группы включить только умножение, то для бесконечной циклической группы $\mathbf{1} \neq a^n$ для любого положительного n. Поэтому замыкание множества $\{a\}$ относительно умножения не содержит единицу. Если же сигнатуру группы как алгебры дополнить унарной операцией взятия обратного, т.е. возведения в степень -1, то циклическая группа с образующим элементом a будет алгеброй с системой образующих $\{a\}$. При таком подходе *аддитивная группа целых чисел*, рассматриваемая как алгебра $(\mathbb{Z}, +, -, 0)$, есть бесконечная циклическая группа, порожденная множеством $\{1\}$.

Пример 4.3. а. Алгебра $(\mathbb{Z}, \cdot, 1)$ (*мультипликативный моноид кольца* целых чисел) не является конечно порожденной. Действительно, в этом моноиде в систему образующих необходимо включить все простые числа, поскольку ни одно из них нельзя представить как произведение других чисел. Но множество простых чисел бесконечно.

б. Любая *конечная алгебра* будет, разумеется, и конечно порожденной. В частности, любое *кольцо вычетов* по модулю k (*поле вычетов* при простом k) — конечно порожденная алгебра, даже если в сигнатуре нет операции нахождения обратного элемента.

в. *Свободный моноид*, порожденный конечным множеством (*алфавитом*) A есть конечно порожденная (и бесконечная) алгебра, система образующих которой равна $A \cup \{\lambda\}$, где λ — *пустой кортеж*.

г. Хорошим примером замыкания служит линейная оболочка заданного множества векторов произвольного линейного пространства [IV]. Как известно, линейная оболочка V множества векторов $x_1, \ldots, x_m$ линейного пространства L есть множество всех линейных комбиниций вида $\sum_{i=1}^{m} \alpha_i x_i$, где $x_i \in V$, $i = \overline{1, m}$, $m \geqslant 1$. Линейная оболочка замкнута относительно операций сложения векторов и умножения вектора на число, так как линейная оболочка множества векторов является линейным подпространством. Более того, линейная оболочка V множества векторов $x_1, \ldots, x_m$ — это наименьшее (относительно отношения включения множеств) замкнутое множество, содержащее заданное множество векторов, поскольку любое замкнутое множество, содержащее векторы $x_1, \ldots, x_m$, содержит и все их линейные комбинации, т.е. включает в себя V. Отметим, что конечномерное линейное пространство — конечно порожденная алгебра, так как оно является линейной оболочкой любого из своих базисов.

4.3. Конгруэнции и фактор-системы

В этой главе нам будет удобно использовать „бесскобочную“ запись для обозначения *результата применения n-арной операции* ω к элементам $a_1, \ldots, a_n$ и писать $a_1 \ldots a_n \omega$ вместо $\omega(a_1, \ldots, a_n)$.

Отношение эквивалентности ρ на *носителе алгебраической системы* $\mathcal{A}$ называют ***конгруэнцией*** на алгебраической системе $\mathcal{A}$, если выполняются условия:

1) для любой n-арной ($n \geqslant 1$) операции ω и любых элементов $a_1, \ldots, a_n, b_1, \ldots, b_n \in A$ из того, что $a_i \rho b_i$ для каждого $i = \overline{1, n}$, следует $(a_1 \ldots a_n \omega) \rho (b_1 \ldots b_n \omega)$;

2) для любого n-арного ($n \geqslant 1$) *отношения* π и для любых элементов $a_1, \ldots, a_n, b_1, \ldots, b_n \in A$ из того, что $a_i \rho b_i$ для каждого $i = \overline{1, n}$ и $(a_1, \ldots, a_n) \in \pi$, следует $(b_1, \ldots, b_n) \in \pi$.

Первое условие означает, что результаты применения любой операции из Ω к попарно эквивалентным аргументам должны быть эквивалентными, а второе — что любое отношение из Π содержит или не содержит кортеж $(b_1, \ldots, b_n)$ независимо от того, какие именно элементы b_i выбираются в соответствующем *классе эквивалентности по отношению* ρ.

Пример 4.4. а. Рассмотрим $(\mathbb{R}, +, \cdot, 0, 1)$ — *поле действительных чисел.* Докажем, что отношение *равенства по модулю* 1 (см. пример 1.14) не является конгруэнцией на этом поле, но является конгруэнцией на $(\mathbb{R}, +, \mathbf{0})$ — его *аддитивной группе*, т.е. на *аддитивной группе действительных чисел.*

Докажем сначала второе утверждение.

Пусть $a = b \pmod 1$ и $c = d \pmod 1$. Тогда числа $a - b$ и $c - d$ являются целыми. Следовательно, и их сумма

$$(a - b) + (c - d) = (a + c) - (b + d) \quad —$$

тоже целое число, т.е. $a + c = b + d \pmod 1$. Это и означает, что отношение равенства по модулю 1 является конгруэнцией на аддитивной группе действительных чисел.

Пусть, как и выше, $a = b \pmod 1$ и $c = d \pmod 1$. Если бы (для любых a, b, c, d) отсюда следовало, что $a \cdot c = b \cdot d \pmod 1$, то тогда дробная часть $a \cdot c$ всегда совпадала бы с дробной частью $b \cdot d$. Но каждое число равно по модулю 1 своей дробной части. Следовательно, тогда дробная часть произведения любых двух

чисел должна была бы равняться произведению дробных частей этих чисел. Простой пример, приведенный ниже, показывает, что это не так.

При $b = a = 1{,}1$ имеем $a = 0{,}1 \pmod 1$, $b = 0{,}1 \pmod 1$ и $0{,}1^2 = 0{,}01$. Так как $ab = 1{,}1^2 = 1{,}21$, то $ab = 0{,}21 \pmod 1$. Это и означает, что равенство по модулю 1 не есть конгруэнция на поле действительных чисел $(\mathbb{R}, +, \cdot, 0, 1)$.

б. Пусть $(\mathbb{Z}, +, \cdot, 0, 1)$ — кольцо целых чисел. Отношение *равенства по модулю* k, введенное в разд. **2.3**, будет конгруэнцией на данном кольце.

Действительно, пусть $m \equiv n \pmod k$ и $r \equiv s \pmod k$. Тогда существуют такие целые числа l_1 и l_2, что

$$m - n = l_1 \cdot k, \qquad r - s = l_2 \cdot k. \tag{4.1}$$

Складывая уравнения системы (4.1), получаем

$$m + r - (n + s) = (l_1 + l_2)k,$$

т.е. $m + r \equiv n + s \pmod k$. Умножая первое уравнение системы (4.1) на r, второе — на n и складывая результаты, получаем

$$mr - ns = (l_1 r + l_2 s)k,$$

т.е. $mr \equiv ns \pmod k$, что и доказывает сформулированное выше утверждение.

в. Рассмотрим алгебраическую систему $(\mathbb{Z}, +, \cdot, 0, 1, \leqslant)$, образованную из кольца целых чисел добавлением *отношения* $\leqslant$ (*естественного числового порядка*). Тогда равенство по модулю k уже не будет конгруэнцией на данной алгебраической системе. Действительно, если, скажем, a и b при делении на k дают один и тот же остаток l, а c и d — один и тот же остаток p, то из $a \leqslant c$ не следует, вообще говоря, что $b \leqslant d$. Например, $5 = 17 \pmod 4$, $6 = 10 \pmod 4$, $5 \leqslant 6$, но $17 > 10$. Таким образом, отношение равенства целых чисел по модулю k не „сохраняет“ отношения $\leqslant$, т.е. справедливость неравенства $a \leqslant b$ зависит от

того, какие элементы a и b в соответствующих классах эквивалентности выбраны.

г. Пусть в *линейном пространстве* L фиксировано *линейное подпространство* V. Рассмотрим L как *модуль* над полем действительных чисел $(\mathbb{R}, +, \cdot, 0, 1)$. На множестве векторов L зададим отношение $\sim_V$ так: $a \sim_V b \Leftrightarrow a - b \in V$. Нетрудно показать, что это отношение эквивалентности. Далее, если $a \sim_V b$ и $c \sim_V d$, то $a + c - (b + d) = (a - b) + (c - d) \in V$, поскольку каждое слагаемое в последней сумме есть вектор из подпространства V. Для произвольного действительного α из $a \sim_V b$ следует, что $\alpha a \sim_V \alpha b$, так как $\alpha a - \alpha b = \alpha(a - b) \in V$. Таким образом, введенное отношение есть конгруэнция на линейном пространстве L.

Напомним, что множество векторов линейного пространства по сложению есть *абелева группа*. Следовательно, рассмотренное отношение эквавалентности $\sim_V$ есть конгруэнция на этой группе. Покажем, что указанную конгруэнцию можно распространить на произвольную абелеву группу. Пусть $\mathcal{G} = (G, +, \mathbf{0})$ — некоторая абелева группа, а $\mathcal{H} = (H, +, \mathbf{0})$ — произвольная *подгруппа* группы $\mathcal{G}$. Зададим отношение $\sim_H$ так: $a \sim_H b \Leftrightarrow a - b \in H$. Рассуждая так же, как и в случае множества векторов линейного пространства, можно показать, что отношение $\sim_H$ является конгруэнцией.

д. Пусть $\mathcal{A} = (A, \leqslant)$ и $\mathcal{B} = (B, \preceq)$ — *упорядоченные множества*. Зададим отображение $f\colon A \to B$ так, чтобы оно было *монотонно*, т.е. чтобы для любых a, $b \in A$ из $a \leqslant b$ следовало $f(a) \preceq f(b)$. Введем отношение эквивалентности $\sim_f$ на A (см. **1.7**), положив $a \sim_f b \Leftrightarrow f(a) = f(b)$. Выясним, будет ли это отношение конгруэнцией на *модели* $\mathcal{A} = (A, \leqslant)$.

Пусть $a_1 \sim_f b_1$, $a_2 \sim_f b_2$ и $a_1 \leqslant a_2$. Тогда в силу монотонности отображения f имеем $f(a_1) \preceq f(a_2)$, а так как $f(a_1) = f(b_1)$ и $f(a_2) = f(b_2)$, то и $f(b_1) \preceq f(b_2)$. Отсюда, однако, нельзя в общем случае сделать вывод, что $b_1 \leqslant b_2$. На рис. 4.1 $f(2) \preceq f(4)$, но элементы 2 и 4 не сравнимы. Данное отображение (как

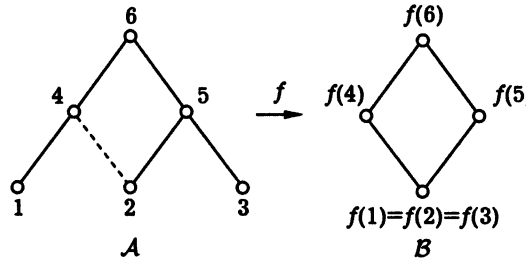


Рис. 4.1

нетрудно понять, монотонное) не будет конгруэнцией хотя бы потому, что $1 \sim_f 2$, но $1 \leqslant 4$, а 2 и 4 не сравнимы.

Если же отображение f таково, что $a \leqslant b \Leftrightarrow f(a) \preceq f(b)$, то $\sim_f$ будет конгруэнцией. Например, если на диаграмме Хассе для множества $\mathcal{A}$ на рис. 4.1 добавить „ребро“, соединяющее элемент 2 с элементом 4 (см. штриховую линию), т.е. считать, что $2 < 4$, то можно будет получить конгруэнцию $\sim_f$ на множестве A. #

Согласно определению конгруэнции, на алгебраической системе $\mathcal{A}$ классы эквивалентности $[a_1]_\rho, \ldots, [a_n]_\rho$ вместе с любой n-арной ($n \geqslant 1$) операцией ω однозначно определяют класс эквивалентности элемента $[a_1 \ldots a_n \omega]_\rho$. Другими словами, для любых элементов $x_1 \in [a_1]_\rho, \ldots, x_n \in [a_n]_\rho$ класс эквивалентности элемента $x_1 \ldots x_n \omega$ зависит только от классов эквивалентности элементов x_i, $i = \overline{1, n}$, но не зависит от выбора элемента в классе.

Таким образом, мы можем „перенести“ операцию ω на классы эквивалентности, положив

$$[a_1]_\rho \ldots [a_n]_\rho \omega = [a_1 \ldots a_n \omega].$$

Аналогично для любого n-арного ($n \geqslant 1$) отношения, по определению, полагаем (для любого $n \geqslant 1$, любого $\pi \in \Pi^{(n)}$ и любых $a_1, \ldots, a_n$)

$$([a_1]_\rho, \ldots, [a_n]_\rho) \in \pi \Leftrightarrow (a_1, \ldots, a_n) \in \pi,$$

поскольку, согласно определению конгруэнтности, для любых $x_1 \in [a_1]_\rho$, ..., $x_n \in [a_n]_\rho$ условие $(x_1, \ldots, x_n) \in \pi$ зависит лишь от классов эквивалентности элементов x_i, $i = \overline{1, n}$.

Заметим, что в этом случае мы использовали одинаковые символы (ω и π) для соответствующих операций и отношения на разных множествах, опираясь на соглашение о сигнатурах однотипных алгебраических систем.

Итак, операции и отношения исходной сигнатуры можно перенести на множество классов эквивалентности по конгруэнции ρ согласно приведенным выше формулам. Получаемая при этом алгебраическая система (*однотипная* с исходной) имеет в качестве носителя *фактор-множество* A/ρ. Ее называют ***фактор-системой*** алгебраической системы A ***по конгруэнции*** ρ и обозначают $\mathcal{A}/\rho$. В том случае, когда исходная алгебраическая система $\mathcal{A}$ является алгеброй (моделью), ее фактор-систему $\mathcal{A}/\rho$ называют ***фактор-алгеброй*** (***фактор-моделью*** соответственно).

Пример 4.5. Вернемся к примеру 4.4.а. Конгруэнция, определенная в этом примере, — не что иное, как отношение $\sim_{\mathbb{Z}}$ (напомним, что для действительных чисел x, y $x \sim_{\mathbb{Z}} y \Leftrightarrow$ $\Leftrightarrow x - y \in \mathbb{Z}$, см. **2.6**). Следовательно, фактор-алгебра $(\mathbb{R}, +, 0)$ аддитивной группы целых чисел по конгрузнции равенства по модулю 1 — это *фактор-группа* $\mathbb{R}/\mathbb{Z}$ аддитивной *группы* действительных чисел *по нормальному делителю* $\mathbb{Z}$, т.е. по подгруппе целых чисел. #

Пример 4.5 есть проявление общей связи между понятиями конгруэнции и нормального делителя группы. Рассмотрим этот вопрос подробно.

Пусть $\mathcal{G} = (G, \cdot, \mathbf{1})$ — группа, а $\mathcal{H} = (H, \cdot, \mathbf{1})$ — ее подгруппа, являющаяся нормальным делителем. Отношение $\sim_H$, определенное на носителе G исходной группы так, что

$$a \sim_H b \Leftrightarrow aH = bH,$$

есть, согласно теореме 2.11, эквивалентность. Докажем, что $\sim_H$ является конгруэнцией. Для этого достаточно доказать, что для любых a, b, c, $d \in G$ из $a \sim_H c$ и $b \sim_H d$ следует $ab \sim_H cd$.

Имеем $a \sim_H c$, и это означает, что $aH = cH$. Точно так же $bH = dH$ в силу $b \sim_H d$. Так как $\mathcal{H}$ — нормальный делитель, то $abH = aHbH$. Далее, $aHbH = cHdH$, и, снова используя свойство нормального делителя, получаем $cHdH = cdH$, откуда $abH = cdH$.

Фактор-алгебра группы $\mathcal{G}$ по конгруэнции $\sim_H$ совпадает (подчеркнем — не просто изоморфна, а именно совпадает) с *фактор-группой* группы $\mathcal{G}$ по нормальному делителю H (см. **2.8**).

Ниже (см. **4.4**) показано, что и, наоборот, фактор-алгебра любой группы по произвольной конгруэнции есть ее фактор-группа по некоторому нормальному делителю. Мы продолжим обсуждение идеи фактор-системы и поймем ее глубже, когда свяжем понятие фактор-системы с общим понятием гомоморфизма, знакомого нам пока только по частным случаям гомоморфизмов групп и колец.

4.4. Гомоморфизмы

Пусть $\mathcal{A} = (A, \Omega, \Pi)$ и $\mathcal{B} = (B, \Omega, \Pi)$ — две *однотипные алгебраические системы.*

Отображение h: $A \to B$ называют ***гомоморфизмом*** алгебраической системы $\mathcal{A}$ в алгебраическую систему $\mathcal{B}$, если выполняются следующие условия:

1) для любой *n-арной операции* ω ($n \geqslant 0$) и любых элементов $a_1, \ldots, a_n \in A$

$$h(a_1 \ldots a_n \omega) = h(a_1) \ldots h(a_n)\omega;$$

2) для любого *n-арного отношения* π ($n \geqslant 1$) и любых элементов $a_1, \ldots, a_n \in A$ из того, что $(a_1, \ldots, a_n) \in \pi$, следует, что $(h(a_1), \ldots, h(a_n)) \in \pi$.

Мы будем использовать обозначение $h\colon \mathcal{A} \to \mathcal{B}$ для отображения h, являющегося гомоморфизмом алгебраической системы $\mathcal{A} = (A, \Omega, \Pi)$ в алгебраическую систему $\mathcal{B} = (B, \Omega, \Pi)$.

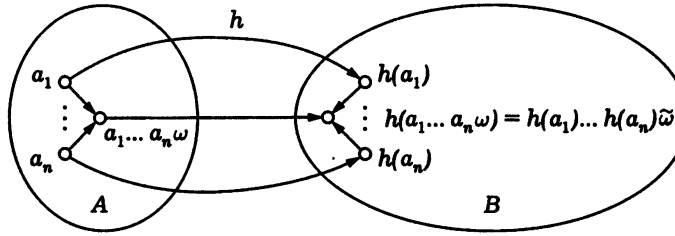


Рис. 4.2

В определении гомоморфизма первое условие, которое можно рассматривать как условие „сохранения операций", означает следующее. Если отображение h — гомоморфизм, то, вычисляя *образ результата применения* любой *операции* $\omega \in \Omega$ к любому *кортежу аргументов* из *носителя* алгебраической системы $\mathcal{A}$, т.е. образ произвольного элемента $a_1 \ldots a_n\omega$, мы можем сначала определить образ каждого из аргументов и уже к ним, т.е. к элементам $h(a_1)$, ..., $h(a_n)$, на носителе алгебраической системы $\mathcal{B}$ применить рассматриваемую операцию (точнее, операцию второй алгебраической системы, которая соответствует операции ω; напомним, что соответствующие друг другу операции и отношения однотипных алгебраических систем мы договорились обозначать одинаково). Эта ситуация проиллюстрирована на рис. 4.2.

Второе условие в определении гомоморфизма выражает „сохранение отношений": если элементы $a_1, \ldots, a_n$ первой алгебраической системы *связаны отношением* ρ, т.е. $(a_1, \ldots, a_n) \in \pi$, то их образы $h(a_1)$, ..., $h(a_n)$ при гомоморфизме h связаны „тем же"* отношением во второй алгебраической системе, т.е. $(h(a_1), \ldots, h(a_n)) \in \pi$.

*Точнее, так же обозначенным. Заключая слова „тем же" в кавычки, мы еще раз подчеркиваем условность одинакового обозначения операций и отношений однотипных алгебраических систем.

Заметим сразу, что из того, что $(h(a_1), \ldots, h(a_n)) \in \pi$, не следует, вообще говоря, что $(a_1, \ldots, a_n) \in \pi$. Если же это имеет место, т.е. для всякого n-арного отношения π $(n \geqslant 1)$ и любых $a_1, \ldots, a_n \in A$ $(a_1, \ldots, a_n) \in \pi$ тогда и только тогда, когда $(h(a_1), \ldots, h(a_n)) \in \pi$, то такой гомоморфизм называют ***строгим гомоморфизмом***.

Если строгий гомоморфизм алгебраической системы $\mathcal{A}$ в алгебраическую систему $\mathcal{B}$ является биекцией $A \to B$, то его называют ***изоморфизмом***. Из определения изоморфизма следует, что для изоморфизма h: $\mathcal{A} \to \mathcal{B}$ *обратное отображение* h^{-1}: $\mathcal{B} \to \mathcal{A}$ также является изоморфизмом. В этом случае алгебраические системы $\mathcal{A}$ и $\mathcal{B}$ называют ***изоморфными*** и пишут $\mathcal{A} \cong \mathcal{B}$. Если алгебраические системы интересуют нас лишь со стороны свойств их операций и отношений, то изоморфные алгебраические системы в этом смысле не различимы, и тогда говорят о совпадении алгебраических систем с точностью до изоморфизма.

Если гомоморфизм является инъекцией, то его называют ***мономорфизмом*** или ***вложением***. В том случае, когда существует вложение алгебраической системы $\mathcal{A}$ в алгебраическую систему $\mathcal{B}$, которое является также строгим гомоморфизмом, то говорят, что первая алгебраическая система изоморфно вкладывается во вторую.

Если гомоморфизм h: $\mathcal{A} \to \mathcal{B}$ является сюръекцией, то его называют ***эпиморфизмом*** $\mathcal{A}$ на $\mathcal{B}$. При эпиморфизме носитель алгебраической системы $\mathcal{B}$ совпадает с образом носителя алгебраической системы $\mathcal{A}$ при отображении h, т.е. $B = h(A)$. В этом случае говорят также, что алгебраическая система $\mathcal{B}$ является ***гомоморфным образом системы*** $\mathcal{A}$ при гомоморфизме h, и записывают это как $\mathcal{B} = h(\mathcal{A})$.

Гомоморфизм алгебраической системы $\mathcal{A}$ в себя называют ***эндоморфизмом*** алгебраической системы $\mathcal{A}$. Эндоморфизм, являющийся изморфизмом, называют ***автоморфизмом***.

Легко доказать следующее утверждение.

Теорема 4.2. Если $h\colon \mathcal{A} \to \mathcal{B}$ и $g\colon \mathcal{B} \to \mathcal{D}$ — гомоморфизмы, то композиция $h \circ g\colon \mathcal{A} \to \mathcal{D}$ — тоже гомоморфизм. #

Ранее (см. **2** и **3**) мы достаточно подробно обсудили понятие гомоморфизма для *алгебр — групп* и *колец*. Заметим, что для алгебр любой гомоморфизм является строгим, так как сигнатура алгебры включает только операции и условие 2 в определении гомоморфизма для нее не рассматривается. Приведем некоторые дополнительные примеры гомоморфизмов.

Пример 4.6. а. Рассмотрим *левый $\mathcal{R}$-модуль*

$$\mathcal{M} = (M,\ +,\ \{\omega_\alpha\colon \alpha \in R\},\ \mathbf{0})$$

как алгебраическую систему, сигнатура которой помимо групповой операции сложения и нуля группы $\mathbf{0}$ содержит и унарные операции умножения элементов группы на элементы кольца $\mathcal{R}$ (мощность множества этих операций равна мощности $|R|$ носителя R кольца $\mathcal{R}$). Гомоморфизм $h\colon \mathcal{M}_1 \to \mathcal{M}_2$ этих $\mathcal{R}$-модулей есть такое отображение, что для всех $x,\ y \in M$ и всех $\alpha \in R$

$$h(x+y) = h(x) + h(y),$$
$$h(\omega_\alpha(x)) = \omega_\alpha(h(x)).$$

Используя вместо выражения $\omega_\alpha(x)$ более привычную запись $\alpha \circ x$, приведенные выше условия представим в виде

$$h(x+y) = h(x) + h(y),$$
$$h(\alpha \circ x) = \alpha \circ h(x).$$

В случае *линейного пространства над полем* отображение, удовлетворяющее этим условиям, есть не что иное, как *линейный оператор*. Таким образом, линейные операторы суть гомоморфизмы линейных пространств.

б. Для линейного пространства отображение, сопоставляющее каждому вектору $\boldsymbol{x}$ вектор $\boldsymbol{x} + \boldsymbol{a}$ для фиксированного ненулевого вектора $\boldsymbol{a}$, не является линейным оператором и, следовательно, не является гомоморфизмом заданного линейного

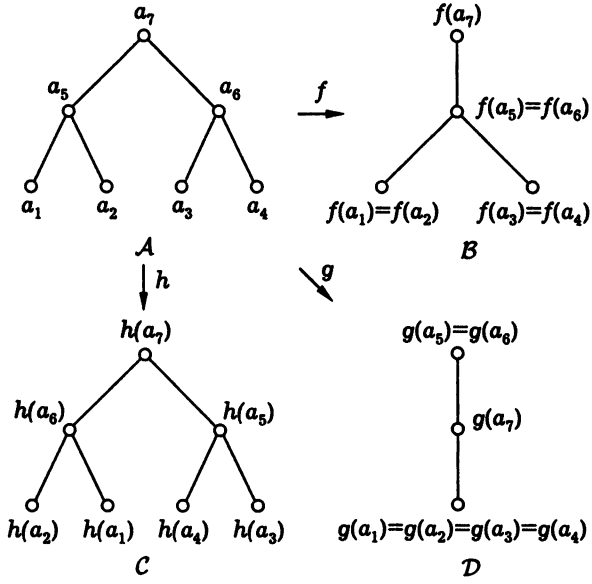


Рис. 4.3

пространства в себя. Действительно, для любого гомоморфизма модулей (и линейных пространств, в частности) образ нулевого вектора есть нулевой вектор. Здесь же $\mathbf{0} \mapsto a \neq \mathbf{0}$.

в. Пусть $\mathcal{A} = (A, \rho)$ и $\mathcal{B} = (B, \rho)$ — *модели* с одним бинарным отношением. Отображение h: $A \to B$ будет гомоморфизмом первой модели во вторую, согласно с общим определением, тогда и только тогда, когда для любых x, $y \in A$ из $x \rho y$ следует $h(x) \rho h(y)$. В частности, если ρ — *отношение порядка*, то получаем $x \leqslant y \Rightarrow h(x) \leqslant h(y)$.

Таким образом, гомоморфизмы *упорядоченных множеств* — это *монотонные отображения*. На рис. 4.3 в виде диаграмм Хассе изображены четыре упорядоченных множества. Множество B является гомоморфным образом множества A, но не является его строгим гомоморфным образом; множество C есть строгий гомоморфный образ множества A; наконец, множество D не является гомоморфным образом множества A. #

Установим теперь связь между понятием гомоморфизма и понятием *фактор-системы*. Доказываемые ниже результаты конкретизируют для алгебраических систем связь между понятием отображения и понятием отношения эквивалентности (см. **1.7**).

Теорема 4.3. Для любой конгруэнции ρ на алгебраической системе $\mathcal{A} = (A, \Omega, \Pi)$ *каноническая сюръекция* h_ρ множества A является строгим эпиморфизмом алгебраической системы $\mathcal{A}$ на ее фактор-систему $\mathcal{A}/\rho$.

◀ Для канонической сюръекции задается h_ρ. Имеем $h_\rho(x) = [x]_\rho$ (для произвольного $x \in A$). В силу определения конгруэнции для произвольных n-арной операции ω и n-арного отношения π и любых $a_1, \ldots, a_n \in A$, согласно определению операций и отношений на фактор-множестве A/ρ (см. **4.3**), имеем

$$h_\rho(a_1 \ldots a_n\omega) = [a_1 \ldots a_n\omega]_\rho = [a_1]_\rho \ldots [a_n]_\rho\omega = h_\rho(a_1) \ldots h_\rho(a_n)\omega,$$

$$(a_1, \ldots, a_n) \in \pi \Leftrightarrow ([a_1]_\rho, \ldots, [a_n]_\rho) \in \pi,$$

откуда и следует, что h_ρ — строгий гомоморфизм. ▶

Ввиду теоремы 4.3 каноническую сюръекцию h_ρ (для конгруэнции ρ) можно назвать ***каноническим гомоморфизмом*** алгебраической системы $\mathcal{A}$.

Справедлива теорема, обратная теореме 4.3.

Теорема 4.4. Для любого строгого гомоморфизма $h\colon \mathcal{A} \to \mathcal{B}$ отношение ρ_h на носителе A алгебраической системы $\mathcal{A}$, определенное так, что $x\rho_h y \Leftrightarrow h(x) = h(y)$ для любых $x, y \in A$, является конгруэнцией, причем имеет место изоморфизм $h(\mathcal{A}) \cong \mathcal{A}/\rho_h$.

◀ Пусть $h\colon \mathcal{A} \to \mathcal{B}$ — гомоморфизм, тогда ρ_h — эквивалентность (см. **1.7**).

Введем отображение $i\colon A/\rho_h \to h(A)$, полагая $i([a]_{\rho_h}) = h(a)$. Это действительно отображение, так как эквивалентные элементы множества A имеют один и тот же образ. Поскольку

неэквивалентные элементы имеют разные образы, то i — инъекция. Очевидно, что i является сюръекцией. Следовательно, i — биекция.

Далее, если $a_1 \rho_h b_1, \ldots, a_n \rho_h b_n$, то

$$h(a_1 \ldots a_n \omega) = h(a_1) \ldots h(a_n)\omega = h(b_1) \ldots h(b_n)\omega = h(b_1 \ldots b_n \omega),$$

т.е. $a_1 \ldots a_n \omega \, \rho_h \, b_1 \ldots b_n \omega$ для любой n-арной операции ω.

Точно так же

$$(a_1, \ldots, a_n) \in \pi \Rightarrow (h(a_1), \ldots, h(a_n)) \in \pi \Rightarrow \\ \Rightarrow (h(b_1), \ldots, h(b_n)) \in \pi \Rightarrow (b_1, \ldots, b_n) \in \pi,$$

причем последняя импликация справедлива в силу того, что h — строгий гомоморфизм.

Итак, ρ_h — конгруэнция на A. Остается доказать, что имеет место изоморфизм $h(\mathcal{A}) \simeq \mathcal{A}/\rho_h$.

Далее, если $\omega \in \Omega^{(n)}$, то

$$i([a_1]_{\rho_h}, \ldots, [a_n]_{\rho_h}\omega) = i([a_1 \ldots a_n \omega]_{\rho_h}) = \\ = h(a_1 \ldots a_n \omega) = h(a_1) \ldots h(a_n)\omega = \\ = i([a_1]_{\rho_h}) \ldots i([a_n]_{\rho_h})\omega,$$

т.е. i „сохраняет“ операции.

Рассуждая аналогично, можно доказать, что i „сохраняет“ и отношения как любой гомоморфизм, и, более того, поскольку h — строгий гомоморфизм, то для любого отношения π

$$(i([a_1]_{\rho_h}), \ldots, i([a_n]_{\rho_h})) \in \pi \Rightarrow (h(a_1), \ldots, h(a_n)) \in \pi \Rightarrow \\ \Rightarrow (a_1, \ldots, a_n) \in \pi \Rightarrow ([a_1]_{\rho_h}, \ldots, [a_n]_{\rho_h}) \in \pi,$$

и, следовательно, i — изоморфизм $\mathcal{A}/\rho_h$ на $h(\mathcal{A})$ (и вложение $\mathcal{A}/\rho_h$ в $\mathcal{B}$). ▶

Таким образом, любой гомоморфный образ алгебраической системы совпадает с точностью до изоморфизма с некоторой ее фактор-системой.

Пример 4.7. Требование строгости гомоморфизма в теореме 4.4 является существенным. В примере 4.4.д монотонное отображение f не определяет конгруэнции на упорядоченном множестве $\mathcal{A}$, так как не является строгим гомоморфизмом (см. пример 4.6.в). #

Из теоремы 4.4 вытекает, что любой строгий гомоморфизм $h\colon \mathcal{A} \to \mathcal{B}$ можно разложить в композицию канонического гомоморфизма $\mathcal{A}$ на $\mathcal{A}/\rho_h$ и мономорфизма $\mathcal{A}/\rho_h$ в $\mathcal{B}$ (который будет изоморфизмом $\mathcal{A}/\rho_h \cong h(\mathcal{A})$).

Применяя доказанные теоремы 4.3 и 4.4 к частному случаю алгебраических систем — алгебрам, получаем следствие.

Следствие 4.1. 1. Любой гомоморфизм h алгебры $\mathcal{A} = (A, \Omega)$ однозначно определяет конгруэнцию ρ_h на A, такую, что $h(\mathcal{A}) \cong \mathcal{A}/\rho_h$.

2. Любая конгруэнция ρ на алгебре $\mathcal{A} = (A, \Omega)$ однозначно определяет некоторый гомоморфизм h_ρ данной алгебры на фактор-алгебру $\mathcal{A}/\rho$.

Применим полученные результаты к теории групп. Докажем, что *фактор-группа* заданной группы *по нормальному делителю* (см. **2**) совпадает с фактор-алгеброй указанной группы по некоторой конгруэнции.

Прежде всего заметим, что поскольку фактор-алгебра любой группы (по любой конгруэнции) изоморфна некоторому *гомоморфному образу* этой *группы*, а гомоморфный образ всякой группы является группой (см. **2.8**), то фактор-алгебра группы по любой конгруэнции есть группа.

Докажем теперь следующую теорему.

Теорема 4.5. Пусть $\mathcal{G} = (G, \cdot, \mathbf{1})$ — произвольная группа и ρ — конгруэнция на ней. Тогда фактор-группа $\mathcal{G}/\rho$ совпадает с фактор-группой $\mathcal{G}/\mathcal{H}$ по некоторому нормальному делителю $\mathcal{H}$ группы $\mathcal{G}$.

◀ Рассмотрим канонический гомоморфизм h_ρ группы $\mathcal{G}$. Его *ядром* является множество всех элементов, эквивалентных (по

конгруэнции ρ) *единице группы*. Но поскольку, согласно теореме 2.19, ядро каждого *гомоморфизма групп* есть *нормальный делитель*, то множество $[\mathbf{1}]_\rho$ (класс эквивалентности единицы группы) является нормальным делителем*. Этот нормальный делитель, обозначаемый далее $\mathcal{H}$, будучи классом эквивалентности единицы группы $\mathcal{G}$, является единицей фактор-группы $\mathcal{G}/\rho$ и фактор-группы $\mathcal{G}/\mathcal{H}$. Осталось показать, что произвольный левый смежный класс $a[\mathbf{1}]_\rho$ совпадает с классом эквивалентности элемента $a \in G$. Для всякого элемента y этого класса имеем $y = ax$ для некоторого $x\rho\mathbf{1}$. Тогда, так как ρ – конгруэнция, получим

$$[y]_\rho = [ax]_\rho = [a]_\rho[x]_\rho = [a]_\rho[\mathbf{1}]_\rho = [a]_\rho,$$

откуда $a[\mathbf{1}]_\rho = [a]_\rho$. Итак, каждый левый смежный класс по нормальному делителю $\mathcal{H}$ является одновременно классом эквивалентности по исходной конгруэнции ρ, а группы $\mathcal{G}/\rho$ и $\mathcal{G}/\mathcal{H}$ совпадают. ▶

Таким образом, имеет место взаимно однозначное соответствие между конгруэнциями на группе и нормальными делителями этой группы, и каждая фактор-группа по конгруэнции является в то же время и фактор-группой по нормальному делителю, и наоборот.

Важным результатом для групп является и следующая теорема.

Теорема 4.6. Ядрами гомоморфизмов групп служат нормальные делители, и только они.

◀ То, что ядро гомоморфизма групп есть нормальный делитель, доказано выше (см. **2.8**). Наоборот, если $\mathcal{H}$ — нормальный делитель $\mathcal{G}$, то отношение $\sim_H$ — конгруэнция (см. **4.3**), а множество H — ядро соответствующего канонического гомоморфизма. ▶

*Точнее, нормальным делителем будет подгруппа, носителем которой является множество $[\mathbf{1}]_\rho$.

В силу установленной связи между фактор-системами и гомоморфизмами можно утверждать, что фактор-алгебра любого кольца по любой конгруэнции на этом кольце является кольцом. Естественно назвать его ***фактор-кольцом*** (по заданной конгруэнции).

Пример 4.8. Как уже было показано в примере 4.4.б, отношение $=_k$ есть конгруэнция на кольце целых чисел. Можно доказать, что фактор-кольцо кольца целых чисел по этой конгруэнции изоморфно кольцу $\mathbb{Z}_k$ вычетов по модулю k, поскольку соответствующий канонический гомоморфизм сопоставляет каждому целому числу $m \in \mathbb{Z}$ его класс эквивалентности $[m]_{=_k}$ и существует естественное, сохраняющее операции взаимно однозначное соответствие между множеством этих классов (т.е. фактор-множеством $\mathbb{Z}/=_{(\mathrm{mod}\, k)}$) и множеством $\{0, 1, \ldots, k-1\}$ остатков от деления на k (см. также пример 2.25). #

В заключение докажем три теоремы, которые описывают с точностью до изоморфизма все полугруппы, группы и кольца.

Будем говорить, что полугруппа $\mathcal{S} = (S, \cdot)$ изоморфно вкладывается в моноид $\mathcal{M} = (M, \cdot, 1)$, если существует мономорфизм $\mathcal{S}$ в $\mathcal{M}$, т.е. если образ полугруппы $\mathcal{S}$ при этом мономорфизме является некоторой подполугруппой (но, вообще говоря, не подмоноидом) моноида $\mathcal{M}$.

Например, полугруппа $((0, 1), \cdot)$ изоморфно вкладывается в моноид $((0, 1], \cdot, 1)$, где операция $\cdot$ — обычное умножение чисел.

Теорема 4.7. Любая полугруппа изоморфно вкладывается в некоторый моноид.

◀ Пусть $\mathcal{S} = (S, \cdot)$ — полугруппа, не являющаяся моноидом (так как иначе она тривиально вкладывается сама в себя). Пусть $\{\varepsilon\}$ — произвольное одноэлементное множество, не пересекающееся с S. Определим на множестве $S \cup \{\varepsilon\}$ операцию $\cdot$ следующим образом: на S операция $\cdot$ совпадает с операцией полугруппы $\mathcal{S}$, а для каждого $a \in S \cup \{\varepsilon\}$ она удовлетворяет со-

отношениям $a \cdot \varepsilon = \varepsilon \cdot a = a$. Очевидно, что $\mathcal{M} = (S \cup \varepsilon, \cdot, \varepsilon)$ — требуемый моноид. ▶

Теорема 4.8. Любой моноид изоморфно вкладывается в *симметрический моноид* некоторого множества A.

◀ Пусть $\mathcal{S} = (S, \cdot, 1)$ — моноид. Сопоставим каждому $a \in S$ преобразование f_a: $x \mapsto x \cdot a$ (правый сдвиг на a) множества S. Отображение $a \mapsto f_a$ множества S в множество всех преобразований S инъективно, поскольку если $a \neq b$, то $f_a(1) \neq f_b(1)$ и $f_a \neq f_b$. Далее, если $c = a \cdot b$, то

$$f_{ab}(x) = x \cdot c = x(ab) = (xa)b = f_b(f_a(x)) = (f_a \circ f_b)(x).$$

Итак, $a \mapsto f_a$ есть мономорфизм $\mathcal{S}$ в симметрический моноид множества A. ▶

Теорема 4.9 (*теорема Кэли*). Любая группа изоморфно вкладывается в *симметрическую группу* некоторого множества A.

◀ Если $\mathcal{S} = (S, \cdot, 1)$ — группа, то введенное в доказательстве теоремы 4.8 преобразование f_a: $x \mapsto x \cdot a$ множества S будет уже биекцией. Чтобы это доказать, достаточно построить *отображение*, *обратное* f_a. Действительно, сдвиг $f_{a^{-1}}$ на a^{-1} будет отображением, обратным сдвигу на a:

$$f_{a^{-1}}(f_a(x)) = (x \cdot a) \cdot a^{-1} = x \cdot (a \cdot a^{-1}) = x \cdot 1 = x.$$

Точно так же и $f_a(f_{a^{-1}}(x) = x$. Из доказанного следует, что множество всех правых сдвигов множества S образует по операции композиции группу, являющуюся подгруппой симметрической группы множества S.

Из доказательства теоремы 4.8 заключаем, что отображение $a \mapsto f_a$, сопоставляющее каждому элементу a множества S (носителя моноида $\mathcal{S}$) сдвиг на a, инъективно и является, более того, мономорфизмом моноида $\mathcal{S}$ в симметрический моноид множества S. Но поскольку, как мы только что показали, в том

случае, когда моноид S является группой, для любого $a \in S$ выполняется равенство $f_{a^{-1}}(x) = f_a^{-1}(x)$, то данный мономорфизм отображает элемент, обратный к a, в сдвиг, обратный сдвигу f_a. Тем самым он оказывается уже мономорфизмом группы S в группу всех подстановок множества S (и изоморфизмом на подгруппу всех правых сдвигов множества S), т.е. изоморфным вложением первой группы во вторую. ▶

Пусть $\mathcal{K}=(K,+,\mathbf{0})$ — абелева группа. На множестве $\mathrm{End}(\mathcal{K})$ всех эндоморфизмов группы $\mathcal{K}$ можно определить структуру кольца следующим образом. Умножение эндоморфизмов определим как их композицию, а сложение — так, что для любого $x \in K$ выполнено равенство $(f+g)(x) = f(x)+g(x)$.

Введем также нулевой эндоморфизм O, для всех x положив $O(x) = \mathbf{0}$, и каждому эндоморфизму f сопоставим противоположный эндоморфизм $-f$, для каждого x положив $(-f)(x) = -f(x)$.

Можно доказать, что тем самым действительно определено кольцо (проверив все *аксиомы кольца*, см. **2.3**). Его называют ***кольцом эндоморфизмов абелевой группы*** $\mathcal{K}$.

Теорема 4.10. Любое кольцо изоморфно вкладывается в кольцо эндоморфизмов некоторой абелевой группы. #

Доказательство этой теоремы проводится по аналогии с доказательствами теорем 4.8 и 4.9. Искомое вложение определяется следующим образом: пусть $\mathcal{R} = (R, +, \cdot, \mathbf{0}, 1)$ — кольцо. Для любого $r \in R$ положим $f_r(x) = x \cdot r$. Тогда отображение $r \mapsto f_r$ и есть требуемое вложение, причем в качестве абелевой группы выступает аддитивная группа кольца $\mathcal{R}$.

4.5. Прямые произведения алгебраических систем

Часто возникает необходимость, имея некоторые исходные *однотипные алгебраические системы*, определенным образом „распространить“ их *операции* и *отношения* на *декартово*

произведение их *носителей*: например, определить структуру *группы* (*кольца*, *поля*) на декартовом произведении носителей некоторых групп (колец, полей) или перенести структуру *индуктивного упорядоченного множества* на декартово произведение носителей заданных индуктивных упорядоченных множеств и т.п. Выясним, как осуществляется такой перенос операций и отношений, а также сформулируем некоторые условия, при которых все свойства исходных алгебраических систем сохраняются в их декартовом произведении.

Пусть $\mathcal{A}_1 = (A_1, \Omega, \Pi), \ldots, \mathcal{A}_n = (A_n, \Omega, \Pi)$ — однотипные алгебраические системы (их *сигнатуры*, как и элементы этих сигнатур, обозначаются одинаково). Рассмотрим декартово произведение $B = A_1 \times \ldots \times A_n$ их носителей и перенесем на B операции и отношения исходной сигнатуры следующим образом.

1. Для любой m-*арной* $(m \geqslant 1)$ *операции* $\omega \in \Omega$ и произвольных *кортежей* $\boldsymbol{x}_i = (x_{i1}, \ldots, x_{in}) \in B$, $i = \overline{1, m}$, положим

$$\boldsymbol{x}_1 \ldots \boldsymbol{x}_m \omega = (x_{11} \ldots x_{m1}\omega, \ldots, x_{1n} \ldots x_{mn}\omega).$$

Для любых *нульарных операций* $a_1, \ldots, a_n$ алгебраических систем $\mathcal{A}_1, \ldots, \mathcal{A}_n$ определим кортеж $\boldsymbol{a} = (a_1, \ldots, a_n)$ как нульарную операцию на множестве B.

2. Для любого s-*арного отношения* $\pi \in \Pi$ $(s \geqslant 1)$ и произвольных кортежей $\boldsymbol{x}_i = (x_{i1}, \ldots, x_{in}) \in B$, $i = \overline{1, s}$, положим $(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_s) \in \pi$ тогда и только тогда, когда $(x_{1j}, \ldots, x_{sj}) \in \pi$ для каждого $j = \overline{1, n}$.

Полученную таким образом алгебраическую систему $\mathcal{B}$ на множестве $B = A_1 \times \ldots \times A_n$ называют ***прямым*** (декартовым) ***произведением алгебраических систем*** $\mathcal{A}_1, \ldots, \mathcal{A}_n$ и обозначают $\mathcal{A}_1 \times \ldots \times \mathcal{A}_n$.

В том случае, когда $\mathcal{A}_1 = \ldots = \mathcal{A}_n = \mathcal{A}$, получаем ***n-ю декартову степень*** алгебраической системы $\mathcal{A}$, обозначаемую $\mathcal{A}^n$.

Замечание 4.4. Несколько особняком стоит случай $n = 0$. Как известно (см. **2.2**), нулевая декартова степень произвольно-

го множества A есть, по определению, одноэлементное множество $\{\lambda\}$ и его элемент λ называется пустым кортежем. Тогда нулевая декартова степень $\mathcal{A}^0$ алгебраической системы $\mathcal{A}$ есть алгебраическая система $(\{\lambda\}, \Omega, \Pi)$, где $\lambda \ldots \lambda\omega = \lambda$ для всех $\omega \in \Omega$ и $(\lambda, \ldots, \lambda) \in \pi$ для всех $\pi \in \Pi$. #

Итак, операции и отношения исходных однотипных алгебраических систем переносятся на декартово произведение их носителей покомпонентно.

Пример 4.9. Рассмотрим алгебраическую систему $\mathcal{R} = (\mathbb{R}, +, \cdot, 0, 1, \leqslant)$, сигнатура которой состоит из обычных операций сложения, умножения (бинарные операции), 0 и 1 (нульарные операции) и *естественного числового порядка* (*бинарное отношение*). Распространим эти операции и отношения на декартов квадрат $\mathbb{R} \times \mathbb{R} = \mathbb{R}^2$ множества действительных чисел согласно определению, данному выше.

Сложение *упорядоченных пар* действительных чисел определяется тогда равенством

$$(a_1, b_1) + (a_2, b_2) = (a_1 + a_2, b_1 + b_2),$$

а умножение — равенством

$$(a_1, b_1) \cdot (a_2, b_2) = (a_1 \cdot a_2, b_1 \cdot b_2).$$

При этом легко понять, что упорядоченная пара $(0, 0)$ будет *нейтральным элементом* по сложению в $\mathbb{R}^2$, а упорядоченная пара $(1, 1)$ — нейтральным элементом по умножению. Кроме того, для любых действительных чисел a, b будем иметь $(a, b) \cdot (0, 0) = (0, 0)$, т.е. пара $(0, 0)$ играет роль *нуля* относительно умножения в $\mathbb{R}^2$.

Отношение порядка на множестве упорядоченных пар вводится по правилу*

$$(a_1, b_1) \leqslant (a_2, b_2) \Leftrightarrow (a_1 \leqslant a_2 \text{ и } b_1 \leqslant b_2).$$

*Нужно, разумеется, проверить, что построено действительно отношение порядка, но эта проверка легко выполняется.

Заметим, что мы уже ранее многократно пользовались подобным определением отношения порядка на множестве упорядоченных пар (и, в более общем случае, кортежей). #

Для каждого $i = \overline{1,n}$ определим *проекцию* $\mathrm{pr}_i\colon B \to A_i$, полагая $\mathrm{pr}_i(\boldsymbol{x}) = x_i$. Можно показать, что pr_i есть *строгий гомоморфизм*, называемый ***проектирующим гомоморфизмом***.

Из определения декартова произведения алгебраических систем априори не следует, что в нем сохраняются все свойства операций и отношений перемножаемых алгебраических систем. Разберем в этой связи такой пример.

Пример 4.10. Пусть $\mathcal{K}_1$ и $\mathcal{K}_2$ — поля. Их произведение $\mathcal{K}_1 \times \mathcal{K}_2$ не будет полем, так как в этом произведении возникают делители нуля. В самом деле, если $a \in K_1 \setminus \{\mathbf{0}\}$, $b \in K_2 \setminus \{\mathbf{0}\}$, то $(a,\mathbf{0}) \cdot (\mathbf{0},b) = (\mathbf{0},\mathbf{0})$ — элемент, являющийся нулем произведения $\mathcal{K}_1 \times \mathcal{K}_2$. Таким образом, алгебра $\mathcal{K}_1 \times \mathcal{K}_2$ будет, как нетрудно убедиться, только кольцом. #

Этот пример показывает, что в декартовом произведении могут теряться некоторые свойства исходных алгебраических систем. В частности, декартово произведение полей не будет полем. Здесь уместно вспомнить о *поле комплексных чисел*, носителем которого является декартов квадрат $\mathbb{R}^2$. Но если сложение в этом поле определяется покомпонентно, т.е. по правилам декартова произведения алгебр, то умножение введено по более сложному правилу, позволяющему сохранить *аксиомы поля*.

Теорема 4.11. 1. Прямое произведение полугрупп (моноидов, групп) есть полугруппа (моноид, группа).

2. Прямое произведение полуколец (идемпотентных полуколец, колец) есть полукольцо (идемпотентное полукольцо, кольцо).

3. Прямое произведение полурешеток, решеток, симметричных полуколец, булевых алгебр есть соответственно полурешетка, решетка, симметричное полукольцо, булева алгебра.

4. Прямое произведение (индуктивных) упорядоченных множеств есть (индуктивное) упорядоченное множество.

◀ Для простоты будем рассматривать доказательство для произведения двух алгебраических систем.

1. Если $\mathcal{S}_1 = (S_1, \cdot)$ и $\mathcal{S}_2 = (S_2, \cdot)$ — две полугруппы, то, вводя на множестве $S_1 \times S_2$ операцию $\cdot$ так, что для любых a_1, $b_1 \in S_1$, a_2, $b_2 \in S_2$ справедливо

$$(a_1, a_2) \cdot (b_1, b_2) = (a_1 \cdot b_1, a_2 \cdot b_2),$$

получим полугруппу, поскольку в силу ассоциативности операции $\cdot$ на множествах S_1 и S_2 будем иметь

$$\begin{aligned}(a_1, a_2) \cdot ((b_1, b_2) \cdot (c_1, c_2)) &= (a_1, a_2) \cdot (b_1 \cdot c_1, b_2 \cdot c_2) = \\ &= (a_1 \cdot (b_1 \cdot c_1), a_2 \cdot (b_2 \cdot c_2)) = ((a_1 \cdot b_1) \cdot c_1, (a_2 \cdot b_2) \cdot c_2) = \\ &= ((a_1, a_2) \cdot (b_1, b_2)) \cdot (c_1, c_2).\end{aligned}$$

Если в каждой из указанных выше полугрупп существует нейтральный элемент e_1 и e_2 соответственно, то легко проверить, что упорядоченная пара (e_1, e_2) является нейтральным элементом по операции $\cdot$ на декартовом произведении $S_1 \times S_2$.

Если же моноиды $\mathcal{S}_1$ и $\mathcal{S}_2$ суть группы, то элемент из $S_1 \times S_2$, обратный к (a_1, a_2), равен $\left(a_1^{-1}, a_2^{-1}\right)$.

2. Доказательство для полукольца и кольца проводится аналогично предыдущему.

3. Точно так же, совершенно аналогично доказательству для групп, проводится доказательство для полурешеток, решеток, симметричных полуколец и булевых алгебр.

4. Пусть теперь $\mathcal{M}_1 = (M_1, \leqslant)$ и $\mathcal{M}_2 = (M_2, \leqslant)$ — индуктивные упорядоченные множества, *наименьшие элементы* которых суть $\mathbf{0}_1$ и $\mathbf{0}_2$.

Определяя отношение порядка $\leqslant$ на декартовом произведении $M_1 \times M_2$ покомпонентно, так же как это сделано в примере 4.9, получаем, что упорядоченная пара $(\mathbf{0}_1, \mathbf{0}_2)$ является на указанном произведении наименьшим элементом.

Далее, произвольной неубывающей последовательности упорядоченных пар

$$(a_1, b_1) \leqslant (a_2, b_2) \leqslant \ldots \leqslant (a_n, b_n) \leqslant \ldots$$

соответствуют две неубывающие последовательности

$$a_1 \leqslant a_2 \leqslant \ldots \leqslant a_n \leqslant \ldots \quad \text{и} \quad b_1 \leqslant b_2 \leqslant \ldots \leqslant b_n \leqslant \ldots$$

в множествах M_1 и M_2, каждая из которых имеет *точную верхнюю грань* — элементы c_1 и c_2. Нетрудно показать, что упорядоченная пара (c_1, c_2) есть точная верхняя грань записанной выше последовательности упорядоченных пар. ▶

Рассмотрим теперь конструкцию, во многом аналогичную прямому произведению алгебраических систем.

Пусть $\mathcal{A} = (A, \Omega, \Pi)$ — некоторая алгебраическая система, а X — произвольное множество. Распространим операции и отношения алгебраической системы $\mathcal{A}$ на множество A^X всех отображений из X в A.

Это осуществляется следующим образом:

1) для любого $n \geqslant 0$, любой операции $\omega \in \Omega^{(n)}$ и любых отображений $f_1, \ldots, f_n \in A^X$ полагаем

$$(f_1 \ldots f_n \omega)(x) = f_1(x) \ldots f_n(x)\omega,$$

где $x \in X$; тем самым определено отображение $f_1 \ldots f_n \omega \in A^X$ как результат применения операции ω к отображениям $f_1, \ldots, f_n$ (в частности, для нульарной операции $a \in A$ соответствующая нульарная операция на A^X есть отображение f_a, такое, что $f_a(x) = a$ для любого $x \in X$);

2) для любого $n \geqslant 1$, любого отношения $\pi \in \Pi^{(n)}$ и любых отображений $f_1, \ldots, f_n \in A^X$ полагаем

$$(f_1, \ldots, f_n) \in \pi \Leftrightarrow (\forall x \in X)((f_1(x), \ldots, f_n(x)) \in \pi).$$

Построенную таким образом алгебраическую систему с носителем A^X, однотипную с $\mathcal{A}$, обозначим $\mathcal{A}^X$. Докажем, что

в случае конечного множества $X = \{1, \ldots, n\}$ алгебраическая система $\mathcal{A}^X$ изоморфна алгебраической системе $\mathcal{A}^n$.

Действительно, в этом случае каждому отображению $f \in A^X$ можно однозначно сопоставить кортеж $(a_1, \ldots, a_n) \in A^n$ таким образом, что $a_i = f(i)$, $i = \overline{1, n}$. В то же время каждый кортеж $(a_1, \ldots, a_n) \in A^n$ однозначно определяет отображение $f \in A^X$, для которого $f(i) = a_i$, $i = \overline{1, n}$. Тем самым построена биекция φ множества A^X на множество A^n. Докажем, что она „сохраняет" операции и отношения (в том смысле, как это определено выше, см. **4.4**). Для произвольных отображений $f_1, \ldots, f_m \in A^X$ и произвольной операции $\omega \in \Omega^{(m)}$ имеем

$$\begin{aligned}\varphi(f_1 \ldots f_m\omega) &= ((f_1 \ldots f_m\omega)(1), \ldots, (f_1 \ldots f_m\omega)(n)) = \\ &= (f_1(1) \ldots f_m(1)\omega, \ldots, f_1(n) \ldots f_m(n)\omega) = \\ &= (f_1(1), \ldots, f_1(n)) \ldots (f_m(1), \ldots, f_m(n))\omega = \\ &= \varphi(f_1) \ldots \varphi(f_m)\omega,\end{aligned}$$

что и требовалось доказать.

„Сохранение" отношений доказывается аналогично. Тем самым *изоморфизм* $\mathcal{A}^X \cong \mathcal{A}^n$ доказан полностью.

Пример 4.11. а. Для *аддитивной группы действительных чисел* на множестве всех функций из $\mathbb{R}$ в $\mathbb{R}$ по приведенной выше конструкции строится аддитивная группа функций из $\mathbb{R}$ в $\mathbb{R}$. В этой группе сумма функций f и g есть функция $f + g$, такая, что для любого $x \in \mathbb{R}$ выполняется равенство $(f + g)(x) = f(x) + g(x)$. Функция, противоположная к f, определяется так: $(-f)(x) = -f(x)$. Нейтральный элемент есть нулевая функция $\mathbf{0}$, т.е. для любого $x \in \mathbb{R}$ имеет место $\mathbf{0}(x) = 0$.

Аналогично можно построить кольцо функций из $\mathbb{R}$ в $\mathbb{R}$ на базе кольца действительных чисел. Несмотря на то что действительные числа образуют поле, кольцо функций, как можно легко показать, полем не будет. Этот „отрицательный" результат вполне соответствует ранее доказанному результату, согласно которому прямое произведение полей не является полем.

б. Если $\mathcal{A} = (A, \leqslant)$ — упорядоченное множество, то упорядоченное множество всех функций из X в A строится так: полагаем $f \leqslant g$ тогда и только тогда, когда $f(x) \leqslant g(x)$ для любого $x \in X$. Если порядок на A индуктивен, то и порядок на A^X индуктивен, что доказывается так же, как в п. 4 теоремы 4.11.

4.6. Конечные булевы алгебры

Покажем применение понятия *прямого произведения алгебраических систем* к теории *булевых алгебр*. Мы докажем здесь интересный факт, состоящий в том, что *мощность* любой *конечной* булевой алгебры есть некоторая степень двойки. Отсюда будет следовать, например, что в конечной булевой алгебре может быть 1, 2, 8, 16, 32, 64 и т.д. элементов, но не может быть, скажем, 100 или 75 элементов. Чтобы доказать сформулированное утверждение о конечных булевых алгебрах, необходимо ввести некоторые вспомогательные определения и доказать некоторые утверждения. Напомним, что мы определили булеву алгебру $\mathbb{B}^n$ *булевых векторов размерности* n (см. пример 3.11). Эта алгебра есть не что иное, как n-*я декартова степень двухэлементной булевой алгебры* $\mathbb{B} = (\{0, 1\}, \vee, \wedge, 0, 1)$ (см. **4.5**).

Пусть $\mathcal{L} = \{L, \vee, \wedge, \mathbf{0}, \mathbf{1}\}$ — *симметричное полукольцо*. Рассмотрим произвольные элементы a, $b \in L$, такие, что $a \leqslant b$. Множество $[a, b] = \{x\colon a \leqslant x \leqslant b\}$ будем называть ***отрезком***, элемент a — ***левым***, а элемент b — ***правым концом отрезка***.

Замечание 4.5. Напомним, что в симметричном полукольце *нуль полукольца* является *наименьшим*, а *единица полукольца* — *наибольшим элементом* в данном полукольце относительно *естественного порядка* этого *идемпотентного полукольца*. Поэтому для любого элемента x симметричного полукольца справедливо неравенство $\mathbf{0} \leqslant x \leqslant \mathbf{1}$, и тем самым все симметричное полукольцо можно рассматривать как отрезок $[\mathbf{0}, \mathbf{1}]$ симметричного полукольца. #

Любой отрезок $[a, b]$ симметричного полукольца *замкнут относительно операций* $\vee$ и $\wedge$, но не является, вообще говоря, *подполукольцом* $\mathcal{L}$, так как не содержит $\mathbf{0}$ (если $a \neq \mathbf{0}$) и не содержит $\mathbf{1}$ (при $b \neq \mathbf{1}$). Но поскольку элемент a будет наименьшим, а элемент b — наибольшим элементом отрезка $[a, b]$, алгебра

$$([a, b],\ \vee,\ \wedge,\ a,\ b)$$

будет симметричным полукольцом с нулем a и единицей b, которое мы будем обозначать тоже через $[a, b]$.

Для произвольно фиксированного элемента a симметричного полукольца $\mathcal{L}$ зададим отображение Θ_a, сопоставляющее каждому $x \in L$ *упорядоченную пару* $(x \wedge a,\ x \vee a) \in L^2$. Так как $\mathbf{0} \leqslant x \wedge a \leqslant a$ и $a \leqslant x \vee a \leqslant \mathbf{1}$ для любого $x \in L$ (в силу свойств симметричного полукольца), то тем самым задано отображение Θ_a носителя полукольца $\mathcal{L}$ в декартово произведение отрезков $[\mathbf{0}, a] \times [a, \mathbf{1}]$:

$$\Theta_a\colon L \to [\mathbf{0}, a] \times [a, \mathbf{1}].$$

Каждый из отрезков есть симметричное полукольцо. В силу теоремы 4.11 их декартово произведение также является симметричным полукольцом.

Теорема 4.12. Пусть L — носитель симметричного полукольца $\mathcal{L} = (L,\ \vee,\ \wedge,\ \mathbf{0},\ \mathbf{1})$. Для любого $a \in L$ отображение Θ_a есть *мономорфизм* полукольца $\mathcal{L}$ в полукольцо $[\mathbf{0}, a] \times [a, \mathbf{1}]$.

◀ Докажем, что Θ_a — гомоморфизм. Имеем $\Theta_a(\mathbf{0}) = (\mathbf{0},\ a)$, и эта упорядоченная пара и является наименьшим элементом, т.е. нулем, полукольца $[\mathbf{0}, a] \times [a, \mathbf{1}]$. Точно так же $\Theta_a(\mathbf{1}) = (a, \mathbf{1})$ — наибольший элемент, т.е. единица того же полукольца.

Далее,

$$\Theta_a(x \vee y) = ((x \vee y) \wedge a,\ (x \vee y) \vee a) =$$
$$= ((x \wedge a) \vee (y \wedge a),\ (x \vee a) \vee (y \vee a)) = \Theta_a(x) \vee \Theta_a(y).$$

Аналогично доказывается, что

$$\Theta_a(x \wedge y) = \Theta_a(x) \wedge \Theta_a(y).$$

Итак, Θ_a — гомоморфизм $\mathcal{L}$ в $[\mathbf{0}, a] \times [a, \mathbf{1}]$.

Теперь надо доказать, что Θ_a — инъекция. Для этого нужно показать, что из равенства $\Theta_a(x) = \Theta_a(y)$ вытекает $x = y$. Если $\Theta_a(x) = \Theta_a(y)$, то верны равенства

$$x \wedge a = y \wedge a \text{ и } x \vee a = y \vee a, \tag{4.2}$$

так как равенство упорядоченных пар означает равенство их одноименных компонент.

Теперь, используя равенства (4.2) и аксиомы симметричного полукольца (см. **3.4**), получим

$$\begin{aligned} x = x \wedge (x \vee a) = x \wedge (y \vee a) = \\ = (x \wedge y) \vee (x \wedge a) = (y \wedge x) \vee (y \wedge a) = \\ = (y \vee y) \wedge (y \vee a) \wedge (x \vee y) \wedge (x \vee a) = \\ = y \wedge (y \vee a) \wedge (x \vee y) \wedge (x \vee a) = \\ = y \wedge (x \vee y) \wedge (y \vee a) = y \wedge (y \vee a) = y. \end{aligned}$$

Таким образом, если $\Theta_a(x) = \Theta_a(y)$, то $x = y$, и Θ_a — инъекция. ▶

Теорема 4.13. Если симметричное полукольцо

$$\mathcal{L} = (L, \vee, \wedge, \mathbf{0}, \mathbf{1})$$

есть булева алгебра, то для любого $a \in L$ полукольца $[\mathbf{0}, a]$ и $[a, \mathbf{1}]$ тоже булевы алгебры.

◀ Поскольку $[\mathbf{0}, a]$ и $[a, \mathbf{1}]$ — симметричные полукольца, то достаточно доказать, что в каждом из отрезков $[\mathbf{0}, a]$ и $[a, \mathbf{1}]$ любой элемент имеет *дополнение*.

Для произвольного $u \in [\mathbf{0}, a]$ определим элемент $\overline{u}_a$ равенством $\overline{u}_a = \overline{u} \wedge a$. Докажем, что этот элемент и есть дополнение u в полукольце $[\mathbf{0}, a]$. Для этого, как следует из свойства единственности дополнения в булевой алгебре, достаточно убедиться в том, что $\overline{u}_a \vee u = a$, а $\overline{u}_a \wedge u = \mathbf{0}$.

Действительно,

$$\overline{u}_a \vee u = (\overline{u} \wedge a) \vee u = \mathbf{1} \wedge (a \vee u).$$

Так как $u \leqslant a$, то $a \vee u = a$, и $\mathbf{1} \wedge (a \vee u) = \mathbf{1} \wedge a = a$. Итак, $\overline{u}_a \vee u = a$. Аналогично $\overline{u}_a \wedge u = (\overline{u} \wedge a) \wedge u = \mathbf{0}$. Таким образом, $\overline{u}_a$ действительно является дополнением элемента u в полукольце $[\mathbf{0}, a]$.

Теперь для произвольного $v \in [a, \mathbf{1}]$ определим элемент $\overline{v}^a$ равенством $\overline{v}^a = \overline{v} \vee a$. Как и выше, аналогично доказывается, что $\overline{v}^a \vee v = \mathbf{1}$, $\overline{v}^a \wedge v = a$. Следовательно, элемент $\overline{v}^a$ является дополнением v в полукольце $[a, \mathbf{1}]$. ▶

Теорема 4.14. Если в условиях теоремы 4.12 полукольцо $\mathcal{L}$ является булевой алгеброй, то Θ_a — изоморфизм булевых алгебр $\mathcal{L}$ и $[\mathbf{0}, a] \times [a, \mathbf{1}]$.

◀ В силу теоремы 4.12 достаточно доказать, что Θ_a — *эпиморфизм*, сохраняющий дополнение, т.е.

1) для любой пары (y, z), где $y \leqslant a$, $z \geqslant a$, существует элемент $x \in L$, такой, что $\Theta_a(x) = (y, z)$;

2) $\Theta_a(\overline{x}) = (\overline{x} \wedge a, \overline{x} \vee a) = \overline{\Theta_a(x)}$.

Докажем первое утверждение. Убедимся, что указанный в нем элемент x может быть определен равенством $x = y \vee (z \wedge \overline{a})$. Имеем

$$x \wedge a = [y \vee (z \wedge \overline{a})] \wedge a = (y \vee z) \wedge (y \vee \overline{a}) \wedge a = \\ = [(y \vee z) \wedge a] \wedge (y \vee \overline{a}).$$

Поскольку $y \leqslant a \leqslant z$, то

$$(y \vee z) \wedge a = (y \wedge a) \vee (z \wedge a) = y \vee a = a.$$

Следовательно, $x \wedge a = a \wedge (y \vee \overline{a}) = a \wedge y = y$ (так как $y \leqslant a$). Аналогично доказывается, что $x \vee a = z$. Таким образом, $\Theta_a(x) = (y, z)$.

Второе утверждение следует из того, что элемент $\overline{x} \wedge a$ есть дополнение элемента $u = x \wedge a$ в булевой алгебре $[\mathbf{0}, a]$, а элемент $\overline{x} \vee a$ — дополнение элемента $v = x \vee a$ в булевой алгебре $[a, \mathbf{1}]$.

Действительно, согласно теореме 4.13,

$$\overline{u}_a = (\overline{x \wedge a})_a = (\overline{x \wedge a}) \wedge a = \\ = (\overline{x} \vee \overline{a}) \wedge a = (\overline{x} \wedge a) \vee (\overline{a} \wedge a) = (\overline{x} \wedge a) \vee \mathbf{0} = \overline{x} \wedge a.$$

Согласно *принципу двойственности*, $\overline{v}^a = (\overline{x \vee a})^a = \overline{x} \vee a$.

Итак,

$$\overline{\Theta_a(x)} = ((\overline{x \wedge a})_a, (\overline{x \vee a})^a) = (\overline{x} \wedge a, \overline{x} \vee a) = \Theta_a(\overline{x}). \quad \blacktriangleright$$

В силу доказанных теорем имеем следующий результат.

Следствие 4.2. Любая булева алгебра изоморфна прямому произведению некоторых двух булевых алгебр. #

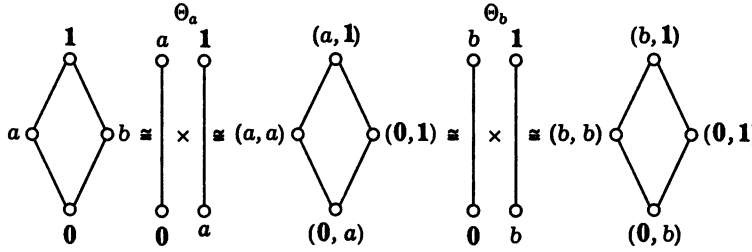


Рис. 4.4

На рис. 4.4 представлены все возможные способы представления четырехэлементной булевой алгебры (ее элементы обозначены $\mathbf{0}$, $\mathbf{1}$, a, b) в виде прямого произведения двух двухэлементных булевых алгебр. Эти представления определяются изоморфизмами Θ_a и Θ_b. Изоморфизм Θ_a есть изоморфизм исходной четырехэлеменной булевой алгебры на декартово произведение ее отрезков $[\mathbf{0}, a]$ и $[a, \mathbf{1}]$, каждый из которых изоморфен двухэлементной булевой алгебре $\mathbb{B}$. Вместе с тем декартово произведение указанных отрезков дает (см. **4.5**) четырехэлементную булеву алгебру, элементами которой служат

упорядоченные пары $(\mathbf{0}, a)$, (a, a), $(a, \mathbf{1})$ и $(\mathbf{0}, \mathbf{1})$, причем пара $(\mathbf{0}, a)$ будет нулем, а пара $(a, \mathbf{1})$ — единицей этой булевой алгебры, которая изоморфна исходной. Аналогично рассматривается изоморфизм Θ_b.

Интересно отметить, что могут быть заданы и изоморфизмы $\Theta_{\mathbf{0}}$ и $\Theta_{\mathbf{1}}$, но каждый из них определяет тривиальное разложение исходной булевой алгебры в виде ее прямого произведения на одноэлементную булеву алгебру, т.е. в виде $[\mathbf{0}, \mathbf{1}] \times [\mathbf{0}, \mathbf{0}]$ или в виде $[\mathbf{0}, \mathbf{1}] \times [\mathbf{1}, \mathbf{1}]$. Интерес представляют, следовательно, такие изоморфизмы Θ_a, где элемент a не является ни нулем, ни единицей исходной булевой алгебры.

Теперь, наконец, мы докажем основной результат.

Теорема 4.15. Любая конечная булева алгебра изоморфна булевой алгебре $\mathbb{B}^n$ для некоторого n.

◀ Доказательство проведем методом математической индукции по числу элементов булевой алгебры $\mathcal{A} = (A, \vee, \wedge, \mathbf{0}, \mathbf{1})$. *Одноэлементная булева алгебра* изоморфна нулевой степени алгебры $\mathbb{B}$ (см. замечание 4.4). Для самой алгебры $\mathbb{B}$, содержащей два элемента, доказывать нечего.

Пусть утверждение теоремы доказано для всех булевых алгебр с числом элементов, не большим некоторого $k \geqslant 2$. Рассмотрим произвольную булеву алгебру $\mathcal{A}$, содержащую $k+1$ элемент, т.е. $|A| = k + 1$, и пусть $a \in A$. Поскольку число элементов в данной алгебре не меньше трех, то, во-первых, $\mathbf{0} \neq \mathbf{1}$ (нуль совпадает с единицей только в одноэлементной булевой алгебре), а во-вторых, можно выбрать элемент a так, что $\mathbf{0} < a < \mathbf{1}$ (т.е. a отлично и от нуля и от единицы).

Тогда по теореме 4.14 $\mathcal{A} \cong [\mathbf{0}, a] \times [a, \mathbf{1}]$. Так как элемент a отличен от единицы алгебры $\mathcal{A}$, то отрезок $[\mathbf{0}, a]$ не содержит единицы $\mathbf{1}$, а так как $a \neq \mathbf{0}$, то отрезок $[a, \mathbf{1}]$ не содержит нуля алгебры $\mathcal{A}$. Следовательно, число элементов в каждом из отрезков не превышает k.

В соответствии с предположением индукции найдутся такие неотрицательные целые числа r и s, что $[\mathbf{0}, a] \cong \mathbb{B}^r$, а $[a, \mathbf{1}] \cong \mathbb{B}^s$. Поэтому $\mathcal{B} \cong \mathbb{B}^r \times \mathbb{B}^s \cong \mathbb{B}^{r+s}$. ▶

Следствие 4.3. Мощность конечной булевой алгебры есть некоторая степень двойки.

4.7. Многосортные алгебры

Среди всех *алгебр*, рассмотренных выше, несколько особое положение занимает *модуль* (и, как частный случай, линейное пространство). Модуль был определен как алгебра, в *сигнатуру* которой входят *бинарная операция* сложения и бесконечное (в общем случае) множество *унарных операций*. Относительно операции сложения модуль является *абелевой группой*, а множество унарных операций имеет структуру *кольца*. Каждая унарная операция рассматривается как умножение (левое или правое) элементов модуля на тот или иной элемент этого кольца. Проще было бы интерпретировать такое умножение как бинарную операцию, но это не вписывается в определение бинарной операции, поскольку *аргументами* умножения оказываются элементы двух разных алгебр. Модуль представляет собой как бы „симбиоз“ двух алгебр: абелевой группы и кольца. Это обстоятельство наводит на мысль ввести *алгебраические системы* с несколькими *носителями*. Так возникает идея многосортной алгебры.

Многосортная алгебра — это упорядоченная пара

$$((A_i)_{i \in I},\ \Omega),$$

где элементы *семейства множеств* $(A_i)_{i \in I}$ называют ***сортами***, а множество Ω, называемое ***многосортной сигнатурой***, состоит из ***многосортных операций*** — отображений вида

$$\omega\colon A_{i_1} \times \ldots \times A_{i_n} \to A_{i_0}.$$

Операцию ω называют при этом ***n-арной операцией типа*** $(i_0, i_1, \ldots, i_n)$.

Задавая конкретную многосортную алгебру, условимся, что будем записывать ее в виде кортежа, в котором сначала перечисляется семейство сортов (как правило, конечное), а затем — многосортные операции, образующие сигнатуру, причем после обозначения каждой операции пишется ее тип (в виде кортежа). Обратим еще раз внимание на то, что в типе операции первая компонента указывает на сорт результата операции, следующие компоненты суть номера сортов аргументов операции. Заметим, что тип *нульарной операции* в многосортной алгебре есть однокомпонентный кортеж: нульарная операция типа (i) — это фиксированный элемент, принадлежащий сорту A_i.

В свете определения многосортной алгебры *левый модуль* над кольцом $\mathcal{R}$ может быть описан как многосортная алгебра

$$\mathcal{M} = ((A_1, A_2),\ +(1,1,1),\ 0(1),\\ \oplus(2,2,2),\ \cdot(2,2,2),\ \mathbf{0}(2),\ \mathbf{1}(2),\ \circ(1,2,1)),$$

где $A_1 = G$ — носитель абелевой группы $\mathcal{G} = (G, +, 0)$, $A_2 = R$ — носитель кольца $\mathcal{R} = (R, \oplus, \cdot, \mathbf{0}, \mathbf{1})$, а $\circ\colon R \times G \to G$ — многосортная операция левого умножения элементов группы $\mathcal{G}$ на элементы кольца $\mathcal{R}$. Ее тип указывает на то, что первым аргументом является элемент кольца, вторым — элемент абелевой группы, а результат принадлежит абелевой группе.

Правый модуль описывается аналогично, но тип операции правого умножения элемента группы на элемент кольца будет равен $(1,1,2)$.

Условимся о следующей терминологии для модуля. Элементы группы $\mathcal{G}$ будем называть ***векторами модуля*** (элементами „первого сорта“), а элементы кольца $\mathcal{R}$ — ***скалярами*** данного ***модуля*** (элементами „второго сорта“).

Многосортные алгебры

$$\mathcal{A} = ((A_i)_{i\in I},\ \Omega) \quad \text{и} \quad \mathcal{B} = ((B_i)_{i\in I},\ \Sigma)$$

называют ***однотипными***, если существует биекция сигнатуры Ω на сигнатуру Σ, сопоставляющая n-арной операции некоторого типа n-арную операцию того же типа.

Сигнатуры однотипных многосортных алгебр и соответствующие друг другу элементы этих сигнатур (т.е. многосортные операции) обычно обозначают одинаково.

Пример 4.12. Алгебра

$$(V_3,\ \mathbb{R},\ +(1,1,1),\ \cdot(1,2,1),\ (\cdot)(2,1,1),\ \times(1,1,1),\ \circ(2,1,1,1))$$

имеет в качестве первого сорта множество V_3 свободных геометрических векторов (в трехмерном пространстве)[III], а в качестве второго сорта — множество действительных чисел.

Первая операция — бинарная операция $+$ (сложения векторов). Результатом операции является вектор, аргументами — также векторы, поэтому она имеет тип $(1,1,1)$.

Вторая операция — бинарная операция $\cdot$ (левого умножения вектора на число). Результат операции — вектор, первый аргумент — число, второй аргумент — вектор, поэтому тип операции — $(1,2,1)$.

Третья операция — бинарная операция $(\cdot)$ (скалярного умножения векторов). Ее результатом является число, и ее тип есть $(2,1,1)$.

Четвертая операция — бинарная операция $\times$ (векторного умножения векторов), а ее тип — $(1,1,1)$.

Последняя, пятая операция — тернарная операция смешанного умножения векторов. Результатом операции является число, и соответственно сорт операции — $(2,1,1,1)$. #

Обычная Ω-алгебра есть многосортная алгебра с одним сортом. Покажем, что и любая алгебраическая система может быть описана как многосортная алгебра. Сигнатура алгебраической системы кроме операций содержит и *отношения*.

Каждому отношению $\pi \in \Pi$ алгебраической системы $\mathcal{A} = (A,\ \Omega,\ \Pi)$ сопоставим *характеристическую функцию*: *ото-*

бражение $\xi_\pi\colon A^n \to \{0,1\}$, такое, что

$$\xi_\pi(a_1,\ldots,a_n) = 1 \Leftrightarrow (a_1,\ \ldots,\ a_n) \in \pi.$$

Тогда алгебраическую систему можно задать как многосортную алгебру с двумя сортами: носителем A и „логическим (булевым)“ сортом $\mathbb{B} = \{0,1\}$, а вместо каждого n-арного отношения рассматривать его характеристическую функцию как многосортную n-арную операцию типа $(2,\ 1,\ \ldots,\ 1)$ (номер 1 приписан сорту A, а номер 2 — сорту $\mathbb{B}$).

Рассмотрим теперь, как на многосортные алгебры распространяются понятия *подалгебры* и *гомоморфизма*.

Семейство $(B_i)_{i\in I}$ называют ***подсемейством*** семейства $(A_i)_{i\in I}$ (для одного и того же множества индексов I), если $(\forall i \in I)\ (B_i \subseteq A_i)$.

Семейство $(C_i)_{i\in I}$ называют ***объединением*** (***пересечением***) ***семейств*** $(A_i)_{i\in I}$ и $(B_i)_{i\in I}$, если $(\forall i \in I)\ (C_i = A_i \cup B_i)$ (соответственно $(\forall i \in I)\ (C_i = A_i \cap B_i)$).

Пусть $\mathcal{A} = ((A_i)_{i\in I},\ \Omega)$ — многосортная алгебра и $(B_i)_{i\in I}$ — подсемейство семейства $(A_i)_{i\in I}$.

Подсемейство $(B_i)_{i\in I}$ называют ***Ω-замкнутым*** (или замкнутым относительно операций многосортной сигнатуры Ω), если $b_{i_1}\ldots b_{i_n}\omega \in B_{i_0}$ для всякой n-арной операции $\omega \in \Omega$ типа $(i_0,\ i_1,\ \ldots,\ i_n)$ (при произвольных n, i_0, i_1, $\ldots$, i_n) и любых $b_{i_1} \in B_{i_1},\ \ldots,\ b_{i_n} \in B_{i_n}$.

Для Ω-замкнутого подсемейства $(B_i)_{i\in I}$ можно тогда определить однотипную с $\mathcal{A}$ многосортную алгебру $\mathcal{B} = ((B_i)_{i\in I},\ \Omega)$, которую называют ***многосортной подалгеброй*** алгебры $\mathcal{A}$.

Можно показать, что пересечение любого семейства Ω-замкнутых подсемейств есть также Ω-замкнутое подсемейство. Следовательно, для произвольного подсемейства $(C_i)_{i\in I}$ семейства $(A_i)_{i\in I}$ существует наименьшее относительно включения Ω-замкнутое подсемейство $(B_i)_{i\in I}$, содержащее подсемейство $(C_i)_{i\in I}$ и называемое тогда ***Ω-замыканием*** этого ***подсемейства***. Если Ω-замыкание подсемейства $(C_i)_{i\in I}$ совпадает со

всем семейством $(A_i)_{i\in I}$, то первое подсемейство называют ***системой образующих многосортной алгебры*** $\mathcal{A}$.

Семейство отображений $(h_i)_{i\in I}$, где $h_i\colon A_i \to B_i$, $i \in I$, для однотипных многосортных алгебр

$$\mathcal{A} = ((A_i)_{i\in I},\ \Omega) \quad \text{и} \quad \mathcal{B} = ((B_i)_{i\in I},\ \Omega)$$

называют ***многосортным гомоморфизмом*** алгебры $\mathcal{A}$ в алгебру $\mathcal{B}$, если для каждой n-арной операции $\omega \in \Omega$ типа $(i_0, i_1, \ldots, i_n)$ (при произвольных n, i_0, i_1, $\ldots$, i_n) и любых $x_{i_1} \in A_{i_1}, \ldots, x_{i_n} \in A_{i_n}$

$$h_{i_0}(x_{i_1} \ldots x_{i_n}\omega) = h_{i_1}(x_{i_1}) \ldots h_{i_n}(x_{i_n})\omega.$$

Многосортный гомоморфизм, все отображения которого суть биекции, называют ***многосортным изоморфизмом*** алгебры $\mathcal{A}$ на алгебру $\mathcal{B}$. Очевидно, что если существует изоморфизм $\mathcal{A}$ на $\mathcal{B}$, то существует и изоморфизм $\mathcal{B}$ на $\mathcal{A}$. Алгебры $\mathcal{A}$ и $\mathcal{B}$ называют в этом случае ***изоморфными многосортными алгебрами***.

Аналогично обычным алгебрам в многосортных алгебрах вводят понятия многосортного моно- и эпиморфизма.

В качестве простого примера сошлемся на понятие ***линейного оператора*** (для линейных пространств над одним и тем же полем), известного из курса линейной алгебры [IV]. Оно является не чем иным, как многосортным гомоморфизмом, где отображение носителя одного поля в носитель другого является тождественным отображением.

Более сложными будут следующие примеры.

Пример 4.13. **а.** Рассмотрим модуль $\mathcal{L}_1$ над кольцом $\mathbb{Z}$ целых чисел и модуль $\mathcal{L}_2$ над кольцом вычетов по модулю k (для некоторого целого $k \geqslant 2$). Отображение h_1 определим как произвольный гомоморфизм аддитивной группы векторов первого модуля в аддитивную группу векторов второго, т.е. для любых векторов x, y первого модуля имеет место равенство

$$h_1(x+y) = h_1(x) + h_1(y).$$

Отображение h_2 зададим как *канонический гомоморфизм* кольца $\mathbb{Z}$ в фактор-кольцо $\mathbb{Z}_k$. Тогда, если для любого вектора x первого модуля и любого целого α выполняется равенство

$$h_1(\alpha \circ x) = h_2(\alpha) \circ h_1(x), \tag{4.3}$$

семейство (h_1, h_2) есть многосортный гомоморфизм первого модуля во второй.

В частности, пусть группа векторов модуля $\mathcal{L}_1$ есть n-я *декартова степень аддитивной группы целых чисел* (т.е. группа целочисленных арифметических векторов размерности n по операции сложения); группу же векторов модуля $\mathcal{L}_2$ зададим как n-ю декартову степень *аддитивной группы вычетов* по модулю k (т.е. как группу по операции сложения по модулю k целочисленных арифметических векторов размерности n, каждая компонента которых принимает значения от 0 до $k-1$). Обозначая через $[x]_k$ остаток от деления x на k ($x \in \mathbb{Z}$), гомоморфизм h_1 зададим равенством $h_1(\boldsymbol{x}) = [\boldsymbol{x}]_k$, где $\boldsymbol{x} = (x_1, \ldots, x_n) \in \mathbb{Z}^n$, а $[\boldsymbol{x}]_k = ([x_1]_k, \ldots, [x_n]_k) \in \{0, 1, \ldots, k-1\}^n$. По определению, $h_2(\alpha) = [\alpha]_k$, $\alpha \in \mathbb{Z}$. Умножение вектора на скаляр в каждом из модулей определим стандартно через умножение в соответствующем кольце, т.е. для первого модуля положим

$$\alpha \circ \boldsymbol{x} = \alpha \cdot \boldsymbol{x} = (\alpha \cdot x_1, \ldots, \alpha \cdot x_n),$$

где $\alpha \in \mathbb{Z}$, $\boldsymbol{x} \in \mathbb{Z}^n$, а для второго —

$$\alpha \circ \boldsymbol{x} = \alpha \odot_k \boldsymbol{x} = (\alpha \odot_k x_1, \ldots, \alpha \odot_k x_n),$$

где $\alpha \in \{0, 1, \ldots, k-1\}$, $\boldsymbol{x} \in \{0, 1, \ldots, k-1\}^n$. Тогда

$$h_1(\alpha \circ \boldsymbol{x}) = [\alpha \cdot \boldsymbol{x}]_k = [\alpha]_k \odot_k [\boldsymbol{x}]_k = h_2(\alpha) \circ h_1(\boldsymbol{x}),$$

и равенство (4.3) выполняется.

В то же время если бы *гомоморфизм* h_1 мы задали как *проектирующий*, т.е. положили бы

$$h_1(\boldsymbol{x}) = h_1(x_1, \ldots, x_n) = x_i$$

(для некоторого фиксированного i, $1 \leqslant i \leqslant n$), тогда $h_1(\alpha \circ \boldsymbol{x}) = \alpha \cdot x_i$, а

$$h_2(\alpha) \circ h_1(\boldsymbol{x}) = [\alpha]_k \odot_k [x_i]_k = [\alpha \cdot x_i]_k \neq \alpha \cdot x_i.$$

Таким образом, в этом случае семейство (h_1, h_2) не будет многосортным гомоморфизмом, при том что каждое отображение семейства будет гомоморфизмом соответственно группы и кольца.

Заметим, что, если группы векторов рассмотренных модулей взять соответственно как аддитивную группу целых чисел и аддитивную группу вычетов по модулю k, многосортный гомоморфизм превращается в обычный канонический гомоморфизм кольца $\mathbb{Z}$.

б. Пусть векторами левого модуля $\mathcal{M}_1$ являются векторы какого-то конечномерного линейного пространства $\mathcal{L}$ над полем действительных чисел [IV], а скалярами модуля служат линейные операторы, действующие в этом пространстве. Пусть *размерность* пространства $\mathcal{L}$ равна n. В качестве векторов модуля $\mathcal{M}_2$ рассмотрим линейное пространство действительных *матриц-столбцов типа* $n \times 1$, а скаляров — *квадратные матрицы* n-го порядка (см. пример 2.12.г). Зафиксировав в пространстве $\mathcal{L}$ *базис*, зададим отображение h_1 так, чтобы оно каждому вектору из $\mathcal{L}$ сопоставляло столбец его *координат* в данном базисе, а отображение h_2 пусть для каждого линейного оператора, действующего в пространстве $\mathcal{L}$, дает его матрицу в том же базисе. Основываясь на известных результатах линейной алгебры [IV], можно утверждать, что тем самым определен многосортный изоморфизм модуля $\mathcal{M}_1$ на модуль $\mathcal{M}_2$.

Замечание 4.6. Как и для обычного „односортного“ случая, в теории многосортных алгебр можно доказать теоремы, аналогичные теоремам 4.3 и 4.4 и связывающие понятия многосортного гомоморфизма и ***многосортной фактор-алгебры***. Не рассматривая этот вопрос подробно, заметим, что переход от многосортной алгебры $\mathcal{A} = ((A_i)_{i \in I}, \Omega)$ к ее фактор-алгебре

осуществляется на основе ***многосортной конгруэнции*** — такого семейства отношений эквивалентности $(\rho_i \subseteq A_i^2)_{i\in I}$, что для любых попарно эквивалентных элементов a_i, $b_i \in A_i$, $i \in I$, т.е. таких, что $a_i\ \rho_i\ b_i$, имеет место также и эквивалентность $(a_{i_1} \dots a_{i_n}\omega)\ \rho_{i_0}\ (b_{i_1} \dots b_{i_n}\omega)$, какова бы ни была n-арная операция $\omega \in \Omega$ типа $(i_0, i_1, \dots, i_n)$. Тогда любая такая операция может быть распространена на семейство *фактор-множеств* $((A_i/\rho_i)_{i\in I}, \Omega)$ согласно равенству

$$[a_{i_1} \dots a_{i_n}\omega]_{\rho_{i_0}} = [a_{i_1}]_{\rho_{i_1}} \dots [a_{i_n}]_{\rho_{i_n}}\omega.$$

Определенную таким образом многосортную алгебру называют фактор-алгеброй исходной алгебры $\mathcal{A} = ((A_i)_{i\in I},\ \Omega)$ относительно многосортной конгруэнции $(\rho_i)_{i\in I}$. #

Многосортные алгебры широко используются в современном теоретическом программировании*.

Вопросы и задачи

4.1. Показать, что отношение $\leqslant$ (см. пример 4.1[б]) на носителе поля $\mathcal{F}$ является отношением *линейного порядка*, т.е. для любых двух элементов $a, b \in F$ имеет место $a \leqslant b$ или $b \leqslant a$.

4.2. Доказать следующие изоморфизмы: $\mathbb{B}^n \cong \mathcal{D}_{q_1 \dots q_n} \cong \mathcal{S}_A$, где q_1, ..., q_n — попарно различные простые числа, а $|A| = n$ (см. примеры 3.3.б, 3.9 и 3.11).

4.3. Установить, сохраняет ли отношение, введенное в примере 4.4.а, на множестве $\mathbb{Z}$ отношение делимости: $m \mid n$ (m делит n).

4.4. На множестве всех отображений множества M в себя определено отношение τ: $f\ \tau\ g$, если и только если $R(f) = R(d)$.

*Об использовании понятия многосортной алгебры в программировании см.: *Гоген Дж.А., Мезегер Ж.*

Доказать, что в общем случае это отношение, будучи эквивалентностью, не является конгруэнцией относительно композиции отношений. Построить пример.

4.5. Имеет ли место изоморфизм $\mathbb{Z}_k^n \cong \mathbb{Z}_{k^n}$?

4.6. Определить группы движений цилиндра и тора, рассуждая так же, как и при решении задачи 2.20. Доказать, что первая изоморфна группе $\mathbb{R}^2/\mathbb{Z}$, а вторая — $\mathbb{R}^2/\mathbb{Z}^2 \cong (\mathbb{R}/\mathbb{Z})^2 \cong$ $\cong \mathbb{S}^1 \times \mathbb{S}^1$.

4.7. Доказать, что полукольцо бинарных отношений на n-элементном множестве $\{a_1, \ldots, a_n\}$ изоморфно полукольцу квадратных матриц порядка n над полукольцом $\mathcal{B}$ (см. **3**).

4.8. Найти все разложения булевой алгебры $\mathbb{B}^3$ в виде

$$\mathbb{B}^3 \cong \mathbb{B} \times \mathbb{B}^2 \cong \mathbb{B}^2 \times \mathbb{B}.$$

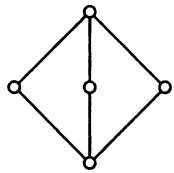


Рис. 4.5

4.9. Доказать, что любой гомоморфный образ решетки, диаграмма Хассе которой изображена на рис. 4.5, изоморфен либо ей самой, либо одноэлементной решетке.

4.10. Найти все гомоморфные образы решетки, диаграмма Хассе которой изображена на рис. 3.5, которые не изоморфны ей самой, а также не изоморфны одноэлементной решетке.

5. ТЕОРИЯ ГРАФОВ

Неформально граф можно рассматривать как множество точек и соединяющих эти точки линий со стрелками или без них.

Первой работой теории графов как математической дисциплины считают статью Эйлера (1736 г.), в которой рассматривалась задача о Кёнингсбергских мостах. Эйлер показал, что нельзя обойти семь городских мостов и вернуться в исходную точку, пройдя по каждому мосту ровно один раз. Следующий импульс теория графов получила спустя почти 100 лет с развитием исследований по электрическим сетям, кристаллографии, органической химии и другим наукам.

С графами, сами того не замечая, мы сталкиваемся постоянно. Например, графом является схема линий метрополитена. Точками на ней представлены станции, а линиями — пути движения поездов. Исследуя свою родословную и возводя ее к далекому предку, мы строим так называемое генеалогическое древо. И это древо — граф.

Графы служат удобным средством описания связей между объектами. Ранее мы уже использовали графы как способ наглядного представления конечных *бинарных отношений* (см. **1.3**). Но граф используют отнюдь не только как иллюстрацию. Например, рассматривая граф, изображающий сеть дорог между населенными пунктами, можно определить маршрут проезда от пункта А до пункта Б. Если таких маршрутов окажется несколько, хотелось бы выбрать в определенном смысле оптимальный, например самый короткий или самый безопасный. Для решения задачи выбора требуется проводить определенные вычисления над графами. При решении подобных задач удобно использовать алгебраическую технику, да и само понятие графа необходимо формализовать.

Методы теории графов широко применяются в дискретной математике. Без них невозможно обойтись при анализе и синтезе различных дискретных преобразователей: функциональных блоков компьютеров, комплексов программ и т.д.

В настоящее время теория графов охватывает большой материал и активно развивается. При ее изложении ограничимся только частью результатов и основной акцент сделаем на описании и обосновании некоторых широко распространенных алгоритмов анализа графов, которые применяются в теории формальных языков.

5.1. Основные определения

Графы, как уже отмечалось в примерах, есть способ „визуализации“ связей между определенными объектами. Связи эти могут быть „направленными“, как, например, в генеалогическом древе, или “ненаправленными“ (сеть дорог с двусторонним движением). В соответствии с этим в теории графов выделяют два основных типа графов: ориентированные (или направленные) и неориентированные.

Построение математического определения графа осуществляется путем формализации и „объектов“, и „связей“ как элементов некоторых (как правило, конечных) множеств.

Основные понятия для неориентированных и ориентированных графов удобно вводить „параллельно“. Такое изложение позволит наглядно сопоставить соответствующие понятия.

Неориентированные графы	**Ориентированные графы**
Неориентированный граф G задается двумя множествами $G = (V, E)$, где V — конечное множество, элементы которого называют ***вершинами*** или ***узлами***; E —	***Ориентированный граф*** G задается двумя множествами $G = (V, E)$, где V — конечное множество, элементы которого называют вершинами или узлами; E —

множество ***неупорядоченных пар*** на V, т.е. подмножество множества двухэлементных подмножеств V, элементы которого называют ***ребрами***. Для каждого ребра $\{u, v\} \in E$ считаем, что u и v — различные вершины.	множество ***упорядоченных пар*** на V, т.е. подмножество множества $V \times V$, элементы которого называют ***дугами***.
Если ребро $e = \{u, v\}$, то говорят, что ребро e соединяет вершины u и v, и обозначают это $u \mapsto v$; если необходимо, указывают имя графа G: $u \mapsto_G v$.	Если дуга $e = (u, v)$, то говорят, что дуга e ведет из вершины u в вершину v, и обозначают это $u \to v$; если необходимо, указывают имя графа G: $u \to_G v$.
Вершины u и v, соединеные ребром $(u \mapsto v)$, называют ***смежными***, а также ***концами ребра*** $\{u, v\}$. Если $u \mapsto v$, говорят, что вершины u и v связаны ***отношением непосредственной достижимости***.	Вершины u и v, такие, что из вершины u в вершину v ведет дуга $(u \to v)$, называют ***смежными***, причем u называют ***началом***, а v — ***концом дуги*** (u, v). Дугу, начало и конец которой есть одна и та же вершина, называют ***петлей***. Если $u \to v$, то говорят, что вершины u и v связаны ***отношением непосредственной достижимости***.
Ребро e называют ***инцидентным*** вершине v, если она является одним из его концов.	Дугу (u,v) называют ***заходящей*** в вершину v и ***исходящей*** из вершины u. Дугу называют ***инцидентной*** вершине v, если она заходит в v или исходит из v.
Степенью вершины v называют число $\deg v$ всех инцидентных ей ребер.	***Полустепенью захода*** вершины v называют число $\deg^-(v)$ заходящих в нее дуг, а ***полустепенью исхода*** вершины v — число $\deg^+(v)$ исходящих из нее дуг. ***Степень вершины*** v, обозначаемая $\deg(v)$, — это сумма полустепеней захода и исхода.

Для вершины v множество $$\Gamma(v) = \{x\colon x \mapsto v\}$$ называют множеством смежных с v вершин. Справедливо равенство $$\deg(v) = \lvert\Gamma(v)\rvert.$$	Для вершины v множество $$\Gamma(v) = \{x\colon v \to x\}$$ называют множеством ***преемников*** вершины v, а множество $$\Gamma^{-1}(v) = \{x\colon x \to v\}$$ — множеством ***предшественников*** вершины v. Справедливы равенства $$\deg^+(v) = \lvert\Gamma(v)\rvert,\ \deg^-(v) = \lvert\Gamma^{-1}(v)\rvert.$$
Цепь в неориентированном графе G — это последовательность вершин (конечная или бесконечная) $v_0, v_1, \ldots, v_n, \ldots$, такая, что $v_i \mapsto v_{i+1}$ для любого i, если v_{i+1} существует. (Под конечной последовательностью понимается ***кортеж*** вершин.)	***Путь*** в ориентированном графе G — это последовательность вершин (конечная или бесконечная) $v_0, v_1, \ldots, v_n, \ldots$, такая, что $v_i \to v_{i+1}$ для любого i, если v_{i+1} существует.
Для конечной цепи $v_0, v_1, \ldots, v_n$ число n $(n \geqslant 0)$ называют ***длиной цепи***. Таким образом, длина цепи есть число ее ребер, т.е. всех ребер, соединяющих вершины v_i и v_{i+1} $(i = \overline{0, n-1})$. Цепь длины 0 — это произвольная вершина графа.	Для конечного пути $v_0, v_1, \ldots, v_n$ число n называют ***длиной пути*** $(n \geqslant 0)$. Тем самым длина пути есть число его дуг, т.е. всех дуг, которые ведут из вершины v_i в вершину v_{i+1} $(i = \overline{0, n-1})$. Путь длины 0 — это произвольная вершина графа.
Говорят, что ***вершина*** v неориентированного графа G ***достижима*** из вершины u этого графа и обозначают $u \rightleftarrows^* v$, если существует цепь $v_0, v_1, \ldots, v_n$, такая, что $u = v_0$, $v_n = v$ (при этом говорят также, что данная цепь соединяет вершины u и v, которые называют ***концами цепи***). Таким образом, задано	Говорят, что ***вершина*** v ориентированного графа G ***достижима*** из вершины u этого графа и обозначают $u \Rightarrow^* v$, если существует путь $v_0, v_1, \ldots, v_n$, такой, что $u = v_0$, $v = v_n$ (при этом говорят, что данный путь ведет из вершины u в вершину v, называя первую вершину ***началом***, а вторую — ***концом***

отношение достижимости $\mapsto\!\!\!\!\mapsto^*$ в неориентированном графе. Оно является *рефлексивно-транзитивным замыканием* отношения $\mapsto$ непосредственной достижимости.	данного *пути*). Таким образом, задано ***отношение достижимости*** $\Rightarrow^*$ в ориентированном графе. Оно является *рефлексивно-транзитивным замыканием* отношения $\rightarrow$ непосредственной достижимости.
Отношение достижимости в неориентированном графе *рефлексивно, симметрично* и *транзитивно*, т.е. является *отношением эквивалентности.*	Отношение достижимости в ориентированном графе рефлексивно и транзитивно, но в общем случае не *антисимметрично*: если две вершины ориентированного графа достижимы одна из другой, то из этого вовсе не следует, что они совпадают. Таким образом, отношение достижимости в ориентированном графе есть *отношение предпорядка.*
Если существует цепь ненулевой длины, соединяющая u и v, то пишут $$u \mapsto\!\!\!\!\mapsto^+ v.$$ Если необходимо явно указать длину цепи, то пишут $$u \mapsto\!\!\!\!\mapsto^n v$$ и говорят, что существует цепь длины n, соединяющая u и v.	Если существует путь ненулевой длины, ведущий из u в v, то пишут $$u \Rightarrow^+ v.$$ Если необходимо явно указать длину пути, то пишут $$u \Rightarrow^n v$$ и говорят, что существует путь длины n, ведущий из u в v.
Простая цепь — это цепь, все вершины которой, кроме, быть может, первой и последней, попарно различны и все ребра попарно различны.	***Простой путь*** — это путь, все вершины которого, кроме, быть может, первой и последней, попарно различны.
Простую цепь ненулевой длины с совпадающими концами называют ***циклом.***	Простой путь ненулевой длины, начало и конец которого совпадают, называют ***контуром.***

Произвольную цепь ненулевой длины с совпадающими концами, все ребра которой попарно различны, будем называть ***замкнутой цепью.***	Произвольный путь ненулевой длины, начало и конец которого совпадают, будем называть ***замкнутым путем.***
Неориентированный граф, не содержащий циклов, называют ***ациклическим графом.***	Ориентированный граф, не содержащий контуров, называют ***бесконтурным графом.***

Замечание 5.1. Требование, чтобы все ребра простой цепи неориентированного графа были попарно различными, существенно. Если его снять, то цепь вида u, v, u, где $u \vdash\!\dashv v$, будет циклом, в котором одно и то же ребро $\{u, v\}$ проходится дважды в противоположных направлениях, хотя такую цепь естественно циклом не считать. Не будет эта цепь в соответствии с принятой терминологией и замкнутой цепью.

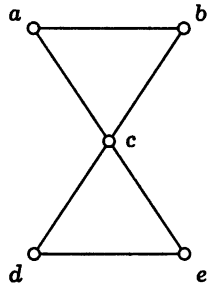


Рис. 5.1

В неориентированном графе на рис. 5.1 цепь $a \vdash\!\dashv b \vdash\!\dashv c \vdash\!\dashv d \vdash\!\dashv e \vdash\!\dashv c \vdash\!\dashv a$ является примером замкнутой цепи.

Замечание 5.2. В общем случае в ориентированном графе пересечение множества преемников $\Gamma(v)$ вершины v и множества $\Gamma^{-1}(v)$ ее предшественников будет не пусто, если есть петля (v, v): $\Gamma(v) \cap \Gamma^{-1}(v) \neq \varnothing$.

Пример 5.1. Рассмотрим неориентированный граф, изображенный на рис. 5.2. Он задается множеством вершин

$$V = \{v_1, v_2, v_3, v_4, v_5, v_6, v_7\}$$

и множеством неупорядоченных пар

$$E = \{\{v_1,v_2\}, \{v_1,v_3\}, \{v_2,v_3\}, \{v_3,v_4\}, \{v_5,v_6\}\}.$$

В этом графе последовательность вершин v_1, v_3, v_4 есть простая цепь, а последовательность v_1, v_3, v_2, v_1, v_3, v_4 — цепь, не являю-

щаяся простой, поскольку в ней есть совпадающие ребра. Последовательность вершин v_3, v_1, v_2, v_4 не является цепью, поскольку в графе нет ребра $\{v_2, v_4\}$. Последовательность v_1, v_3, v_2, v_1 есть цикл, а последовательность v_4, v_3, v_1, v_2, v_3, v_4 — цепь с совпадающими концами, но не цикл, поскольку эта цепь не является простой. Эта цепь не будет и замкнутой, так как в ней есть повторяющееся ребро $\{v_3, v_4\}$.

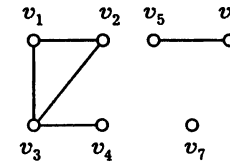


Рис. 5.2

Степени вершин графа следующие: $\deg(v_1) = \deg(v_2) = 2$, $\deg(v_3) = 3$, $\deg(v_4) = \deg(v_5) = \deg(v_6) = 1$, $\deg(v_7) = 0$.

Вершины v_1, v_2, v_3, v_4 попарно достижимы ($v_i \rightleftarrows^* v_j$, $i, j \in \in \{1, 2, 3, 4\}$) и образуют класс эквивалентности по отношению достижимости. Для вершин v_5 и v_6 имеет место $v_5 \rightleftarrows^* v_6$, и они также образуют класс эквивалентности. Заметим, что вершина v_7, по определению, образует цепь длины 0 и эквивалентна по отношению достижимости только самой себе.

Пример 5.2. Обратимся к ориентированному графу, изображенному на рис. 5.3. Он задается множеством вершин $V = \{v_1, v_2, v_3, v_4, v_5, v_6\}$ и множеством дуг

$$E = \{(v_1, v_2),\ (v_1, v_3),\ (v_2, v_1),\ (v_2, v_3),\ (v_2, v_4),\ (v_3, v_1),\ (v_3, v_4),\ (v_5, v_6),\ (v_6, v_4)\}.$$

В этом ориентированном графе последовательность вершин v_1, v_2, v_3, v_4 есть простой путь, а последовательность вершин v_1, v_2, v_3, v_1, v_3, v_4 — путь, не являющийся простым, поскольку в нем, например, два раза встречается вершина v_3, не служащая началом и концом пути.

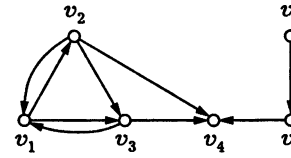


Рис. 5.3

Последовательность вершин v_3, v_1, v_2, v_3 есть контур, а последовательность v_3, v_1, v_2, v_1, v_3 — замкнутый путь, но не

контур, поскольку вершина v_1, не являющаяся началом пути, встречается два раза. Последовательность вершин v_1, v_3, v_4, v_6 не задает путь, так как в рассматриваемом ориентированном графе нет дуги (v_4, v_6).

Полустепени захода, полустепени исхода и степени у вершин следующие: у v_1 и v_3 — $\deg^-(v_1) = \deg^-(v_3) = 2$, $\deg^+(v_1) = \deg^+(v_3) = 2$, $\deg(v_1) = \deg(v_3) = 4$; у v_2 — $\deg^-(v_2) = 1$, $\deg^+(v_2) = 3$, $\deg(v_2) = 4$; у v_4 — $\deg^-(v_4) = 3$, $\deg^+(v_4) = 0$, $\deg(v_4) = 3$; у v_5 — $\deg^-(v_5) = 0$, $\deg^+(v_5) = 1$, $\deg(v_5) = 1$; у v_6 — $\deg^-(v_6) = 1$, $\deg^+(v_6) = 1$, $\deg(v_6) = 2$. #

Отношение достижимости в неориентированных и ориентированных графах обладает следующим важным свойством.

Теорема 5.1. Для любой цепи, соединяющей две вершины неориентированного графа, существует простая цепь, соединяющая те же вершины. Для любого пути, ведущего из вершины u в вершину v ориентированного графа, существует простой путь, ведущий из u в v.

◀ Проведем доказательство для неориентированного графа (для ориентированного графа доказательство проводится аналогично).

Рис. 5.4

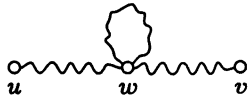


Рис. 5.5

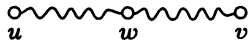


Рис. 5.6

Пусть вершины u и v неориентированного графа таковы, что $u \rightleftarrows^* v$. Если эти вершины являются концами цепи нулевой длины, то утверждение теоремы тривиально. Пусть $u \rightleftarrows^+ v$, т.е. существует цепь ненулевой длины, соединяющая u и v. Рассмотрим какую-либо из таких цепей (рис. 5.4). Обозначим ее C. Если цепь C простая, то доказывать нечего. Пусть существует внутренняя (не совпадающая ни с одним из концов) вершина w цепи C, которая повторяется в этой цепи, т.е. $u \rightleftarrows^* w \rightleftarrows^+ w \rightleftarrows^* v$. Это

значит, что вершина w содержится в некоторой цепи C' ненулевой длины (подцепи цепи C) с совпадающими концами (рис. 5.5). Удалим все вершины и ребра цепи C', кроме вершины w (служащей ее началом и концом одновременно). После этого получим новую цепь C_1, соединяющую вершины u и v (рис. 5.6), в которой число повторений вершины w будет по крайней мере на единицу меньше, чем в цепи C. Если цепь C_1 простая, то утверждение доказано. В противном случае поступаем с ней так же, как и с цепью C. В силу конечности множества вершин и ребер графа после конечного числа шагов получим простую цепь, соединяющую вершины u и v. ▶

Следствие 5.1. Если вершина неориентированного графа содержится в некоторой замкнутой цепи, то она содержится и в некотором цикле. Если вершина ориентированного графа содержится в некотором замкнутом пути, то она содержится и в некотором контуре.

Замечание 5.3. Следствие 5.1 перестает быть верным для произвольной цепи с совпадающими концами. Например, для неориентированного графа, состоящего из двух вершин v_1, v_2 и единственного ребра (v_1, v_2) цепь v_1, v_2, v_1 с совпадающими концами не содержит цикла. #

Перейдем теперь к понятию подграфа. Формулируется это понятие одновременно для неориентированных и ориентированных графов (с учетом различий в терминологии).

Определение 5.1. Неориентированный (ориентированный) граф $G_1 = (V_1, E_1)$ называют ***подграфом*** неориентированного (ориентированного) графа $G = (V, E)$, если $V_1 \subseteq V$ и $E_1 \subseteq E$.

Будем использовать обозначение $G_1 \subseteq G$, аналогичное обозначению включения для множеств.

Замечание 5.4. Так как в определении 5.1 пара $G_1 = (V_1, E_1)$ есть неориентированный (ориентированный) граф, то для лю-

бого ребра $\{u, v\} \in E_1$ (дуги $(u, v) \in E_1$) предполагается, конечно, что $u, v \in V_1$, поскольку иначе пару (V_1, E_1) нельзя будет считать неориентированным (ориентированным) графом. #

Если хотя бы одно из указанных двух включений в определении 5.1 строгое, то G_1 называют ***собственным подграфом*** графа G; если $V_1 = V$, то G_1 называют ***остовным подграфом*** графа G.

Подграф G_1 неориентированного (ориентированного) графа G называют ***подграфом, порожденным множеством вершин*** $V_1 \subseteq V$, если каждое ребро (дуга) тогда и только тогда принадлежит $E_1 \subseteq E$, когда его (ее) концы принадлежат V_1. Часто в случае, если множество вершин V_1 подразумевается, говорят просто о ***порожденном подграфе***.

Отметим, что подграф графа G, порожденный множеством вершин V_1, в отличие от произвольного подграфа графа G с множеством вершин V_1, должен включать все ребра (дуги), концы которых принадлежат множеству V_1.

Подграф $G_1 \subseteq G$ называют ***максимальным подграфом***, обладающим данным свойством P, если он не является собственным подграфом никакого другого подграфа графа G, обладающего свойством P.

Например, на рис. 5.7 подграфы G_1, G_2, G_3 являются максимальными ациклическими подграфами графа G. Отметим, что они также являются собственными и остовными подграфами указанного графа.

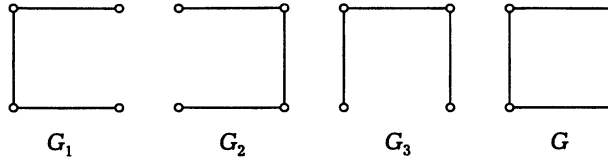


Рис. 5.7

Следующее важное понятие снова введем параллельно для рассматриваемых двух видов графов.

Неориентированные графы	Ориентированные графы
Неориентированный граф называют ***связным***, если любые две его вершины u и v соединены цепью ($u \rightleftarrows^* v$).	Ориентированный граф называют ***связным***, если для любых двух его вершин u, v вершина v достижима из вершины u или вершина u достижима из вершины v ($u \Rightarrow^* v$ или $v \Rightarrow^* u$).
Компонента связности (или просто ***компонента***) неориентированного графа — это его максимальный связный подграф.	***Компонента связности*** (или просто ***компонента***) ориентированного графа — это максимальный связный подграф.

В неориентированном графе две вершины, соединенные цепью, связаны отношением достижимости, которое является эквивалентностью. Поэтому компонента такого графа — это подграф, порожденный некоторым классом эквивалентности вершин по отношению достижимости.

Поскольку каждая компонента неориентированного графа порождается некоторым классом эквивалентности вершин, то две различные компоненты не пересекаются, т.е. не имеют ни общих вершин, ни общих ребер.

Так как отношение достижимости в ориентированном графе не является эквивалентностью, то компоненты ориентированного графа могут пересекаться.

Пример 5.3. Граф, изображенный на рис. 5.2, не является связным. Он состоит из трех компонент. Эти компоненты порождены тремя классами эквивалентности по отношению достижимости, указанными в примере 5.1.

Связными являются все графы, изображенные на рис. 5.7.

Ориентированный граф на рис. 5.8 связный, а ориентированные графы на рис. 5.3 и 5.9 не являются связными. В ориентированном графе на рис. 5.3 вершины v_2 и v_5 не достижимы одна из другой, а в ориенти-

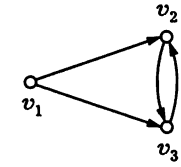


Рис. 5.8

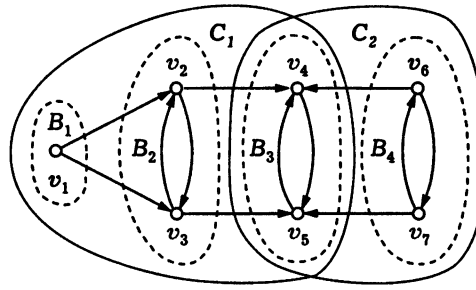


Рис. 5.9

рованном графе на рис. 5.9 взаимно не достижимы, например, вершины v_2 и v_6. В ориентированном графе, изображенном на рис. 5.9, имеются две компоненты связности: C_1 и C_2, которые пересекаются. #

Для ориентированного графа можно определить также понятия сильной и слабой связности.

Определение 5.2. Ориентированный граф называют ***сильно связным***, если для любых двух его вершин u и v вершина v достижима из вершины u и вершина u достижима из вершины v ($u \Rightarrow^* v$ и $v \Rightarrow^* u$). ***Бикомпонента*** ориентированного графа — это его максимальный сильно связный подграф.

Если $u \Rightarrow^* v$ и $v \Rightarrow^* u$, то говорят, что u и v связаны ***отношением взаимной достижимости***. Это бинарное отношение рефлексивно, симметрично и транзитивно, т.е. является отношением эквивалентности. Следовательно, две различные бикомпоненты не пересекаются, т.е. не имеют ни общих вершин, ни общих ребер.

Пример 5.4. а. В ориентированном графе, изображенном на рис. 5.3, бикомпонентой является подграф G_1, порожденный множеством вершин $\{v_1, v_2, v_3\}$. Действительно, эти вершины взаимно достижимы, поэтому ориентированный граф G_1 сильно связный. Так как из вершин v_4, v_5, v_6 ни одна из вершин v_1,

v_2, v_3 не достижима, то выделенный сильно связный подграф G_1 является максимальным.

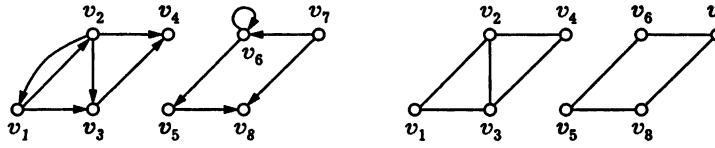
б. В ориентированном графе, представленном на рис. 5.9, имеются четыре бикомпоненты B_1, B_2, B_3 и B_4. Это подграфы, порожденные соответственно множествами вершин $\{v_1\}$, $\{v_2, v_3\}$, $\{v_4, v_5\}$ и $\{v_6, v_7\}$. Отметим, что подграф, порожденный множеством $\{v_1\}$, не содержит ни одной дуги. Тем не менее этот подграф — бикомпонента, поскольку каждая вершина достижима сама из себя (по пути длины 0).

Определение 5.3. Неориентированный граф $G_1 = (V_1, E_1)$ называют ***ассоциированным*** с ориентированном графом $G = (V, E)$, если его множество вершин совпадает с множеством вершин ориентированного графа G, а пара $\{u, v\}$ образует ребро тогда и только тогда, когда $u \neq v$ и из u в v или из v в u ведет дуга, т.е. $V_1 = V$ и

$$E_1 = \{\{u, v\}\colon (u, v) \in E \text{ или } (v, u) \in E,\ u \neq v\}.$$

Таким образом, переход от ориентированного графа к ассоциированному с ним неориентированному графу состоит в „стирании“ ориентации дуг ориентированного графа с учетом того, что все петли исчезают, а дуги (u, v) и (v, u) при $u \neq v$ переходят в одно и то же ребро $\{u, v\}$.

Для ориентированного графа, изображенного на рис. 5.10, *а*, ассоциированный с ним неориентированный граф приведен на рис. 5.10, *б*. Отметим, что дуги (v_1, v_2) и (v_2, v_1) переходят в ребро $\{v_1, v_2\}$, а петля (v_6, v_6) исчезает.



а *б*

Рис. 5.10

Определение 5.4. Ориентированный граф называют ***слабо связным***, если ассоциированный с ним неориентированный граф связный. ***Компонентой слабой связности*** (***слабой компонентой***) ориентированного графа называют его максимальный слабо связный подграф.

Ориентированные графы, представленные на рис. 5.3, 5.9 и 5.8, являются слабо связными. Ориентированный граф, изображенный на рис. 5.10, не является слабо связным, поскольку не является связным ассоциированный с ним неориентированный граф. Ориентированный граф на рис. 5.10, *а* имеет две компоненты слабой связности. Соответственно ассоциированный с ним неориентированный граф на рис. 5.10, *б* имеет две компоненты связности.

5.2. Способы представления

До сих пор мы задавали *ориентированные* и *неориентированные графы*, изображая их с помощью рисунков. Можно задать граф как пару множеств, следуя определению, однако этот способ довольно громоздкий и представляет, скорее, теоретический интерес. Развитие алгоритмических подходов к анализу свойств графов требует иных способов описания графов, более пригодных для практических вычислений, в том числе с использованием ЭВМ. Рассмотрим три наиболее распространенных способа представления графов.

Предположим, что все *вершины* и все *ребра неориентированного графа* или все вершины и все *дуги* (включая *петли*) *ориентированного графа* пронумерованы начиная с единицы. Граф (неориентированный или ориентированный) может быть представлен в виде матрицы типа $n \times m$, где n — число вершин, а m — число ребер (или дуг). Для неориентированного графа элементы этой матрицы задаются следующим образом:

$$a_{ij} = \begin{cases} 1, & \text{для } i\text{-й вершины } j\text{-е } \textit{ребро инцидентное}; \\ 0, & \text{иначе.} \end{cases}$$

Для *ориентированного графа* элементы матрицы задаются так:

$$a_{ij} = \begin{cases} 1, & \text{для } i\text{-й вершины } j\text{-я } \textit{дуга выходящая}; \\ -1, & \text{для } i\text{-й вершины } j\text{-я дуга } \textit{заходящая}; \\ 0, & \text{иначе.} \end{cases}$$

Матрицу (a_{ij}) типа $n \times m$, определенную указанным образом, называют ***матрицей инциденций***.

Пример 5.5. Для ориентированного графа, представленного на рис. 5.8, нумерация вершин уже задана. Зададим нумерацию дуг следующим образом: дуге (v_1, v_2) присвоим номер 1, дуге (v_1, v_3) — 2, дуге (v_2, v_3) — 3 и дуге (v_3, v_2) — 4.

Матрицу инциденций удобно заполнять по столбцам, записывая для k-й дуги (v_i, v_j) 1 в i-й и -1 в j-й строках k-го столбца и 0 во всех остальный строках k-го столбца. В результате получим матрицу

$$\begin{pmatrix} 1 & 1 & 0 & 0 \\ -1 & 0 & 1 & -1 \\ 0 & -1 & -1 & 1 \end{pmatrix}. \quad \#$$

Несмотря на то что представление графа в виде матрицы инциденций играет весьма большую роль в теоретических исследованиях, практически этот способ весьма неэффективен. Прежде всего, в матрице в каждом столбце только два ненулевых элемента, что делает этот способ представления графа неэкономным при большом количестве вершин. Кроме того, решение практических задач с помощью матрицы инциденций весьма трудоемко.

Оценим, например, временные затраты на решение с помощью матрицы инциденций такой простой задачи в ориентированном графе: для данной вершины v_k найти ее „окружение“ — множество *преемников* и множество *предшественников* вершины v_k, т.е. множество всех вершин, непосредственно достижимых из v_k, и множество всех вершин, из которых она непосредственно достижима.

Для решения этой задачи на матрице инциденций ориентированного графа нужно идти по строке с номером k до появления ненулевого элемента ($+1$ или -1). В случае если обнаружена $+1$, в соответствующем столбце надо найти строку, в которой записано число -1. Номер строки, в которой стоит это число, дает номер вершины, непосредственно достижимой из данной вершины. Если обнаружена -1, в столбце надо найти строку, в которой записана 1, и получить номер вершины, из которой непосредственно достижима данная вершина. Для получения всего „окружения" надо проделать указанный поиск для всех ненулевых элементов k-й строки. Наиболее трудоемкой процедурой является поиск ненулевого элемента в столбце. Число таких процедур поиска равно степени вершины v_k. Будем в этом случае говорить, что ***сложность алгоритма*** анализа окружения вершины v_k составляет $O(\deg v_k)$ (порядка $\deg v_k$).

Можно увидеть, что поиск „окружения" всех вершин займет время порядка произведения числа вершин ориентированного графа на сумму степеней всех вершин, которая, как можно показать, пропорциональна числу дуг ориентированного графа. Таким образом, сложность алгоритма поиска „окружения" составляет $O(nm)$, т.е. поиск занимает время порядка произведения числа вершин на число дуг.

Более эффективной матричной структурой, представляющей граф, служит ***матрица смежности вершин***, или ***булева матрица*** графа. Это квадратная матрица B порядка n, элементы которой определяют следующим образом:

для неориентированного графа

$$b_{ij} = \begin{cases} 1, & i\text{-я и } j\text{-я вершины } \textit{смежные}; \\ 0, & \text{иначе}; \end{cases}$$

для ориентированного графа

$$b_{ij} = \begin{cases} 1, & \text{из } i\text{-й вершины в } j\text{-ю ведет дуга}; \\ 0, & \text{иначе}. \end{cases}$$

Заметим, что в k-й строке матрицы ориентированного графа количество единиц равно *полустепени исхода* $\mathrm{dg}^+ v_k$ вершины v_k, а количество единиц в k-м столбце — *полустепени захода* $\mathrm{dg}^- v_k$.

Для неориентированного графа *матрица* смежности вершин *симметрическая*.

Для ориентированного графа, изображенного на рис. 5.8, матрица смежности вершин имеет вид

$$\begin{pmatrix} 0 & 1 & 1 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{pmatrix}.$$

Рассмотрим решение задачи поиска „окружения“ с использованием матрицы смежности вершин. Для определения „окружения“ вершины v_k нужно сначала идти по k-й строке матрицы и искать ненулевые элементы. Если элемент $a_{ki} = 1$, то вершина v_i достижима из вершины v_k. После просмотра k-й строки надо просмотреть k-й столбец. Если элемент $a_{jk} = 1$, то вершина v_k достижима из вершины v_j.

Можно показать, что с использованием матрицы смежности вершин решение задачи поиска „окружения“ всех вершин ориентированного графа будет иметь сложность порядка n^2, что эффективнее предыдущей оценки nm, если число дуг ориентированного графа превышает число его вершин, а это часто бывает в практических задачах.

Матрица смежности вершин является достаточно эффективным способом представления графов. Однако эту матрицу удобно строить по графу, уже заданному каким-либо способом, например рисунком. Во многих задачах граф создается динамически, т.е. в ходе решения задачи меняется множество вершин и множество ребер (или дуг). В этом случае эффективным способом машинного представления графа являются ***списки смежности*** (или ***списки инцидентности***).

Рассмотрим ориентированный граф. Для задания множества вершин, непосредственно достижимых из вершины v, ис-

пользуют линейный однонаправленный список*. Каждый элемент такого списка включает данные (например, некоторое число) и указатель на следующий элемент списка. Список в целом задается указателем на его первый элемент (***голову списка***). Последний элемент списка содержит „пустой“ указатель (рис. 5.11).

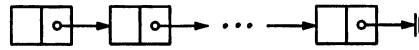


Рис. 5.11

Задать для вершины v ее список смежности означает в произвольном порядке поместить в данные элементов списка номера вершин u, для которых в ориентированном графе есть дуга из v в u $(v \to u)$. ***Список смежности вершины*** v обозначают $L(v)$.

Отметим, что список смежности вершины может при необходимости дополняться. Для этого в последнем элементе списка „пустой“ указатель заменяется указателем на добавляемый элемент, который становится последним элементом списка с „пустым“ указателем.

Если количество вершин ориентированного графа известно заранее, то ориентированный граф удобно задавать в виде структуры, называемой ***массивом лидеров***.

Под ***массивом*** мы понимаем матрицу-столбец, элементами которой могут быть некоторые объекты (например, элементы списка смежности). Их называют ***элементами массива***. Число элементов массива лидеров равно числу вершин графа.

Элементами массива лидеров являются первые элементы списков смежности вершин ориентированного графа. Пример представления ориентированного графа списками смежности, собранными в массив лидеров, представлен на рис. 5.12.

*Подробное описание обработки списков и всей „программистской“ терминологии, использованной в этом абзаце, см.: *Вирт Н.*

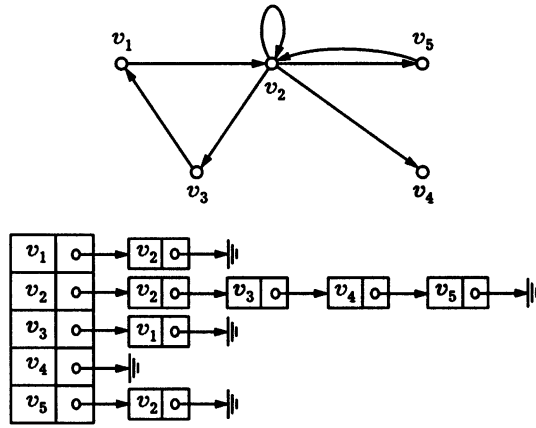


Рис. 5.12

С использованием списков смежности совсем просто решается задача поиска *преемников* данной вершины: для этого достаточно просмотреть список смежности вершины, затратив на это время, пропорциональное ее *полустепени исхода*. Тогда на решение этой задачи для всего ориентированного графа потребуется время порядка числа его дуг.

Менее эффективно решается задача поиска *предшественников* вершины, так как в этом случае необходимо, вообще говоря, просмотреть списки смежности всех вершин с целью поиска в них данной вершины.

Таким образом, задача поиска „окружения“ с использованием списков смежности является более трудоемкой, чем в случае использования матриц. Однако удобство динамического формирования описания ориентированного графа в данном случае перевешивает. Если задача поиска предшественников возникает часто, можно использовать двусторонние списки смежности, сопоставляя каждой вершине уже два списка — преемников и предшественников.

Описанные способы представления графа списками не исчерпывают всех возможных вариантов, и в литературе по программированию можно найти разнообразные варианты ор-

ганизации списков. Поскольку эти особенности относятся к технологии программирования, мы не будем в них углубляться и в дальнейшем для простоты при описании различных алгоритмов будем считать, что (односторонние) списки смежности собраны в массив лидеров, так как в рассматриваемых ниже алгоритмах анализа графов именно анализ множества преемников вершины наиболее важен.

Неориентированный граф задать с помощью списков смежности можно так же, как и ориентированный. Здесь в список смежности вершины v войдут все вершины, смежные с ней, а списки смежности могут быть собраны в массив лидеров. Для неориентированого графа задача поиска „окружения“ одной вершины требует однократного просмотра ее списка смежности, и затраты времени на это пропорциональны *степени вершины*. На решение задачи поиска „окружения“ для всего неориентированного графа потребуется время, пропорциональное произведению числа вершин графа на число его ребер.

В заключение рассмотрим еще одну матрицу, характеризующую граф, — так называемую ***матрицу достижимости***. Это квадратная матрица C порядка $|V|$, каждый элемент c_{ij} которой равен 1, если *j-я вершина достижима* из *i-й* вершины, и равен 0 если иначе. Отметим, что, согласно определению достижимости, элементы $c_{ii} = 1$.

Метод вычисления матрицы достижимости ориентированного графа по его матрице смежности будет рассмотрен в **5.6**.

Матрица достижимости несет очень важную информацию об ориентированном графе. Ее анализ позволяет, например, найти его *бикомпоненты* и *компоненты*.

По определению, в бикомпоненту входят взаимно достижимые вершины. Для двух таких вершин с номерами i и j должно выполняться равенство $c_{ij} = c_{ji} = 1$. Поэтому, чтобы найти бикомпоненту, в которую входит i-я вершина ориентированного графа, нужно просмотреть i-ю строку и i-й столбец матрицы достижимости C и сформировать множество $P_i = \{p\colon c_{ip} = c_{pi} = 1\}$ номеров вершин, порождающих искомую

бикомпоненту. Из определения матрицы достижимости вытекает, что в P_i содержатся номера всех вершин данной бикомпоненты. Поскольку две различные бикомпоненты не пересекаются, вершины с номерами из множества P_i при поиске других бикомпонент из рассмотрения можно исключить. Процесс поиска целесообразно начать с первой вершины. Он закончится, когда для каждой вершины будет найдена содержащая ее бикомпонента. При этом может оказаться, что некоторые (а может быть, и все) бикомпоненты содержат только по одной вершине, поскольку каждая вершина, по определению, достижима сама из себя.

Поиск компонент ориентированного графа сложнее, так как в компоненту входят вершины с номерами i и j, для которых $c_{ij} = 1$ или $c_{ji} = 1$. Кроме того, одна и та же вершина может входить в несколько различных компонент. Отметим, что любая бикомпонента или не пересекается с некоторой компонентой, или целиком в нее входит.

Мы не будем приводить общий алгоритм поиска компонент, а рассмотрим его на примере.

Пример 5.6. Для ориентированного графа, изображенного на рис. 5.9, имеем матрицу достижимости

$$C = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{pmatrix}.$$

Начнем с поиска бикомпонент. Для первой вершины множество $P_1 = \{1\}$ включает только ее саму. Для второй вершины имеем $P_2 = \{2, 3\}$, для четвертой — $P_4 = \{4, 5\}$ и для шестой — $P_6 = \{6, 7\}$. Соответственно полученные множества вершин порождают бикомпоненты B_1, B_2, B_3 и B_4, изображенные на рис. 5.9.

Перейдем к поиску компонент. Используя матрицу C, выпишем для каждой вершины v_i, $i = \overline{1,7}$, множество K_i, состоящее из тех вершин, которые достижимы из i-й вершины или из которых достижима она.

В рассматриваемом ориентированном графе для вершины v_1 в множество K_1 входят вершины, для которых $c_{1j} = 1$ или $c_{j1} = 1$, т.е. $K_1 = \{v_1, v_2, v_3, v_4, v_5\}$. Для вершин v_2 и v_3 множества K_2 и K_3 совпадают с множеством K_1. Поскольку все элементы четвертого и пятого столбцов матрицы C равны единице, то для вершин v_4 и v_5 имеем $K_4 = K_5 = \{v_1, v_2, v_3, v_4, v_5, v_6, v_7\}$.

Для вершин v_6 и v_7 получим $K_6 = K_7 = \{v_4, v_5, v_6, v_7\}$.

Отметим, что множество K_i построено так, что любая компонента, содержащая вершину v_i, может содержать только вершины из множества K_i.

Кроме того, если некоторая компонента C_p содержит вершины v_i и v_j, то вершина v_m будет принадлежать этой компоненте в том и только в том случае, если $v_m \in K_i \cap K_j$.

Действительно, пусть вершина v_m принадлежит компоненте C_p вместе с вершинами v_i и v_j. Тогда либо вершина v_m достижима из вершины v_i, либо, наоборот, v_i достижима из v_m, и, следовательно, $v_m \in K_i$. Из аналогичных рассуждений вытекает, что $v_m \in K_j$. Таким образом, вершина v_m принадлежит пересечению множеств K_i и K_j.

Пусть теперь вершины v_i и v_j принадлежат компоненте C_p и $v_m \in K_i \cap K_j$. Тогда вершина v_m также принадлежит компоненте C_p, поскольку либо вершина v_i достижима из v_m, либо v_m достижима из v_i, и для вершин v_j и v_m ситуация аналогична.

Начнем строить компоненты, содержащие вершину v_1. Рассмотрим множество K_1. Так как вершина v_2 принадлежит множеству K_1, найдется по крайней мере одна компонента, в которую входят обе эти вершины. Обозначим ее через C_1. Вершина v_3 будет принадлежать этой компоненте в том и только в том случае, если $v_3 \in K_1 \cap K_2$. Непосредственная проверка показывает, что последнее условие выполняется.

Можно заметить, что

$$K_1 \cap K_2 \cap K_3 \cap K_4 \cap K_5 = K_1,$$

поэтому компонента C_1 (см. рис. 5.9) есть подграф, порожденный множеством K_1, и вершина v_1 не может входить в какую-либо другую компоненту, отличную от C_1. Поскольку в компоненту C_1 вошли все вершины из множеств K_2 и K_3, то вершины v_2 и v_3 также не входят ни в какие другие компоненты.

Перейдем к поиску компонент, отличных от C_1, в которые входит вершина v_4. В множестве K_4 содержится вершина v_5. Пересечению множеств K_4 и K_5 принадлежат вершины v_6 и v_7, не вошедшие в выделенную компоненту C_1. Рассуждая аналогично и учитывая, что v_4 и v_5 принадлежат бикомпоненте B_3, а v_6 и v_7 — бикомпоненте B_4, получаем, что компонента C_2 порождается множеством вершин

$$K_4 \cap K_5 \cap K_6 \cap K_7 = K_7,$$

и других компонент в рассматриваемом ориентированном графе нет.

5.3. Деревья

Определение 5.5. ***Неориентированным деревом*** называют *связный* и *ациклический неориентированный граф.*

Определение 5.6. ***Ориентированным деревом*** называют *бесконтурный ориентированный граф*, у которого *полустепень захода* любой вершины не больше 1 и существует ровно одна вершина, называемая ***корнем ориентированного дерева***, *полустепень захода* которой равна 0.

Опираясь на данное определение, можно доказать, что в ориентированном дереве любая вершина *достижима* из корня.

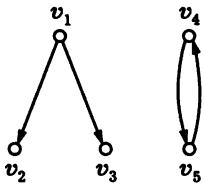


Рис. 5.13

Отметим, что из определения 5.6 нельзя убрать требование бесконтурности ориентированного графа, поскольку бесконтурность не вытекает из других условий. Например, на рис. 5.13 изображен ориентированный граф, не являющийся ориентированным деревом, хотя полустепени захода всех вершин не больше 1 и ровно одна вершина имеет полустепень захода, равную 0.

Определение 5.7. Вершину v ориентированного дерева называют ***потомком*** (***подлинным потомком***) вершины u, если существует путь из u в v (путь ненулевой длины из u в v). В этом же случае вершину u называют ***предком*** (***подлинным предком***) вершины v, а если длина пути из u в v равна 1, то вершину v называют ***сыном*** вершины u, которая при этом вполне естественно именуется ***отцом*** вершины v. Вершину, не имеющую потомков, называют ***листом***.

На рис. 5.14 вершины v_4 и v_5 есть сыновья вершины v_2, которая, в свою очередь, является сыном вершины v_0 — корня дерева. Вершины v_4 и v_5 являются подлинными потомками вершин

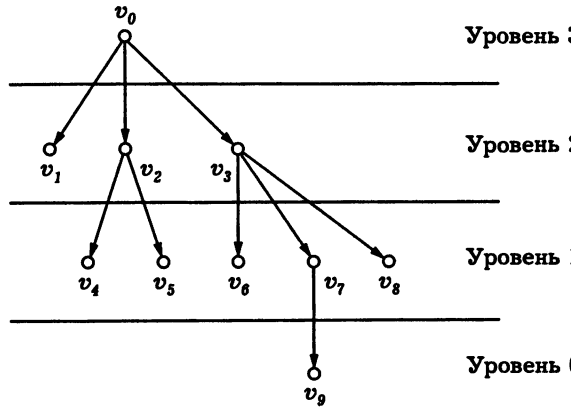


Рис. 5.14

v_0 и v_2, которые соответственно будут их подлинными предками. Вершины v_1, v_4, v_5, v_6, v_9, v_8 — листья дерева. Взаимно недостижимые вершины ориентированного дерева (например, такие, как v_2 и v_9) не являются ни предком, ни потомком одна другой. Каждая вершина будет сама для себя предком и потомком, но не подлинным.

Определение 5.8. Произвольный ациклический граф называют ***неориентированным лесом.*** Если каждая *слабая компонента* ориентированного графа является ориентированным деревом, то такой граф называют ***ориентированным лесом.***

На рис. 5.15, *а* даны примеры неориентированного леса, а на рис. 5.15, *б* — примеры орентированного леса.

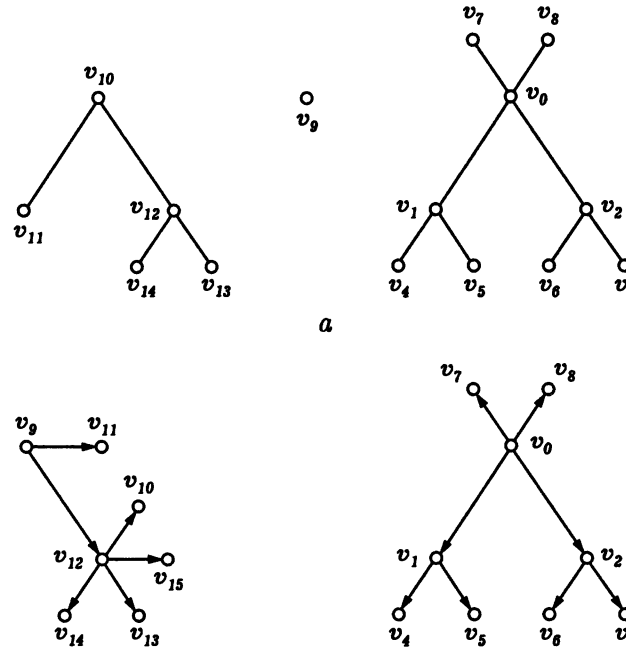


Рис. 5.15

Определение 5.9. Подграф неориентированного (ориентированного) дерева, являющийся неориентированным (ориентированным) деревом, называют ***поддеревом*** исходного дерева.

Из определения следует, что неориентированный лес — это неориентированный граф, каждая компонента которого является неориентированным деревом.

На рис. 5.14 *подграф, порожденный множеством вершин* $\{v_3, v_6, v_7, v_8, v_9\}$, является поддеревом изображенного на рисунке ориентированного дерева.

Определение 5.10. Ориентированное дерево, у которого каждая вершина, отличная от корня, есть лист, называют ***кустом.***

На рис. 5.13 граф, изображенный слева, является кустом.

Рассмотрим теперь некоторые числовые параметры, которыми характеризуется ориентированное дерево.

Определение 5.11. ***Высота ориентированного дерева*** — это наибольшая длина пути из корня в лист; ***глубина*** $d(v)$ ***вершины ориентированного дерева*** v — это длина пути из корня в эту вершину; ***высота*** $h(v)$ ***вершины ориентированного дерева*** v — это наибольшая длина пути из данной вершины в лист и, наконец, ***уровень вершины ориентированного дерева*** — это разность между высотой ориентированного дерева и глубиной данной вершины.

Уровень корня равен высоте ориентированного дерева, но уровни различных листьев, так же как и их глубины, могут быть различными; высота любого листа равна нулю (см. рис. 5.14).

Определение 5.12. Ориентированное дерево называют ***бинарным***, если полустепень исхода любой его вершины не больше 2; бинарное ориентированное дерево называют ***полным***, если из любой его вершины, не являющейся листом, исходят ровно две дуги, а уровни всех листьев совпадают.

Примеры различных бинарных ориентированных деревьев приведены на рис. 5.16: ориентированные деревья на рис. 5.16, *а* и *б* неполные, а ориентированное дерево на рис. 5.16, *в* полное.

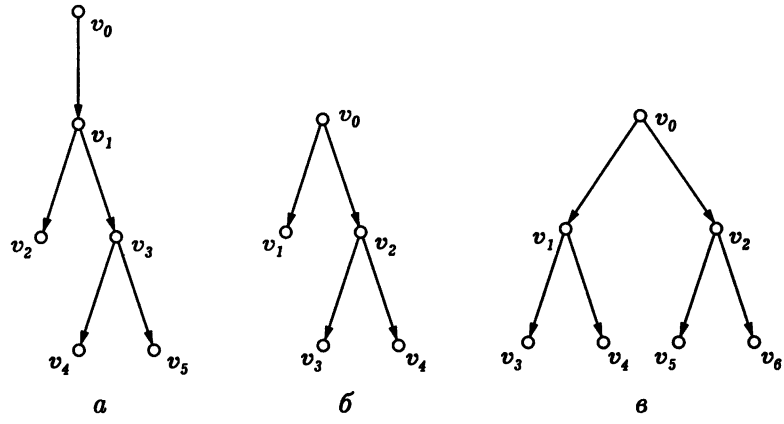


Рис. 5.16

Используя индукцию по высоте ориентированного дерева, можно показать, что в полном бинарном ориентированном дереве высоты h ровно 2^h листьев. Действительно, ориентированное дерево высоты 0 имеет $2^0 = 1$ лист. Полное бинарное ориентированное дерево высоты 1 имеет $2^1 = 2$ листа.

Пусть полное бинарное ориентированное дерево имеет высоту k и соответственно 2^k листьев. Рассмотрим полное бинарное ориентированное дерево высоты $k+1$. Поскольку в полном бинарном ориентированном дереве уровни всех листьев совпадают, легко видеть, что ориентированное дерево высоты $k+1$ можно получить из полного бинарного ориентированного дерева высоты k, если из каждого листа последнего провести по две дуги. Тогда количество листьев в ориентированном дереве высоты $k+1$ будет в 2 раза больше, чем в ориентированном дереве высоты k, т.е. $2^k \cdot 2 = 2^{k+1}$.

Отсюда следует, что в произвольном бинарном дереве высоты h не более 2^h листьев. Таким образом, мы получаем следующий простой, но важный результат.

Теорема 5.2 (теорема о высоте бинарного ориентированного дерева с заданным числом листьев). Бинарное ориентированное дерево с n листьями имеет высоту, не меньшую $\log_2 n$. #

Эта теорема имеет одно весьма интересное приложение. Предположим, что необходимо расположить строго по возрастанию элементы конечного *линейно упорядоченного множества* $\{a_1, \ldots, a_n\}$. Эту задачу называют ***задачей сортировки***, а любой алгоритм, ее решающий, — ***алгоритмом сортировки***. С математической точки зрения алгоритм сортировки должен найти такую *перестановку* $\{a_{i_1}, \ldots, a_{i_n}\}$ элементов множества, которая была бы согласована с заданным на нем отношением $\leqslant$ линейного порядка, т.е. для любых k, l из справедливости неравенства $i_k < i_l$ должно следовать $a_{i_k} \leqslant a_{i_l}$.

Первоначально сортируемые элементы могут быть расположены в произвольном порядке, т.е. исходной может быть любая перестановка элементов сортируемого множества, и мы не имеем никакой априорной информации об этой перестановке. Единственный способ получить такую информацию — проводить попарные сравнения элементов a_i (относительно рассматриваемого линейного порядка) в какой-либо последовательности. Заметим, что при этом совершенно не обязательно проводить все возможные сравнения, т.е. сравнивать a_i с a_j для всех $i \neq j$. Например, можно использовать транзитивность отношения порядка.

Все сравнения, которые в принципе могут быть проведены в процессе работы некоторого алгоритма, изображаются наглядно в виде ориентированного дерева, называемого ***деревом решений***. На рис. 5.17 приведено дерево решений для трехэлементного множества $\{a, b, c\}$. Вершины ориентированного дерева, не являющиеся листьями, помечены проверяемыми неравенствами, а множество листьев находится во *взаимно однозначном соответствии* с множеством всех перестановок исходного множества.

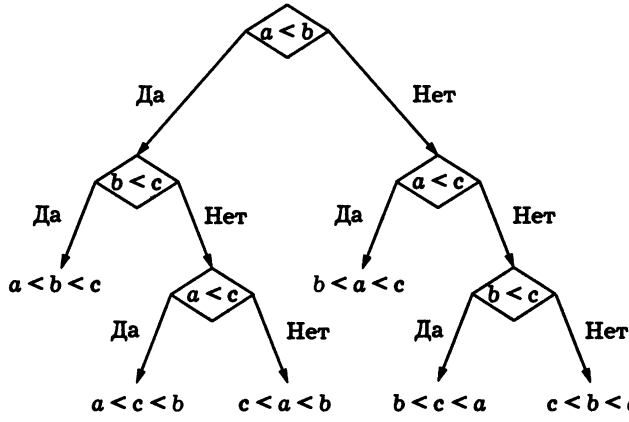


Рис. 5.17

Поскольку в результате сортировки может получиться любая перестановка исходного множества и каждой такой перестановке соответствует лист дерева решений, в общем случае количество листьев будет равно $n!$ — количеству перестановок n-элементного множества.

Следовательно, сортируя входную последовательность, алгоритм обязательно пройдет какой-то путь от корня дерева решений к одному из листьев, и, таким образом, число операций сравнения (число шагов алгоритма сортировки) будет величиной, пропорциональной высоте дерева решений, не меньшей чем $\log_2 n!$, в силу теоремы 5.2.

Используя для оценки факториала при больших n *формулу Стирлинга*

$$n! \approx \sqrt{2\pi n}\left(\frac{n}{e}\right)^n,$$

получаем, что дерево решений имеет высоту порядка $n \log_2 n$.

Таким образом, в общем случае задачу сортировки с помощью попарных сравнений нельзя решить быстрее, чем за указанное число шагов. Безусловно, конкретный путь в дереве решений из корня к одному из листов зависит от исходной перестановки. Например, в дереве решений, приведенном на

рис. 5.17, есть два „коротких“ пути длины 2, однако остальные пути имеют длину 3.

Заметим, что в общем случае ориентированное дерево — это не связный ориентированный граф. Компонентами ориентированного дерева являются его подграфы, порожденные множеством вершин, расположенных на некотором пути из корня в лист.

Остовным лесом (***деревом***) неориентированного (ориентированного) графа называют любой его остовный подграф, являющийся лесом (деревом).

Теорема 5.3. У всякого неориентированного графа существует максимальный остовный лес.

◀ Поскольку неориентированный лес — это произвольный ациклический граф, то достаточно показать, что всякий неориентированный граф содержит максимальный остовный ациклический подграф.

Рассмотрим произвольный неориентированный граф $G_0 = (V, E)$. Если он ациклический, то утверждение очевидно. В противном случае в нем есть хотя бы один цикл. Выберем один из циклов графа и обозначим его C_1. Удалим из цикла C_1 какое-нибудь ребро. Обозначим его e_1. Так как цикл — это простая цепь, то удаление ребра e_1 приводит к подграфу $G_1 = (V, E \setminus \{e_1\})$ графа G_0, в котором будет по крайней мере одним циклом меньше, чем в графе G_0. В то же время любые две вершины, соединенные цепью в исходном графе, будут соединены цепью и в подграфе G_1. Если граф G_1 ациклический, то он и будет искомым максимальным остовным лесом исходного графа, так как возвращение удаленного ребра e_1 приведет к появлению цикла. Если же граф G_1 имеет циклы, то поступим с ним точно так же, как с графом G_0, а именно, выбрав какой-нибудь его цикл C_2, удалим из него одно ребро — обозначим его e_2. В силу конечности множеств ребер исходного графа через конечное число $n \geqslant 1$ шагов, на каждом из которых удаляется ровно одно ребро некоторого

цикла графа G, получим некоторый его ациклический подграф $G_n = (V, T)$, где $T = E \setminus \{e_1, \ldots, e_n\}$, причем подграф $G_{n-1} = (V, E \setminus \{e_1, \ldots, e_{n-1}\})$ содержит цикл, проходящий по ребру e_n. Покажем, что добавление к множеству ребер T хотя бы одного ребра из ранее удаленных, т.е. любого ребра из множества $\{e_1, \ldots, e_n\}$, приведет к появлению цикла.

Действительно, на каждом шаге описанной выше процедуры происходит удаление нового цикла графа G_0, т.е. все циклы C_1, $C_2, \ldots, C_n$ попарно различны. Для каждого $i = \overline{1, n}$ рассмотрим два случая: 1) цикл C_i содержит только одно из удаленных ребер, а именно ребро e_i; 2) цикл C_i содержит не менее двух удаленных ребер, т.е. вместе с ребром e_i он содержит по крайней мере еще одно ребро $e_j \in \{e_1, \ldots, e_n\}$.

Очевидно, что в первом случае добавление ребра e_i к множеству ребер T приведет к появлению цикла, а именно цикла C_i. Рассмотрим второй случай. Для простоты будем считать, что есть только одно ребро e_j, отличное от ребра e_i, которое содержится в цикле C_i (случай нескольких таких ребер анализируется аналогично). Так как ребро e_j содержится в некотором цикле C_j, отличном от C_i, то, добавляя ребро e_i к множеству ребер T, все равно получим цикл (рис. 5.18). Действительно, если ребро e_i не содержится в цикле C_j, то, как видно на рис. 5.18, концы ребра e_i соединены цепью, не содержащей этого ребра. Тогда, согласно теореме 5.1, существует простая цепь (не содержащая ребра e_i), соединяющая его концы. Добавляя к ней

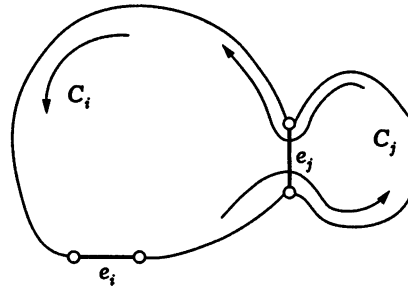


Рис. 5.18

ребро e_i, получим цикл. Если же ребро e_i содержится в цикле C_j, то удаление этого ребра приведет к исчезновению вместе с циклом C_i и цикла C_j, что невозможно, так как C_i и C_j — циклы, удаляемые на разных шагах описанной выше процедуры (на самом деле можно показать, что в рассматриваемой ситуации ребро e_j, а вместе с ним и цикл C_j, удаляется позже ребра e_i). Таким образом, всегда найдется цепь, соединяющая концы ребра e_i (и не содержащая этого ребра). Следовательно, существует и цикл, содержащий ребро e_i (но не содержащий ребра e_j).

Итак, добавляя к подграфу $G_n = (V, T)$ произвольное ребро из множества $\{e_1, \ldots, e_n\}$, получим цикл. Следовательно, этот подграф и будет искомым максимальным остовным лесом графа G. ▶

Утверждение теоремы сохраняет силу и для ориентированных графов. Доказательство в этом случае будет несколько более сложным, и мы его не приводим.

Далее в этой главе мы опишем некоторые способы нахождения множества циклов неориентированного графа и способы построения максимальных остовных лесов. Эти задачи решаются на основе приведенных ниже алгритмов поиска в глубину.

5.4. Остовное дерево наименьшего веса

Следующая задача известна в теории графов под названием ***задачи Штейнера***: на плоскости заданы n точек; нужно соединить их отрезками прямых таким образом, чтобы суммарная длина отрезков была наименьшей.

В процессе поиска решения разрешено вводить новые точки; „длина“ отрезка, соединяющего две заданные точки, не обязательно понимается буквально геометрически — это может быть некоторая количественная оценка „цены“ прохождения из точки в точку по какой-то ломаной линии.

Можно дать следующую графовую интерпретацию задачи Штейнера. Дан *неориентированный граф* $G_0 = (V_0, \varnothing)$ без *ребер*. Требуется найти неориентированый граф $G = (V, T)$ со специальными свойствами.

Мы предполагаем, что каждому ребру любого неориентированного графа, получаемого в процессе решения задачи, можно сопоставить неотрицательное число, называемое весом этого ребра. При этом, во-первых, искомый граф G должен быть *неориентированным деревом*, во-вторых, его множество *вершин* должно включать множество вершин исходного графа, т.е. $V_0 \subseteq V$, и, в-третьих, сумма весов его ребер должна быть наименьшей.

Требование, чтобы искомый граф G был неориентированным деревом, вызвано тем, что в нем любую пару вершин должна соединять единственная *цепь*, так как в противном случае граф не будет иметь наименьшую сумму весов. Действительно, если существуют хотя бы две цепи, соединяющие какую-то пару вершин, то выбирается та, сумма весов ребер которой меньше. Если же предположить, что в рассматриваемой ситуации имеются хотя бы две цепи с одинаковым весом, то все равно для оптимального решения выбирается какая-то одна цепь (и тогда оптимальное решение не единственно).

Эффективных алгоритмов*, дающих точное решение задачи Штейнера, не существует. Однако известны эффективные алгоритмы, находящие некоторые приближения точного решения. Одно из таких решений дает ***алгоритм Краскала***.

Неориентированный (ориентированный) граф, у которого каждому ребру (*дуге*) сопоставлено некоторое действительное число, называют ***взвешенным*** или ***размеченным*** графом. Это число называют ***весом*** или ***меткой*** ребра (дуги). Как

*Алгоритм считают ***эффективным***, если ***сложность алгоритма*** выражается функцией, ограниченной сверху некоторым полиномом от параметра, характеризующего „объем“ исходных данных. Для задачи Штейнера этим параметром является число вершин графа G_0.

правило, мы рассматриваем граф с натуральными метками ребер (дуг).

Алгоритм Краскала вычисляет для заданного взвешенного неориентированного графа G *остовное дерево* с наименьшей суммой весов ребер — ***остовное дерево наименьшего веса.*** Существенное отличие только что сформулированной задачи от задачи Штейнера (в ее графовой постановке) состоит в том, что в новой задаче множество вершин при поиске остовного дерева наименьшего веса не меняется. Поэтому алгоритм Краскала и дает лишь некоторое приближение решения задачи Штейнера.

При описании алгоритма будем использовать способ хранения данных, называемый ***очередью.*** Элементы данных в очереди упорядочиваются по времени поступления. Элементы можно добавлять в очередь и извлекать из очереди. В каждый момент времени доступен только один элемент, который был помещен в очередь раньше других, — „голова“ очереди. При добавлении новый элемент помещается в „хвост“ очереди, т.е. работа ведется по обычному для очереди правилу — „первым пришел — первым вышел“. Чтобы извлечь из очереди некоторый элемент, не доступный в текущий момент, надо извлечь все ранее поступившие элементы, начиная с „головы“ очереди.

Рассмотрим алгоритм нахождения остовного дерева наименьшего веса (алгоритм Краскала). Пусть дан связный неориентированный граф $G = (V, E)$ с числовыми неотрицательными весами ребер. Вес ребра e обозначим $\varphi(e)$.

В результате работы алгоритма получим остовное дерево $T = (V, H)$ графа G, такое, что сумма $\sum\limits_{e \in H} \varphi(e)$ является наименьшей.

Отсортируем все ребра исходного графа по возрастанию весов и сформируем из них очередь так, чтобы в „голове“ очереди находилось ребро с наименьшим весом, а в „хвосте“ — с наибольшим и веса ребер не убывали от „головы“ очереди к „хвосту“.

Метод состоит в „сшивании“ искомого дерева из *компонент* остовного леса. Первоначально *остовный лес* представляет собой множество изолированных вершин исходного графа, т.е. его множество ребер пусто. На первом шаге из очереди извлекается ребро наименьшего веса и добавляется к множеству ребер исходного дерева.

На последующих шагах алгоритма из очереди извлекается по одному ребру. Если это ребро соединяет вершины, принадлежащие разным компонентам текущего остовного леса, то оно добавляется к текущему множеству ребер искомого дерева, а указанные компоненты сливаются в одну. Иначе ребро отбрасывается. Процесс повторяется до тех пор, пока число компонент остовного леса не окажется равным 1. Можно показать, что эта компонента и будет искомым остовным деревом наименьшего веса.

Переходим к формальному описанию алгоритма.

Алгоритм Краскала

1. Множество ребер H искомого остовного дерева полагаем пустым ($H = \varnothing$).

2. Формируем множество $V_S = \{\{v_1\}, \ldots, \{v_n\}\}$, элементами которого являются множества вершин, соответствующих компонентам исходного остовного леса. Каждая такая компонента состоит из единственной вершины.

3. Сортируем множество ребер E исходного графа по возрастанию весов и формируем очередь Q, элементами которой являются ребра графа G.

4. Если множество V_S содержит более одного элемента (т.е. остовный лес состоит из нескольких компонент) и очередь Q не пуста, переходим на шаг **5**, если иначе — на шаг **7**.

5. Извлекаем из очереди Q ребро e. Если *концы* ребра e принадлежат различным множествам вершин V_i и V_j из V_S, то переходим на шаг **6**, если иначе, то отбрасываем извлеченное ребро и возвращаемся на шаг **4**.

6. Объединяем множества вершин V_i и V_j (полагая $W = V_i \cup V_j$), удаляем множества V_i и V_j из множества V_S и добавляем в V_S множество W. Добавляем ребро e в множество H. Возвращаемся на шаг **4**.

7. Прекращаем работу. Множество H есть множество ребер полученного остовного дерева.

Доказательство корректности алгоритма Краскала, т.е. доказательство того факта, что выдаваемое алгоритмом остовное дерево действительно является остовным деревом наименьшего веса, мы не приводим.

На рис. 5.19, *а*–*д* для неориентированного графа показана последовательность построения остовного дерева наименьшего веса. Заметим, что результат работы алгоритма в общем случае зависит от порядка следования ребер одинакового веса в очереди. Предположим, что после сортировки первым в очереди находится ребро $\{v_0, v_1\}$ с весом 2.

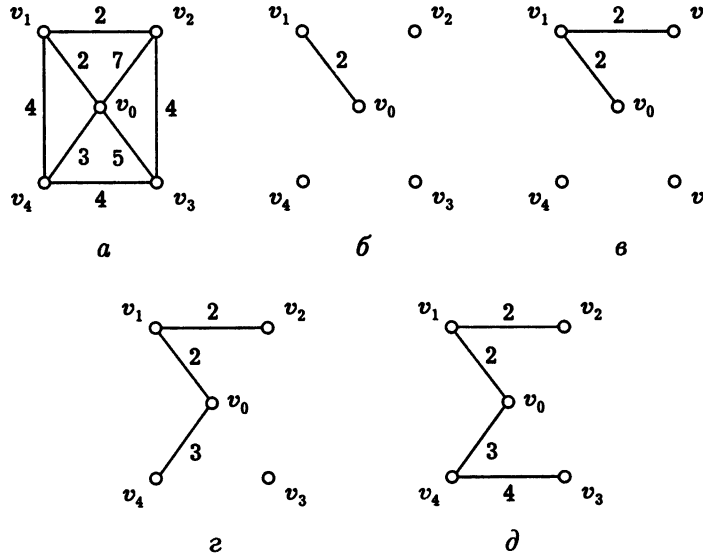


Рис. 5.19

Исходный граф изображен на рис. 5.19, *а*. На рис. 5.19, *б* проиллюстрирован результат выполнения первого шага алгоритма. На рис. 5.19, *в* показан результат добавления следующего ребра $\{v_1, v_2\}$ с весом 2 из очереди. На рис. 5.19, *г* приведен результат добавления ребра $\{v_0, v_4\}$ с весом 3. Если следующим в очереди ребром будет $\{v_1, v_4\}$, оно будет отброшено. Дальнейший ход работы алгоритма зависит от того, в каком порядке в очереди размещены ребра $\{v_2, v_3\}$ и $\{v_3, v_4\}$ с весами 4. Любое из них может быть добавлено в множество ребер остовного дерева, и на этом алгоритм закончит работу. На рис. 5.19, *д* приведено остовное дерево, полученное после добавления ребра $\{v_3, v_4\}$.

Отметим, что для приведенного графа оба ребра с весом 2 войдут в остовное дерево независимо от порядка их расположения в очереди после сортировки, а ребро $\{v_1, v_4\}$ не войдет ни в какое остовное дерево наименьшего веса.

Можно доказать, что наиболее трудоемким шагом в алгоритме Краскала является сортировка ребер графа по возрастанию весов. Как мы уже знаем, *задачу сортировки* n элементов нельзя решить быстрее, чем за время $O(n \log_2 n)$. Следовательно, *сложность алгоритма* Краскала оценивается числом $O(|E| \log_2 |E|)$, где $|E|$ — мощность множества ребер графа. Поскольку справедливо неравенство $|E| \log_2 |E| \leqslant |E|^2$, то алгоритм Краскала можно считать эффективным.

5.5. Методы систематического обхода вершин графа

Важными задачами теории графов являются ***задачи глобального анализа*** как *неориентированных*, так и *ориентированных графов*. К этим задачам относятся, например, задачи поиска *циклов* или *контуров*, вычисление *длин путей* между парами *вершин*, перечисление *путей* с теми или иными свойствами и т.п. Глобальный анализ графа следует отличать от

локального, примером которого может служить задача определения множеств *предшественников* и *преемников* фиксированной вершины ориентированного графа.

Базой для решения многих задач глобального анализа являются так называемые алгоритмы обхода или поиска.

Необходимо уметь обходить все вершины графа таким образом, чтобы каждая вершина была отмечена ровно один раз. Обычно такое „путешествие“ по графу сопровождается нумерацией вершин графа в том порядке, в котором они отмечаются, а также определенной „маркировкой“ *ребер* (или *дуг*) графа.

Существуют две основные стратегии таких обходов: ***поиск в глубину*** и ***поиск в ширину***. Эти стратегии можно описать так.

При поиске в глубину, отправляясь в „путешествие“ по графу из некоторой начальной вершины v_0, мы действуем следующим образом. Пусть, „путешествуя“, мы достигли некоторой вершины v (в начале „путешествия“ $v = v_0$). Отмечаем вершину v и просматриваем вершины из ее *списка смежности* $L[v]$.

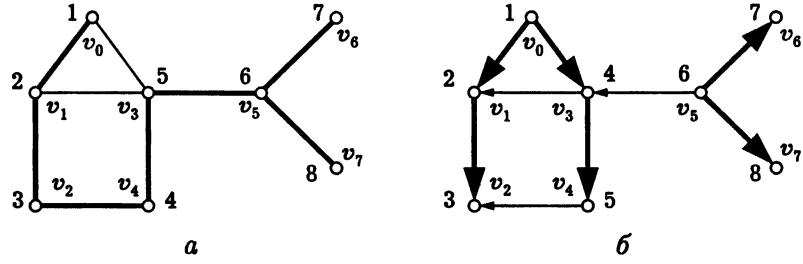
При условии, что в этом списке существует хотя бы одна неотмеченная вершина, продолжаем „путешествие“ из первой такой вершины w, действуя как описано выше — „ныряем“ вглубь, т.е. просматриваем вершины списка смежности $L[w]$ вершины w, откладывая анализ других элементов $L[v]$ как возможных продолжений поиска „на потом“.

Если же неотмеченных вершин в $L[v]$ нет, то возвращаемся из v в ту вершину, из которой мы в нее попали, и продолжаем анализировать список смежности этой вершины.

„Путешествие“ прекратится в тот момент, когда мы вернемся в начальную вершину v_0 и либо все вершины будут отмеченными, либо окажется, что неотмеченные вершины есть, но из v_0 больше никуда пойти нельзя. В последнем случае возможны продолжение поиска из новой вершины или остановка, что определяется конкретной версией алгоритма.

Заметим, что результат поиска в глубину зависит от порядка вершин в списках смежности, представляющих граф, а также от выбора начальной вершины v_0.

На рис. 5.20 показаны примеры поиска в глубину на неориентированном и ориентированном графах.



а *б*

Рис. 5.20

Рассмотрим неориентированный граф, изображенный на рис. 5.20, *а*. Списки смежности вершин графа удобно задать *кортежами*:

$$\begin{cases} v_0 \to (v_1,\, v_3), \quad v_1 \to (v_0,\, v_2,\, v_3), \quad v_2 \to (v_1,\, v_4), \\ v_3 \to (v_0,\, v_1,\, v_4,\, v_5), \quad v_4 \to (v_2,\, v_3), \\ v_5 \to (v_3,\, v_6,\, v_7), \quad v_6 \to (v_5), \quad v_7 \to (v_5). \end{cases} \tag{5.1}$$

Здесь запись $v_i \to (v_k, \ldots, v_m)$ означает, что кортеж $(v_k, \ldots, v_m)$ задает список смежности вершины v_i.

Результаты работы алгоритма с использованием приведенных списков смежности проиллюстрированы графически. При поиске отмечаем вершины, присваивая им номера. При этом каждая новая вершина получает номер, на единицу больший, чем текущая. Вершине v_0 присвоим номер 1. Номера остальных вершин, присвоенные им в процессе поиска, приведены на графе. Ребра, по которым из текущей вершины переходим к новой, изображены более толстыми линиями.

Прокомментируем поиск в глубину в ориентированном графе, изображенном на рис. 5.20, *б*. Пусть списки смежности

вершин имеют следующий вид:

$$v_0 \to (v_1, v_3), \quad v_1 \to (v_2), \quad v_2 \to (),$$
$$v_3 \to (v_1, v_4), \quad v_4 \to (v_2),$$
$$v_5 \to (v_3, v_6, v_7), \quad v_6 \to (), \quad v_7 \to ().$$

„Путешествие“ начинается в вершине v_0, и ей присваивается номер 1. Первой в списке смежности вершины v_0 стоит вершина v_1. Даем ей номер 2 и продолжаем поиск из этой вершины. В списке смежности последней только одна вершина v_2. Даем ей номер 3 и, так как ее список смежности пуст, возвращаемся в вершину v_1. В списке смежности вершины v_1 нет других вершин, кроме вершины v_2, уже помеченной номером 3, поэтому возвращаемся в вершину v_0.

Второй элемент списка смежности вершины v_0 — вершина v_3. Эта вершина новая. Помечаем ее номером 4 и переходим к рассмотрению ее списка смежности. Первая в списке смежности вершина v_1 уже получила номер 2. Поэтому переходим в новую вершину v_4 и помечаем ее номером 5. Единственный элемент ее списка смежности, вершина v_2, уже отмечена. Возвращаемся в вершину v_3. Здесь тоже нет никаких „отложенных продолжений“, и мы снова оказываемся в стартовой вершине v_0. Все вершины из списка смежности стартовой вершины уже отмечены, т.е. все возможные пути поиска из этой вершины пройдены. Тем не менее в графе остались непосещенные (непронумерованные) вершины. Следующей по исходной нумерации вершин графа является вершина v_5. Продолжим поиск из этой вершины и пометим ее номером 6. Первая вершина ее списка смежности —вершина v_3 — уже отмечена. Посещаем следующую вершину — вершину v_6 — и помечаем ее номером 7. Ее список смежности пуст. Возвращаемся в вершину v_5. Посещаем последнюю неотмеченную вершину из ее списка смежности — вершину v_7 — и помечаем ее номером 8. Поскольку все вершины из списка смежности вершины v_5 отмечены, поиск в глубину закончен.

На рис. 5.20, *б* более толстыми стрелками изображены *дуги*, по которым из очередной текущей вершины мы переходили к новой.

Рассмотрим теперь поиск в ширину. При поиске в ширину „правила игры" такие: достигнув некоторой вершины v, отмечаем ее. Затем просматриваем ее список смежности $L[v]$ и отмечаем все ранее не отмеченные вершины списка (при старте поиска $v = v_0$). После того как отмечены все вершины из $L[v]$, вершину v считаем полностью обработанной и продолжаем обработку вершин из списка $L[v]$ по очереди согласно описанным правилам.

Именно в обработке сразу всего списка смежности текущей вершины заключается принципиальное отличие поиска в ширину от поиска в глубину: там мы „ныряли" как можно „глубже", а здесь идем с „бреднем", „загребая" сразу все, что можно.

Поиск в ширину заканчивается, когда все вершины полностью обработаны или продолжение поиска невозможно.

Для сравнения возьмем неориентированный и ориентированный графы предыдущего примера и рассмотрим для них поиск в ширину (рис. 5.21).

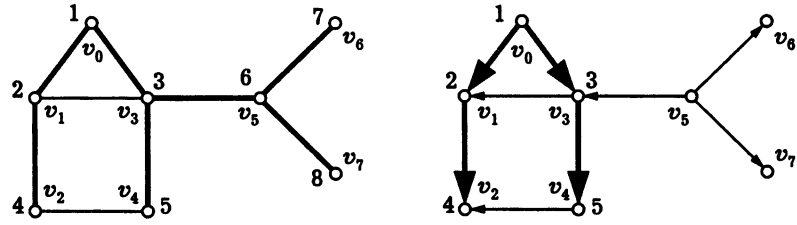


Рис. 5.21

Для неориентированного графа (см. рис. 5.21, *а*) поиск из стартовой вершины v_0 будем осуществлять так. Стартовой вершине присвоим номер 1. Затем пронумеруем все вершины из списка смежности стартовой вершины в порядке следования

их в списке. При этом вершина v_1 получит номер 2, а вершина v_3 — номер 3. Теперь, когда все вершины из списка смежности вершины v_0 отмечены, перейдем в первую по очереди вершину v_1. В ее списке смежности только одна ранее не отмеченная вершина — v_2. Присвоим ей номер 4 и перейдем в вершину v_3. На этом этапе номер 5 получит вершина v_4, а номер 6 — вершина v_5. Затем перейдем в вершину v_2. Поскольку все вершины из списка смежности вершины v_2 уже отмечены, перейдем в вершину v_4. Здесь также все вершины из списка смежности отмечены, поэтому перейдем в вершину v_5. Просмотрим неотмеченные вершины из ее списка смежности и отметим вершины v_6 и v_7, присваивая им номера 7 и 8 соответственно. Теперь, когда список смежности вершины v_5 обработан полностью, перейдем в вершину v_6. Так как в ее списке смежности нет неотмеченных вершин, перейдем в вершину v_7. Здесь также в списке смежности нет неотмеченных вершин. Все вершины обработаны, и поиск в ширину для графа закончен.

Результаты поиска в ширину из вершины v_0 в ориентированном графе представлены на рис. 5.21, *б*.

Обратим внимание на то, что нумерация вершин при поиске в ширину отличается от нумерации вершин при поиске в глубину. Так, в ориентированном графе на рис. 5.21, *б* мы сразу отмечаем оба элемента списка смежности вершины v_0. Поэтому вершине, получившей при поиске в глубину номер 4, при поиске в ширину присваивается номер 3.

Перейдем теперь к подробному описанию алгоритмов поиска в глубину и поиска в ширину.

Рассмотрим алгоритм поиска в глубину в неориентированном графе. Будем считать, что граф задан списками смежности, собранными в ***массив лидеров***.

При поиске вершины графа нумеруются в порядке их посещения. Номер вершины v графа, присваиваемый ей при поиске в глубину, обозначим $D[v]$ и будем называть ***D-номером***.

В процессе обхода будем находить ***фундаментальные циклы*** графа. Понятие фундаментального цикла в неориентированном графе вводится следующим образом. Пусть в неориентированном графе $G = (V, E)$ произвольно фиксирован *максимальный остовный лес* (см. теорему 5.3). Для связного графа это будет максимальное остовное дерево. Множество его ребер обозначим T. Все ребра из T назовем ***древесными***, а ребра исходного графа G, не принадлежащие T, — ***обратными***. Таким образом, фиксируя в графе G максимальный остовный лес, мы тем самым фиксируем разбиение множества его ребер на подмножества древесных и обратных ребер. В силу максимальности остовного леса любая пара вершин графа G, принадлежащих одной и той же компоненте связности, соединена цепью, все ребра которой являются древесными. Возьмем теперь произвольно две вершины u и v, принадлежащие одной и той же компоненте связности графа G. Пусть эти вершины соединены обратным ребром. Поскольку они также соединены цепью, содержащей исключительно древесные ребра, то существует цикл, содержащий эти две вершины, в котором будет единственное обратное ребро $\{u, v\}$. Любой цикл графа G, содержащий только одно обратное ребро, назовем фундаментальным.

Докажем, что не существует двух различных фундаментальных циклов, содержащих одно и то же обратное ребро (рис. 5.22). Действительно, если предположить противное, то возникает цикл C, содержащий лишь древесные ребра, что невозможно.

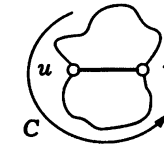


Рис. 5.22

Итак, фиксируя в графе максимальный остовный лес, мы тем самым фиксируем взаимно однозначное соответствие между множеством обратных ребер и множеством фундаментальных циклов.

Одним из способов построения максимального остовного леса может служить поиск в глубину. Именно при поиске в глубину всякое ребро, по которому из текущей вершины мы

попадаем в неотмеченную ранее вершину, будет древесным. Каждое ребро, не являющееся древесным, будет обратным. Максимальный остовный лес, находимый с помощью алгоритма поиска в глубину, называют ***остовным лесом поиска в глубину*** или ***глубинным остовным лесом***.

Заметим, что классификация ребер зависит от хода работы алгоритма, который определяется стартовой вершиной и расположением вершин в списках смежности.

Таким образом, алгоритм поиска в глубину в неориентированном графе может быть использован для вычисления множества фундаментальных циклов графа, т.е. решения одной из задач глобального анализа.

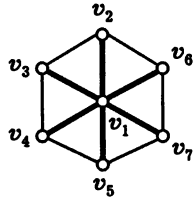


Рис. 5.23

Отметим, что не всякий максимальный остовный лес может быть получен с помощью алгоритма поиска в глубину. Например, для графа, изображенного на рис. 5.23, выделенный максимальный остовный лес нельзя получить при поиске в глубину, из какой бы вершины мы не начинали поиск.

Для организации работы алгоритма поиска в глубину используется способ хранения данных, называемый ***стеком***. Элементы в стеке упорядочиваются в порядке поступления. В стек можно добавлять новые элементы и из него можно извлекать элементы. При этом доступен только последний добавленный элемент — ***вершина стека***, чем и определяется режим работы стека (последним пришел — первым ушел). Образно стек можно представить в виде стопки тарелок. Из стопки можно взять только верхнюю тарелку и только наверх можно добавить новую. Обычно стек реализуется в виде списка.

В алгоритме поиска в глубину используется стек вершин. Мы будем считать, что из стека можно считывать информацию без изменения его содержимого, но в том порядке, в каком ее удаляли бы из стека. Указанная операция считывания понадобится нам для поиска фундаментальных циклов.

Алгоритм поиска в глубину

Вход. Граф $G = (V, E)$, заданный списками смежности.

Выход. Множества *Tree* древесных и *Back* обратных ребер соответственно; множество F_c фундаментальных циклов, массив D, содержащий D-номера вершин.

0. Просмотреть массив лидеров и сформировать множество V_0 вершин графа. Это множество используется для хранения новых (не обработанных алгоритмом) вершин. В ходе работы алгоритма обработанные вершины будут удаляться из этого множества.

В процессе работы алгоритма для каждой вершины v необходимо знать, какие вершины из ее списка смежности $L[v]$ обработаны на предыдущих шагах работы алгоритма. Для этого формируется массив L_p размера $|V_0|$, в котором хранятся номера первых еще не обработанных алгоритмом вершин списка смежности $L[v]$ каждой вершины v. Первоначально все элементы массива L_p полагают равными 1.

Стек вершин $STACK$ и список вершин фундаментального цикла *Cycle* положить пустыми ($STACK = \varnothing$, $Cycle = \varnothing$).

Множества *Tree* древесных ребер, *Back* обратных ребер и F_c фундаментальных циклов положить пустыми ($Tree = \varnothing$, $Back = \varnothing$, $F_c = \varnothing$). Массив D-номеров D обнулить.

Задать начальную вершину для поиска ($v = v_0$). (Обычно это первая вершина массива лидеров.) Счетчик D-номеров вершин *count* положить равным 1 ($count = 1$).

1. Если множество V_0 не пусто ($V_0 \neq \varnothing$), перейти на шаг **2**, если иначе — на шаг **8** (выход).

2. Если стек пуст ($STACK = \varnothing$) и алгоритм стартует из вершины v_0 ($count = 1$), то перейти на шаг **3**, если иначе, то выбрать из множества V_0 произвольную вершину, из которой поиск в глубину будет продолжен ($v = u$, $u \in V_0$), и перейти на шаг **3**.

3. Поместить текущую вершину v в стек $STACK$. Присвоить вершине v D-номер ($D[v] = count$). Увеличить счетчик

D-номеров на 1 ($count = count + 1$). Удалить вершину v из множества V_0 новых вершин. Перейти на шаг **4**.

4. Проверить по элементу $L_p[v]$, что не достигнут конец списка смежности $L[v]$ текущей вершины v. Если в списке смежности есть еще не обработанные вершины, получить из списка смежности очередную вершину w, увеличить $L_p[v]$ на 1 и перейти на шаг **6**.

Если список смежности $L[v]$ вершины v обработан полностью, удалить вершину v из стека $STACK$ и перейти на шаг **5**.

5. Если стек $STACK$ пуст, вернуться на шаг **1**, если иначе, то в качестве текущей вершины v взять вершину, находящуюся в вершине стека, и перейти на шаг **4**.

6. Если вершина w новая ($w \in V_0$), то добавить ребро $\{v, w\}$ в множество древесных ребер

$$(Tree = Tree \cup \{\{v, w\}\}),$$

сделать вершину w текущей ($v = w$) и вернуться на шаг **3**.

Если вершина w не новая ($w \notin V_0$), то перейти на шаг **7**.

7. Если ребра $\{v, w\}$ нет среди древесных ($\{v, w\} \notin Tree$), поместить ребро в список обратных ребер

$$(Back = Back \cup \{\{v, w\}\}).$$

Поместить вершину w в список $Cycle$. Читать содержимое стека $STACK$, начиная с вершины стека, до появления вершины w и помещать в список $Cycle$ (включая вершину w), т.е. формировать фундаментальный цикл, соответствующий обратному ребру $\{v, w\}$.

Добавить список $Cycle$ в множество фундаментальных циклов F_c

$$(F_c = F_c \cup \{Cycle\}).$$

Список $Cycle$ положить пустым ($Cycle = \varnothing$).

Вернуться на шаг **4**.

8. Закончить работу.

Пусть для неориентированного графа, изображенного на рис. 5.20, *а*, списки смежности имеют вид (5.1). При выполнении поиска в глубину выделенные ребра являются древесными, остальные — обратными. Около каждой вершины указан присвоенный ей D-номер. Фундаментальные циклы — в порядке их нахождения — таковы: v_1, v_2, v_4, v_3, v_1 и $v_0, v_1, v_2, v_4, v_3, v_0$.

Поиск в глубину в неориентированном графе можно использовать и для нахождения числа его компонент. Очевидно, что это число равно числу опустошений стека от начала до конца поиска в глубину. Регистрируя удаляемые из стека вершины, можно после очередного его опустошения распечатать номера вершин, принадлежащих данной компоненте.

В ориентированном графе поиск в глубину реализуется во многом аналогично поиску в глубину в неориентированном графе. В этом случае возникает остовный ориентированный лес поиска в глубину, дуги которого — это в точности те дуги, по которым в процессе работы алгоритма переходят от очередной вершины к новой, еще не отмеченной вершине. Можно показать, что это максимальный остовный лес исходного графа.

Слабыми компонентами этого глубинного остовного леса будут некоторые *ориентированные деревья*: поэтому, используемая далее терминология из теории деревьев относится к той или иной слабой компоненте глубинного остовного леса. В ориентированном графе вершинам также присваиваются D-номера. Но классификация дуг при поиске в глубину в ориентированном графе сложнее по сравнению с аналогичной классификацией ребер при поиске в глубину в неориентированном графе. Различают четыре класса дуг:

1) ***древесные дуги*** — каждая такая *дуга ведет* от *отца* к *сыну* в глубинном остовном лесу;

2) ***прямые дуги*** — каждая такая дуга ведет от *подлинного предка* к *подлинному потомку* (но не от отца к сыну) в глубинном остовном лесу;

3) ***обратные дуги*** — от *потомков* к *предкам* (включая все петли);

4) ***поперечные дуги*** — все дуги, не являющиеся ни древесными, ни прямыми, ни обратными.

Следовательно, в результате работы алгоритма будут получены множества *Tree* — древесных дуг, *Back* — обратных дуг, *Forward* — прямых дуг, C — поперечных дуг и массив D, содержащий D-номера вершин.

В процессе работы алгоритма по сравнению с алгоритмом поиска в глубину в неориентированном графе имеется ряд особенностей. Так, если очередная вершина w, извлеченная из списка смежности текущей вершины v, новая, т.е. $w \in V_0$, то дуга (v, w) является древесной. Следует добавить дугу (v, w) в множество древесных дуг ($Tree = Tree \cup \{(v, w)\}$), сделать вершину w текущей ($v = w$) и начать ее обработку.

Если вершина w не новая ($w \notin V_0$), то дуга (v, w) будет либо прямой, либо обратной, либо поперечной.

Если D-номер вершины v строго меньше D-номера вершины w ($D[v] < D[w]$), то дуга (v, w) является прямой. Добавляем ее в множество прямых дуг *Forward* ($Forward = Forward \cup \{(v, w)\}$).

Если D-номер вершины v не меньше D-номера вершины w ($D[v] \geqslant D[w]$), необходимо проверить, есть ли в стеке $STACK$ вершина w. Если вершина w находится в стеке, то дуга (v, w) является обратной и ее следует добавить в множество обратных дуг *Back* ($Back = Back \cup \{(v, w)\}$). Если вершины w в стеке нет, то дуга является поперечной. Тогда нужно добавить ее в множество поперечных дуг C ($C = C \cup \{(v, w)\}$). Далее следует перейти к рассмотрению очередной вершины из списка смежности текущей вершины (если он не пуст).

Если стек пуст, но не все вершины ориентированного графа обработаны, поиск продолжают из любой необработанной вершины.

В случае ориентированного графа поиск контуров на базе поиска в глубину существенно сложнее, чем в случае неориентированного графа, и здесь он не рассматривается. Однако

можно доказать следующий критерий бесконтурности ориентированного графа: ориентированный граф является бесконтурным тогда и только тогда, когда при поиске в глубину от некоторой начальной вершины множество обратных дуг оказывается пустым.

Заметим также, что для ориентированного графа нет такой простой связи между числом опустошений стека и числом компонент, как для неориентированного графа.

На базе алгоритма поиска в глубину может быть разработан эффективный алгоритм поиска *бикомпонент* в ориентированном графе. Однако здесь мы не будем останавливаться на задаче поиска бикомпонент, так как эта задача обсуждается в рамках общей задачи о путях в графах (см. **5.6**).

Можно показать, что алгоритм поиска в глубину имеет сложность порядка $\max(|V|, |E|)$.

Пример 5.7. Проведем поиск в глубину на ориентированном графе, представленом на рис. 5.24. Поиск начнем из вершины v_0. Пусть списки смежности вершин имеют следующий вид:

$$\begin{cases} v_0 \to (v_2, v_3, v_1), \quad v_1 \to (v_3), \quad v_2 \to (v_4, v_3), \\ v_3 \to (v_4), \quad v_4 \to (v_1), \quad v_5 \to (v_1), \\ v_6 \to (v_3, v_5), \quad v_7 \to (v_5, v_6). \end{cases} \tag{5.2}$$

Результаты выполнения поиска в глубину представлены на рисунке: около каждой вершины указан ее D-номер, все дре-

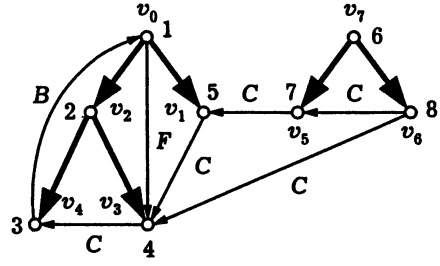


Рис. 5.24

весные дуги выделены более толстыми стрелками, а прямые, обратные и поперечные помечены буквами F, B, C соответственно.

Первое опустошение стека происходит после обработки первых пяти вершин (в порядке обхода): v_0, v_2, v_4, v_3, v_1.

После опустошения поиск продолжается из вершины v_7, которой присваивается D-номер 6. (Напомним, что в приведенном выше алгоритме после опустошения стека для продолжения поиска выбирается любая из необработанных вершин.) #

В отличие от алгоритма поиска в глубину, алгоритм поиска в ширину использует не стек, а *очередь* вершин. Мы дадим здесь вариант поиска в ширину, когда, начиная поиск в некоторой начальной вершине v_0, мы останавливаемся при первом же опустошении очереди. Это значит, что некоторые из вершин могут остаться неотмеченными. Таким образом может быть решена задача распознавания достижимости от заданной вершины. Мы совместим также поиск в ширину с вычислением длины кратчайшего пути от v_0 до любой вершины графа. Будем предполагать, что вершины графа пронумерованы без пропусков числами от 0 до N: $V = \{v_i : i = \overline{0, N}\}$.

Поиск в ширину рассматриваем только для ориентированного графа.

Алгоритм поиска в ширину в ориентированном графе

Вход. Граф $G = (V, E)$, заданный списками смежности; v_0 — начальная вершина (не обязательно первый элемент массива лидеров).

Выход. Массив M меток вершин, где каждая метка равна длине пути от v_0 до v.

0. Очередь Q положить пустой ($Q = \varnothing$). Все вершины пометить как недостижимые из вершины v_0, присваивая элементам массива M значение $+\infty$ ($M[v_i] = +\infty$, $i = \overline{1, N}$).

Стартовую вершину v_0 пометить номером 0, т.е. длину пути от стартовой вершины v_0 до самой себя положить равной 0 ($M[v_0] = 0$). Поместить вершину v_0 в очередь Q. Перейти на шаг **1**.

1. Если очередь Q не пуста ($Q \neq \varnothing$), то из „головы“ очереди извлечь (с удалением из очереди) вершину u и перейти на шаг **2**. Если очередь пуста, перейти на шаг **3**.

2. Если список смежности $L(u)$ вершины u пуст, вернуться на шаг **1**.

Если список смежности $L(u)$ вершины u не пуст, для каждой вершины w из списка смежности, где $M[w] = +\infty$, т.е. вершины, которую еще не посещали, положить длину пути из стартовой вершины v_0 до вершины w равной длине пути от v_0 до вершины u плюс одна дуга ($M[w] = M[u] + 1$), т.е. отметить вершину w и поместить ее в очередь Q. После просмотра всех вершин списка смежности $L(u)$ вернуться на шаг **1**.

3. Распечатать массив M. Закончить работу.

Алгоритм поиска в ширину может быть дополнен процедурой „обратного хода“, определяющей номера вершин, лежащих на кратчайшем пути из вершины v_0 в данную вершину u. Для этого необходимо завести массив PR размера $|V|$, каждый элемент $PR[w]$ которого содержит номер той вершины, из которой был осуществлен переход в вершину w при ее пометке.

Если вершина w находится в списке смежности $L(u)$ вершины u, заполнение элемента массива $PR[w]$ происходит при изменении метки вершины w $M[w]$ с $+\infty$ на единицу. При этом в элементе $PR[w]$ сохраняется номер вершины u ($PR[w] = u$). Для начальной вершины $PR[v_0]$ можно положить равным 0, в предположении, что начальная вершина v_0 имеет номер 0 и остальные вершины пронумерованы от 1 до N.

Сложность рассмотренного алгоритма поиска в ширину, известного под названием „Алгоритм волнового фронта“, составляет $O(\max(|V|, |E|))$.

Пример 5.8. На рис. 5.25 дан пример работы алгоритма волнового фронта (при поиске из вершины v_0) для ориентированного графа, изображенного на рис. 5.24. Списки смежности ориентированного графа имеют вид (5.2).

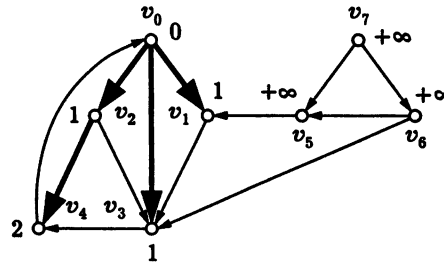


Рис. 5.25

Около каждой вершины v_i поставлена метка $M[v_i]$, которую получает вершина при поиске в ширину. Выделены дуги, составляющие кратчайшие пути из вершины v_0 в остальные. Отметим, что вершины v_5, v_6, и v_7 не достижимы из v_0 и их начальные метки $+\infty$ остались без изменения. При рассмотренном ходе алгоритма массив PR будет содержать следующие номера вершин: $PR[v_0] = 0$, $PR[v_1] = 0$, $PR[v_2] = 0$, $PR[v_3] = 0$, $PR[v_4] = 2$. Для остальных вершин соответствующие значения не определены, поскольку они не „посещались".

Используя массив PR, восстановим кратчайший путь из вершины v_0 в вершину v_4. Поскольку $PR[v_4] = 2$, а $PR[v_2] = 0$, искомый путь есть v_0, v_2, v_4.

5.6. Задача о путях во взвешенных ориентированных графах

Среди задач анализа *ориентированных графов* весьма важны следующие задачи.

1. Вычисление для заданного ориентированного графа его *матрицы достижимости.* Эту задачу будем называть ***задачей*** построения ***транзитивного замыкания ориентиро-***

ванного графа. Такое название связано с тем, что матрицу достижимости можно рассматривать как матрицу *транзитивного* и *рефлексивного замыкания бинарного отношения непосредственной достижимости* в ориентированном графе.

2. Вычисление наименьших расстояний между всеми парами *вершин* в ориентированном графе. Эту задачу будем называть ***задачей о кратчайших расстояниях.***

Задачу о кратчайших расстояниях можно сформулировать так. Пусть задан *взвешенный ориентированный граф* и пусть из вершины v *достижима* вершина w. Фиксируем какой-либо *путь* S, ведущий из v в w. ***Расстоянием*** от вершины v до вершины w ***по пути*** S называют сумму меток дуг, входящих в этот путь, а наименьшим — минимальное из расстояний между вершинами v и w по всем возможным путям.

Отметим, что задача о кратчайших расстояниях не всегда имеет решение. Например, если в ориентированном графе есть *петля*, метка которой — отрицательное число, то по этой петле можно проходить сколько угодно раз и тем самым уменьшать сумму меток дуг пути, включающего эту петлю, до любого наперед заданного значения.

3. Перечисление всех путей между двумя произвольными вершинами. Эту задачу будем называть ***задачей о перечислении путей.*** При ее решении требуется для любой заданной пары вершин u и v ориентированного графа получить все пути, для которых u является началом, а v — концом.

Все указанные задачи (перечисленные в порядке возрастания сложности) можно решить в рамках единого подхода, суть которого сводится к следующему.

Ранее мы ввели понятие взвешенного неориентированного (ориентированного) графа и метки ребер (дуг) определили как числа, поставленные в соответствие ребру (дуге). Обобщим это понятие для ориентированного графа.

Определение 5.13. ***Взвешенным*** (или ***размеченным***) ***ориентированным графом*** называют пару $W = (G, \varphi)$, где

$G = (V, E)$ — обычный ориентированный граф, а $\varphi\colon E \to \mathcal{R}$ — ***весовая функция*** (или ***функция разметки***) со значениями в некотором *идемпотентном полукольце* $\mathcal{R} = (R, +, \cdot, \mathbf{0}, \mathbf{1})$, причем $(\forall e \in E)(\varphi(e) \neq \mathbf{0})$.

Мы будем в этом случае также говорить, что ориентированный граф размечен над идемпотентным полукольцом $\mathcal{R}$. Часто полукольцо $\mathcal{R}$ является *замкнутым*, хотя это требование необязательно. Будем, однако, везде в этом пункте предполагать, что $\mathcal{R}$ — *полукольцо с итерацией*.

Пусть вершины ориентированного графа каким-либо образом пронумерованы. Тогда взвешенный ориентированный граф может быть задан матрицей A, элемент которой a_{ij} равен значению $\varphi((i, j))$ весовой функции на дуге (i, j), если из вершины i ведет дуга в вершину j, или *нулю полукольца* в противном случае. Эту матрицу будем называть ***матрицей меток дуг***.

Оказывается, что вычисление *итерации* A^* матрицы A дает решение всех сформулированных выше задач, если для каждой задачи выбирать соответствующее полукольцо. А именно в случае полукольца $\mathcal{B}$ (см. пример 3.2) получаем решение задачи о транзитивном замыкании, в случае полукольца $\mathcal{R}^+$ (см. пример 3.1) — решение задачи о кратчайших расстояниях*.

Будем называть задачу вычисления матрицы A^* для ориентированного графа, размеченного над произвольным полукольцом с итерацией, в частности над *замкнутым полукольцом*, ***общей задачей о путях*** во взвешенных ориентированных графах.

В такой общности подхода, когда множество, казалось бы, не связанных друг с другом задач решается единым алгоритмом, „настраиваемым“ на разные задачи выбором соответствующего идемпотентного полукольца (т.е. разных областей значений весовой функции), состоит несомненное преимущество

*О методе решения третьей из сформулированных выше задач см. задачу 7.36.

абстрактно-алгебраического подхода к решению таких задач на графах.

Рассмотрим теперь решение общей задачи о путях для произвольного замкнутого полукольца $\mathcal{R}$.

Метка пути, ведущего из вершины v_i в вершину v_j, есть *произведение в полукольце* $\mathcal{R}$ меток входящих в путь дуг в порядке их следования (для пути ненулевой *длины*) и есть $\mathbf{1}$ (*единица полукольца* $\mathcal{R}$) для пути нулевой длины.

Стоимость прохождения из вершины v_i в вершину v_j (или между i-й и j-й вершинами) — это *сумма в полукольце* $\mathcal{R}$ меток всех путей, ведущих из вершины v_i в вершину v_j.

Заметим, что сумма, определяющая стоимость прохождения, вообще говоря, есть *бесконечная сумма* замкнутого полукольца, т.е. *точная верхняя грань* соответствующей *последовательности* меток. Это связано с тем, что множество всех путей, ведущих из одной вершины графа в другую, в общем случае бесконечно (но, как можно доказать, не более чем счетно).

Аналогично можно определить стоимость прохождения из вершины в вершину по какому-либо множеству путей. Отметим, что если стоимость прохождения между парой вершин по какому-либо множеству путей равна $\mathbf{0}$, то это означает, что не существует пути, принадлежащего данному множеству путей, ведущего из первой вершины рассматриваемой пары во вторую вершину.

Матрица меток дуг является элементом *полукольца матриц над полукольцом* $\mathcal{R}$. В этом полукольце определены операции сложения и умножения матриц, а также возведение матрицы в неотрицательную степень. Докажем следующее утверждение.

Лемма 5.1. Элемент $a_{ij}^{(l)}$ матрицы A^l, $l \geqslant 0$, равен стоимости прохождения из вершины v_i в вершину v_j по всем путям длины l.

◀ Доказательство проведем индукцией по l. При $l = 0$ утверждение очевидно, так как $A^0 = E$, где E — единичная матрица,

которая будет матрицей стоимости прохождения по всем путям длины 0.

При $l = 1$ утверждение также очевидно. Далее,

$$a_{ij}^{(l)} = \sum_{k=1}^{n} a_{ik}^{(l-1)} a_{kj}.$$

Согласно предположению индукции, элемент $a_{ik}^{(l-1)}$ равен стоимости прохождения из вершины v_i в вершину v_k по всем путям длины $l-1$. Множество всех путей длины l из вершины v_i в вершину v_j, проходящих через фиксированную k-ю вершину так, что вершина v_k связана дугой с вершиной v_j ($v_k \to v_j$,) образуется путем присоединения дуги (v_k, v_j) к каждому из путей, ведущих из v_i в v_k и имеющих длину $l-1$. Тогда видно, что написанное выше выражение для элемента $a_{ij}^{(l)}$ дает стоимость прохождения из вершины v_i в вершину v_j по всем путям длины l (рис. 5.26). ▶

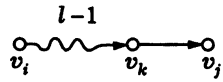


Рис. 5.26

Так как стоимость прохождения между парой вершин (v_i, v_j) равна сумме меток всех путей, ведущих из первой вершины во вторую, а указанную сумму можно можно получить, суммируя последовательно метки путей длины 0, длины 1, длины 2 и т.д., то матрица стоимостей взвешенного ориентированного графа с учетом доказанной леммы 5.1 может быть представлена в виде

$$C = A^0 + A^1 + A^2 + \ldots + A^n + \ldots = \sum_{n \geqslant 0} A^n = A^*.$$

До сих пор мы рассматривали матрицы над замкнутым полукольцом. Однако, если элементы матрицы A принадлежат некоторому полукольцу с итерацией, из теоремы 3.9 следует, что и все элементы матрицы стоимостей $C = A^*$ останутся в этом же полукольце. Таким образом, полученные результаты можно перенести на произвольное полукольцо с итерацией.

Теорема 5.4. Матрица стоимостей ориентированного графа G, размеченного над полукольцом с итерацией $\mathcal{R}$ (в частности, над замкнутым полукольцом), равна итерации матрицы A меток дуг ориентированного графа G. #

Для вычисления $C = A^*$ достаточно решить (т.е. найти наименьшее решение) в $\mathcal{R}$ при всех $j = \overline{1,n}$ систему уравнений

$$\xi = A\xi + \varepsilon_j,$$

где $\varepsilon_j \in \mathcal{R}^n$ — j-й единичный вектор, т.е. вектор, все элементы которого, кроме j-го, равны $\mathbf{0}$, а j-й равен единице полукольца $\mathcal{R}$. Наименьшее решение имеет вид $\xi = A^*\varepsilon_j$ (см. **3.3**). Тогда столбец $\xi = A^*\varepsilon_j$ есть j-й столбец матрицы A^*. Такой метод вычисления матрицы A^* аналогичен известному из линейной алгебры методу элементарных преобразований при вычислении обратной матрицы.

Выясним теперь смысл матрицы стоимостей $C = A^*$ для полуколец $\mathcal{B}$ и $\mathcal{R}^+$.

В первом из этих полуколец метка отдельного пути всегда равна 1 (так как метка дуги в размеченном над полукольцом графе не может, согласно определению, быть нулем полукольца). Следовательно, стоимость $c_{ij} = 1$, если существует хотя бы один путь из i-й вершины в j-ю, и $c_{ij} = 0$, если иначе. Другими словами, для полукольца $\mathcal{B}$ матрица стоимостей совпадает с матрицей достижимости ориентированного графа.

В полукольце $\mathcal{R}^+$ метка пути — это арифметическая сумма меток его дуг, так как умножение в $\mathcal{R}^+$ — это обычное арифметическое сложение. Поскольку сложение в $\mathcal{R}^+$ — это взятие наименьшего из слагаемых, то стоимость c_{ij} — это наименьшая из меток пути среди всех путей, ведущих из i-й вершины в j-ю, т.е. это и есть наименьшая длина пути между указанными вершинами. Таким образом, в полукольце $\mathcal{R}^+$ матрица стоимостей является матрицей кратчайших расстояний, т.е. наименьших длин путей между всеми парами вершин ориентированного графа.

Пример 5.9. Рассмотрим граф, изображенный на рис. 5.27, и решим для него задачу вычисления матрицы достижимости. На числовые метки дуг внимания пока не обращаем, считая, что ориентированный граф размечен над полукольцом $\mathcal{B}$ и метка каждой дуги равна 1, т.е. ориентированный граф задан матрицей

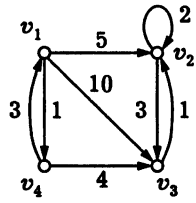


Рис. 5.27

$$A = \begin{pmatrix} 0 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \end{pmatrix}.$$

Запишем систему уравнений в полукольце $\mathcal{B}$ для определения первого столбца матрицы A^*:

$$\begin{cases} x_1 = & x_2 + x_3 + x_4 + 1, \\ x_2 = & x_2 + x_3 & + 0, \\ x_3 = & x_2 & + 0, \\ x_4 = x_1 & + x_3 & + 0. \end{cases} \tag{5.3}$$

Отметим, что часто нулевые слагаемые не записывают, как и в системах уравнений в *поле действительных чисел.*

Воспользуемся *методом последовательного исключения неизвестных* (см. **3.3**). Поскольку в правой части первого уравнения нет переменной x_1, можно исключить эту переменную из системы, подставив в остальные уравнения. С учетом *идемпотентности* сложения получим

$$\begin{cases} x_2 = x_2 + x_3 & + 0, \\ x_3 = x_2 & + 0, \\ x_4 = x_2 + x_3 + x_4 + 1. \end{cases}$$

Из второго уравнения имеем $x_2 = 1^*(x_3 + 0)$. В полукольце $\mathcal{B}$ итерация любого элемента равна единице полукольца. Поэтому $x_2 = x_3 + 0$. Исключив x_2 из системы, получим

$$\begin{cases} x_3 = x_3 & + 0, \\ x_4 = x_3 + x_4 + 1. \end{cases}$$

Далее вычислим $x_3 = 1^*0 = 1 \cdot 0 = 0$. Подставив $x_3 = 0$ в последнее уравнение, найдем $x_4 = 1^*1 = 1$.

Итак, первый столбец A^* есть

$$\begin{pmatrix} 1 \\ 0 \\ 0 \\ 1 \end{pmatrix}.$$

Второй столбец определяется из системы

$$\begin{cases} x_1 = & x_2 + x_3 + x_4 + 0, \\ x_2 = & x_2 + x_3 & + 1, \\ x_3 = & x_2 & + 0, \\ x_4 = x_1 & + x_3 & + 0. \end{cases}$$

Исключая x_1, получаем

$$\begin{cases} x_2 = x_2 + x_3 & + 1, \\ x_3 = x_2 & + 0, \\ x_4 = x_2 + x_3 + x_4 + 0. \end{cases}$$

Из второго уравнения имеем $x_2 = 1^*(x_3 + 1) = x_3 + 1$. Далее находим

$$\begin{cases} x_3 = x_3 & + 1, \\ x_4 = x_3 + x_4 + 1. \end{cases}$$

Отсюда $x_3 = 1^*1 = 1$ и $x_4 = x_4 + 1$. В итоге $x_4 = 1^*1 = 1$, $x_2 = 1 + 1 = 1$, $x_1 = 1 + 1 + 1 + 0 = 1$. Таким образом, второй столбец A^* есть

$$\begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \end{pmatrix}.$$

Аналогично вычисляем третий и четвертый столбцы и в результате получаем матрицу A^*:

$$A^* = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 \end{pmatrix}.$$

Анализ этой матрицы показывает (см. **5.2**), что данный граф связен и имеет две бикомпоненты: $\{v_1, v_4\}$ и $\{v_2, v_3\}$.

Заметим, что в полукольце $\mathcal{B}$ можно упростить решение систем уравнений, воспользовавшись свойствами полукольца. Легко видеть, что наименьшее решение уравнения

$$x_k = \sum_{i=1}^{n} a_i x_i + 1$$

есть $x_k = 1$ и не зависит от значений переменных в правой части уравнения. С учетом этого решение системы (5.3) упростится. Так, из первого уравнения сразу получаем $x_1 = 1$. Тогда четвертое уравнение принимает вид $x_4 = x_3 + 1$, откуда $x_4 = 1$. Поскольку x_1 и x_4 не входят в оставшиеся два уравнения, их решение нужно искать, используя метод исключения.

Пример 5.10. Для графа, изображенного на рис. 5.27, вычислим матрицу кратчайших расстояний, перейдя к полукольцу $\mathcal{R}^+$. Договоримся, что для упрощения записи ∞ здесь будем понимать как $+\infty$.

Наш взвешенный ориентированный граф задается теперь следующей матрицей:

$$A = \begin{pmatrix} \infty & 5 & 10 & 1 \\ \infty & 2 & 3 & \infty \\ \infty & 1 & \infty & \infty \\ 3 & \infty & 4 & \infty \end{pmatrix}. \tag{5.4}$$

Система для вычисления первого столбца матрицы A^* имеет вид

$$\begin{cases} x_1 = & 5x_2 + 10x_3 + 1x_4 + 0, \\ x_2 = & 2x_2 + 3x_3 & + \infty, \\ x_3 = & 1x_2 & + \infty, \\ x_4 = 3x_1 & + 4x_3 & + \infty. \end{cases}$$

Обратим внимание на нюансы, связанные с работой в полукольце $\mathcal{R}^+$: элементы 1 и 0 не являются *единицей* и *нулем полукольца*, т.е. $x \neq x + 0$ и $x \neq 1 \cdot x$ в общем случае. Напомним, что сложение в полукольце $\mathcal{R}^+$ — взятие наименьшего из двух чисел, а умножение — обычное арифметическое сложение. Заметим, что наличие слагаемого 0 в любой сумме (в полукольце) означает, что вся сумма равна 0; слагаемое $+\infty$ можно не записывать (как нуль полукольца).

Из первого уравнения системы сразу следует, что $x_1 = 0$, так как одно из слагаемых в правой части есть элемент 0. Напомним, что итерация любого элемента в рассматриваемом полукольце равна единице полукольца. Учитавая этот факт, из второго уравнения получаем

$$x_2 = 2^*(3x_3 + \infty) = 3x_3.$$

Исключая x_2 из остальных уравнений системы и учитывая, что $x_1 = 0$, получаем

$$\begin{cases} x_2 = 3x_3 + \infty, \\ x_3 = 1(3x_3) + \infty, \\ x_4 = 3 \cdot 0 + 4x_3 + \infty. \end{cases}$$

Далее, из второго уравнения имеем

$$x_3 = (1 \cdot 3)x_3 + \infty = 4x_3 + \infty,$$

откуда $x_3 = 4^* \cdot \infty = \infty$, и поэтому

$$x_4 = 3 \cdot 0 + 4 \cdot \infty + \infty = 3 + \infty = 3.$$

Подставляя найденное значение x_3 в выражение для x_2, получаем $x_2 = \infty$. Первый столбец искомой матрицы вычислен:

$$\begin{pmatrix} 0 \\ \infty \\ \infty \\ 3 \end{pmatrix}.$$

Этот столбец содержит кратчайшие расстояния от всех вершин графа до вершины v_1. Наличие в нем нулей полукольца во второй и третьей строках говорит о том, что вершина v_1 не достижима из вершин v_2 и v_3.

Аналогично вычисляются остальные столбцы матрицы A^*. Результат будет следующим:

$$A^* = \begin{pmatrix} 0 & 5 & 5 & 1 \\ \infty & 0 & 3 & \infty \\ \infty & 1 & 0 & \infty \\ 3 & 5 & 4 & 0 \end{pmatrix}.$$

Для данного простого ориентированного графа легко сопоставить полученный алгебраический результат с результатом „визуального“ анализа ориентированного графа. Рассмотрим, например, пару вершин (v_1, v_3). В ориентированном графе есть различные пути из вершины v_1 в вершину v_3. Легко видеть, что заведомо „не выгодны“ пути, содержащие контуры и петли, поэтому их рассматривать не будем и вычислим метки по *простым путям*. По пути $v_1 \to v_4 \to v_3$ сумма меток равна 5, по пути $v_1 \to v_3$ — 10, а по пути $v_1 \to v_2 \to v_3$ — 8. Кратчайшее расстояние — 5, что совпадает с ответом, полученным алгебраически: элемент a^*_{13} также равен 5. #

Помимо изложенного есть еще один способ вычисления замыкания матрицы с элементами в замкнутом полукольце. Он основан на понятии пути ранга k из вершины v_i в вершину v_j.

Пусть в ориентированном графе выбрана и зафиксирована нумерация вершин. Будем полагать, что все вершины пронумерованы подряд натуральными числами, начиная с 1.

Путь $v_{i_0} \to v_{i_1} \to \ldots \to v_{i_m}$ длины m называют путем ранга k при $m > 1$, если k — наибольшее среди чисел $i_1, \ldots, i_{m-1}$, и путем ранга 0 при $m = 1$. Путь нулевой длины также считают путем ранга 0. Таким образом, ***ранг пути*** — это максимальный номер вершины, в которую разрешено заходить по пути из v_i в v_j (исключая вершины v_i и v_j). Путь ранга 0 не содержит промежуточных вершин. Максимальный ранг пути в ориентированном графе при указанном выше способе нумерации равен числу его вершин.

Пример 5.11. В ориентированном графе, изображенном на рис. 5.27, путь $v_1 \to v_4 \to v_1$ имеет ранг 4, путь $v_4 \to v_1 \to \to v_2$ — ранг 1, путь $v_4 \to v_1 \to v_3 \to v_2$ — ранг 3. Пути $v_4 \to v_3 \to v_2$, $v_4 \to v_1 \to v_3 \to v_2$ и $v_4 \to v_2 \to v_2 \to v_3 \to v_2$ также имеют ранг 3. #

Обозначим через $C^{(k)}$ матрицу стоимостей прохождения между различными парами вершин по всем путям ранга, не превосходящего k. Ее элемент $c_{ij}^{(k)}$ содержит стоимость прохождения из вершины v_i в вершину v_j по всем путям рангов 0, 1, ..., $k-1$, k.

Выведем формулу для вычисления элемента $c_{ij}^{(k)}$ матрицы $C^{(k)}$. Для этого заметим следующее. По пути ранга, не большего k, из вершины v_i в вершину v_j можно пройти следующими способами:

1) идя из вершины v_i в вершину v_j по некоторому пути ранга, не превосходящего $k-1$, т.е. минуя вершину v_k;

2) сначала идя из v_i в v_k по пути ранга, не большего $k-1$, затем „покрутившись“ любое число раз (а может быть, и ни разу) по какому-либо контуру или любому замкнутому пути из v_k в v_k ранга, не большего $k-1$, и, наконец, идя из вершины v_k в вершину v_j по пути ранга, не большего $k-1$ (рис. 5.28).

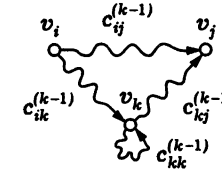


Рис. 5.28

При первом способе следования стоимость прохождения из вершины v_i в v_j по всем путям ранга, не большего $k-1$, составит $c_{ij}^{(k-1)}$.

При втором способе следования стоимость прохождения из v_i в v_k по всем путям ранга, не большего $k-1$, будет равна $c_{ik}^{(k-1)}$. Стоимость прохождения из v_k в v_k по всем замкнутым путям ранга, не большего $k-1$, составит $\left(c_{kk}^{(k-1)}\right)^*$.

Поясним это в частном случае, когда вершина v_k содержится в каком-то одном контуре. Пусть Γ — такой контур, а μ_Γ — метка этого контура. Тогда очевидно, что метка пути, образованного нуль-кратным прохождением по контуру Γ, равна единице полукольца (как метка всякого пути длины 0), метка же пути, образованного m-кратным прохождением по Γ при $m \geqslant 1$, равна μ_Γ^m. Следовательно, стоимость прохождения по всем путям, которые получаются при произвольном числе прохождений по контуру Γ, составит $\sum\limits_{m\geqslant 0} \mu_\Gamma^m = \mu_\Gamma^*$.

Стоимость прохождения из вершины v_k в вершину v_j по пути ранга, не большего $k-1$, равна $c_{kj}^{(k-1)}$ (см. рис. 5.28). Таким образом, стоимость прохождения по пути ранга, не большего k, при указанном способе следования составит

$$c_{ik}^{(k-1)}\left(c_{kk}^{(k-1)}\right)^* c_{kj}^{(k-1)}.$$

Таким образом, словесное описание „путешествия“ из v_i в v_j по путям ранга, не большего k, приводит к следующей формуле для вычисления элемента матрицы $C^{(k)}$:

$$c_{ij}^{(k)} = c_{ij}^{(k-1)} + c_{ik}^{(k-1)}\left(c_{kk}^{(k-1)}\right)^* c_{kj}^{(k-1)}. \tag{5.5}$$

Пусть a_{ij} — элементы *матрицы меток дуг* ориентированного графа. Поскольку каждый путь ранга 0 между несовпа-

дающими вершинами состоит из одной дуги, а каждая вершина достижима сама из себя по пути нулевой длины с меткой $\mathbf{1}$ или по петле с меткой a_{ii}, то элементы матрицы $C^{(0)}$ имеют вид

$$c_{ij}^{(0)} = \begin{cases} a_{ij}, & i \neq j; \\ \mathbf{1} + a_{ii}, & i = j. \end{cases} \tag{5.6}$$

Тогда матрицу стоимостей $C = A^*$ можно найти, вычисляя последовательно матрицы $C^{(k)}$, $k = \overline{0, n}$, по формулам (5.5) и (5.6).

Вычисления по формулам 5.5 и 5.6 образуют ***алгоритм Флойда — Уоршелла — Клини*** определения стоимости прохождения между любыми парами вершин.

Для полуколец $\mathcal{B}$ и $\mathcal{R}^+$ в силу того, что в них итерация любого элемента x равна единице полукольца, получим упрощенный вариант формулы (5.5):

$$c_{ij}^{(k)} = c_{ij}^{(k-1)} + c_{ik}^{(k-1)} c_{kj}^{(k-1)}. \tag{5.7}$$

Вычисления по формуле (5.7) начинают с матрицы $C^{(0)}$, определяемой соотношением (5.6). Все дальнейшие вычисления удобно также проводить в матричном виде. Для нахождения матрицы $C^{(k)}$ удобно определить сначала матрицу $D^{(k)}$, элементы которой вычисляются по формуле

$$d_{ij}^{(k)} = c_{ik}^{(k-1)} c_{kj}^{(k-1)}.$$

Чтобы найти j-й столбец матрицы $D^{(k)}$, достаточно k-й столбец матрицы $C^{(k-1)}$ умножить (в смысле соответствующего полукольца) на j-й элемент k-й строки этой же матрицы.

Решим описанным способом задачу о кратчайших расстояниях в графе, изображенном на рис. 5.27. Для него $C^{(0)} = A$, где матрица A имеет вид (5.4). Используя формулу (5.7), по-

следовательно находим

$$C^{(1)} = \begin{pmatrix} 0 & 5 & 10 & 1 \\ \infty & 0 & 3 & \infty \\ \infty & 1 & 0 & \infty \\ 3 & 8 & 4 & 0 \end{pmatrix}, \qquad C^{(2)} = \begin{pmatrix} 0 & 5 & 8 & 1 \\ \infty & 0 & 3 & \infty \\ \infty & 1 & 0 & \infty \\ 3 & 8 & 4 & 0 \end{pmatrix},$$

$$C^{(3)} = \begin{pmatrix} 0 & 5 & 8 & 1 \\ \infty & 0 & 3 & \infty \\ \infty & 1 & 0 & \infty \\ 3 & 5 & 4 & 0 \end{pmatrix}, \qquad C^{(4)} = \begin{pmatrix} 0 & 5 & 5 & 1 \\ \infty & 0 & 3 & \infty \\ \infty & 1 & 0 & \infty \\ 3 & 5 & 4 & 0 \end{pmatrix}.$$

Например, матрица $C^{(2)}$ по матрице $C^{(1)}$ вычисляется так. Сначала выделим в $C^{(1)}$ вторую строку и второй столбец:

$$C^{(1)} = \left(\begin{array}{c|c|cc} 0 & 5 & 10 & 1 \\ \hline \infty & 0 & 3 & \infty \\ \hline \infty & 1 & 0 & \infty \\ 3 & 8 & 4 & 0 \end{array}\right).$$

Затем, чтобы вычислить первый столбец матрицы $C^{(2)}$, берем второй (выделенный) столбец матрицы $C^{(1)}$ и умножаем (в полукольце $\mathcal{R}^+$) его элементы по очереди на первый элемент второй (выделенной) строки той же матрицы $C^{(1)}$. Каждое такое произведение складываем в полукольце с одноименным элементом первого столбца матрицы $C^{(1)}$. Поскольку умножение в полукольце $\mathcal{R}^+$ — это арифметическое сложение, а сложение — взятие наименьшего из двух чисел, мы получим следующие элементы первого столбца матрицы $C^{(2)}$:

$$c_{11}^{(2)} = \min\{c_{11}^{(1)}, c_{12}^{(1)} + c_{21}^{(1)}\} = \min\{0, \infty\} = 0,$$

$$c_{21}^{(2)} = \min\{c_{21}^{(1)}, c_{22}^{(1)} + c_{21}^{(1)}\} = \min\{\infty, \infty\} = \infty,$$

$$c_{31}^{(2)} = \min\{c_{31}^{(1)}, c_{32}^{(1)} + c_{21}^{(1)}\} = \min\{\infty, \infty\} = \infty,$$

$$c_{41}^{(2)} = \min\{c_{41}^{(1)}, c_{42}^{(1)} + c_{21}^{(1)}\} = \min\{3, \infty\} = 3.$$

Как видим, первый столбец матрицы $C^{(2)}$ не изменился по сравнению с первым столбцом матрицы $C^{(1)}$. Это означает, что нельзя уменьшить стоимость прохождения в первую вершину из других вершин графа, идя через вторую вершину (см. рис. 5.27).

Точно так же для вычисления второго столбца матрицы $C^{(2)}$ умножаем второй столбец $C^{(1)}$ на второй элемент второй строки той же матрицы, для вычисления третьего столбца — на третий элемент второй строки и т.д. Не выписывая подробно всех вычислений, отметим характерный момент — изменение элемента $c_{13}^{(2)} = 8$ по сравнению с $c_{13}^{(1)} = 10$, связанное с тем, что стоимость прохождения из v_1 в v_3 по пути ранга 2 оказалась меньше, чем стоимость прохождения по пути ранга 1. Минимальная же стоимость прохождения $c_{13}^{(4)} = 5$ получена по пути ранга 4.

5.7. Изоморфизм графов

Для *ориентированного графа* (V, E) множество E *дуг* можно рассматривать как *график бинарного отношения непосредственной достижимости*, заданного на множестве *вершин*.

В *неориентированном графе* (V, E) множество E *ребер* является множеством неупорядоченных пар. Для каждой неупорядоченной пары $\{u, v\} \in E$ можно считать, что вершины u и v связаны *симметричным бинарным отношением* ρ, т.е. $(u, v) \in \rho$ и $(v, u) \in \rho$.

Таким образом, с каждым неориентированым и ориентированным графом связано бинарное отношение ρ. Это отношение будем называть ***отношением смежности***.

С алгебраической точки зрения как ориентированный, так и неориентированный граф можно рассматривать как *модель*, *сигнатура* которой состоит из одного *бинарного отношения*: $\mathcal{G} = (V, \rho)$. В классической теории графов изучаются преимущественно конечные модели такого рода.

При указанном подходе различия между ориентированным и неориентированным графами проявляется только в свойствах отношения смежности ρ. Если это отношение *симметрично*, то граф называют неориентированным, и с этой точки зрения неориентированный граф можно считать частным случаем ориентированного*.

Применяя к данному частному случаю моделей общие понятия *гомоморфизма* и *изоморфизма* (см. **4.4**), мы можем сформулировать следующие определения.

Определение 5.14. Отображение $h\colon V_1 \to V_2$ множества вершин графа $G_1 = (V_1, \rho_1)$ в множество вершин графа $G_2 = (V_2, \rho_2)$ называют ***гомоморфизмом графов*** (графа G_1 в граф G_2), если для любых двух вершин, *смежных* в первом графе, их образы при отображении h смежны во втором графе, т.е. если

$$(\forall u, v \in V_1)(u\,\rho_1\,v \Rightarrow h(u)\,\rho_2\,h(v)).$$

Биективный гомоморфизм h, такой, что любые две вершины смежны в первом графе тогда и только тогда, когда их образы смежны во втором графе, т.е.

$$(\forall u, v \in V_1)(u\,\rho_1\,v \Leftrightarrow h(u)\,\rho_2\,h(v)),$$

называют ***изоморфизмом графов*** G_1 и G_2 (графа G_1 на граф G_2), а графы G_1 и G_2 — ***изоморфными***, что записывают в виде $G_1 \cong G_2$.

Гомоморфизм графов, который является *эпиморфизмом*, называется также гомоморфизмом одного графа на другой.

Возвращаясь к нашему определению графа посредством двух множеств: множества вершин V и множества ребер (дуг) E, получим следующие варианты определений гомоморфизма и изоморфизма.

*При таком подходе в неориентированном графе могут быть *петли*, если отношение ρ содержит некоторые элементы *диагонали*, в частности является *рефлексивным*.

Гомоморфизм неориентированного графа $G_1 = (V_1, E_1)$ в неориентированный граф $G_2 = (V_2, E_2)$ есть такое отображение h: $V_1 \to V_2$, что для любых двух вершин первого графа, соединенных ребром, их образы при отображении h также соединены ребром, т.е.

$$(\forall u, v \in V_1)(\{u, v\} \in E_1 \Rightarrow \{h(u), h(v)\} \in E_2).$$

Гомоморфизм ориентированного графа $G_1 = (V_1, E_1)$ в ориентированный граф $G_2 = (V_2, E_2)$ есть такое отображение h: $V_1 \to V_2$, что для любых двух вершин u, v первого графа, таких, что есть дуга, ведущая из u в v, из вершины $h(u)$ тоже ведет дуга в $h(v)$, т.е.

$$(\forall u, v \in V_1)((u, v) \in E_1 \Rightarrow (h(u), h(v)) \in E_2).$$

Изоморфизм неориентированного графа G_1 на неориентированный граф G_2 есть такая *биекция* h: $V_1 \to V_2$, при которой две вершины u и v графа G_1 соединены ребром тогда и только тогда, когда соединены ребром их образы $h(u)$ и $h(v)$, т.е.

$$(\forall u, v \in V_1)(u \mapsto v \Leftrightarrow h(u) \mapsto h(v)).$$

Аналогично изоморфизм ориентированного графа G_1 на ориентированный граф G_2 есть такая биекция h: $V_1 \to V_2$, при которой в ориентированном графе G_1 из вершины u ведет дуга в вершину v тогда и только тогда, когда в ориентированном графе G_2 из образа $h(u)$ вершины u ведет дуга в образ $h(v)$ вершины v, т.е.

$$(\forall u, v \in V_1)(u \to v \Leftrightarrow h(u) \to h(v)).$$

Пример 5.12. **а.** На рис. 5.29 отображение h, где $h(v_1) = w_1$, $h(v_2) = w_2$, $h(v_3) = h(v_4) = w_3$, есть гомоморфизм ориентированного графа G_1 в ориентированный граф G_2.

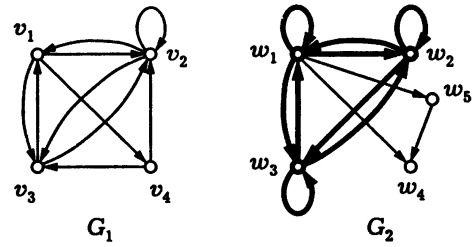


Рис. 5.29

Обратим внимание на петлю (w_3, w_3): эта петля является образом дуги (v_4, v_3), так как $h(v_4) = w_3$ и $h(v_3) = w_3$. В противоположность этому петля (w_1, w_1) не имеет прообраза в G_1.

На этом же рисунке более толстыми стрелками выделен *подграф* G_2' графа G_2, *порожденный подмножеством вершин* $\{w_1, w_2, w_3\}$. Этот подграф будет гомоморфным образом графа G_1. Удаляя петлю в вершине w_1, получим (для того же подмножества вершин) *порожденный подграф*, являющийся *строгим гомоморфным образом* графа G_1. Отметим, что каждая дуга в строгом гомоморфном образе ориентированного графа G_1 имеет прообраз в G_1.

б. На рис. 5.30 неориентированный граф G_2 есть строгий гомоморфный образ графа G_1 (если рассматривать неориентированные графы без петель).

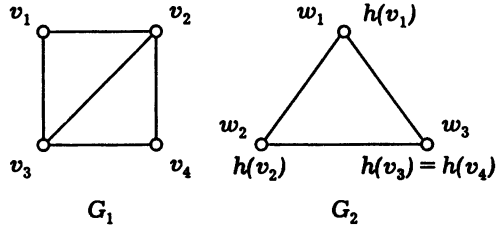
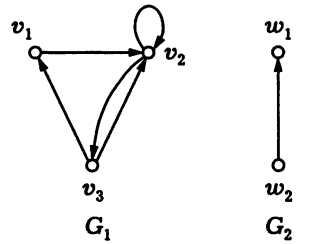


Рис. 5.30

в. На рис. 5.31 отображение h не является гомоморфизмом G_1 на G_2, поскольку $v_1 \to v_2$, но $h(v_1) \not\to h(v_2)$ (петли (w_1, w_1)

нет); $h(v_2) \nrightarrow h(v_3)$, хотя $v_2 \to v_3$ и т.д. Легко сообразить, что любой двухвершинный гомоморфный образ графа G_1 на рис. 5.31 должен иметь петлю, и, следовательно, G_2 не является гомоморфным образом G_1 ни при каком отображении.



$h(v_1) = h(v_2) = w_1, \quad h(v_3) = w_2$

Рис. 5.31

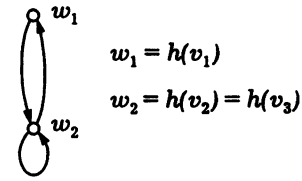


Рис. 5.32

г. На рис. 5.32 изображен один граф из множества двухвершинных гомоморфных образов G_1.

д. Графы, изображенные на рис. 5.33, изоморфны, и изоморфизмом является, например, отображение h, такое, что $h(v_1) = w_1$, $h(v_2) = w_4$, $h(v_3) = w_2$, $h(v_4) = w_5$, $h(v_5) = w_3$, $h(v_6) = w_6$. #

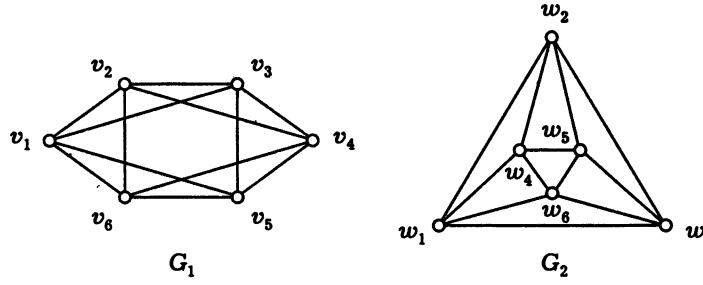


Рис. 5.33

Зачастую для того, чтобы распознать изоморфизм графов, удобно перейти от исходных графов к их дополнениям.

Определение 5.15. Неориентированный (ориентированный) граф $\overline{G} = V; \overline{E}$ называют ***дополнением неориентированного (ориентированного) графа*** $G = (V, E)$, где $\overline{E}$ — *дополнение множества* E до множества всех *неупорядоченных (упорядоченных пар)* на V.

Можно доказать, что графы изоморфны тогда и только тогда, когда изоморфны их дополнения. Например, перейдем к дополнениям графов, изображенных на рис. 5.33. Анализ указанных дополнений (рис. 5.34) позволяет сразу обнаружить их изоморфизм, а следовательно, и изоморфизм исходных графов.

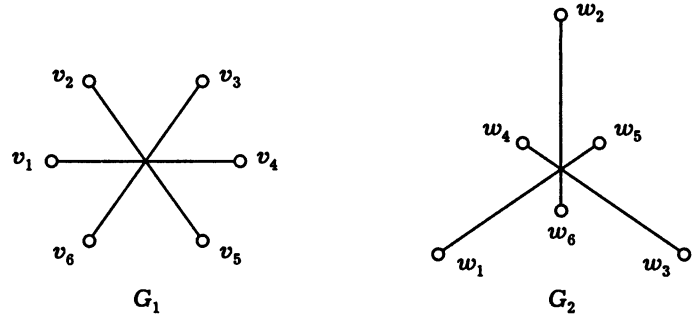


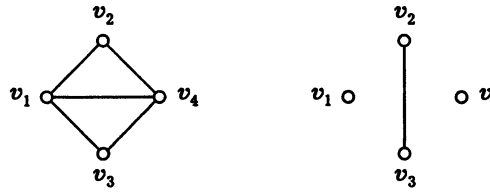
Рис. 5.34

Более сложные случаи распознавания изоморфизма (главным образом, для неориентированных графов) будут рассмотрены в **5.9**.

Определение 5.16. ***Автоморфизм графа*** $G = (V, \rho)$ — это любая *подстановка* множества его вершин, являющаяся изоморфизмом G на себя.

Можно показать, что ***композиция*** двух любых автоморфизмов графа снова есть автоморфизм этого графа, а подстановка, *обратная* к автоморфизму, опять-таки есть автоморфизм. Таким образом, множество всех автоморфизмов данного графа образует *группу* по операции композиции, которая является *подгруппой симметрической группы множества* вершин графа. Ее называют ***группой автоморфизмов*** данного графа.

Рассмотрим некоторые примеры групп автоморфизмов.



а *б*

Рис. 5.35

Для графа, изображенного на рис. 5.35, *а*, группа автоморфизмов порождается *транспозициями* (1 4) и (2 3), что легко усматривается из анализа дополнения этого графа (рис. 5.35, *б*). Очевидно, что группа автоморфизмов графа есть подгруппа группы перестановок множества его вершин. Поэтому, согласно теореме 2.13 Лагранжа, *порядок группы* автоморфизмов графа есть делитель факториала числа его вершин. В частности, для графа на рис. 5.35, *а* группа автоморфизмов имеет порядок 4 и состоит из тождественной подстановки ε и подстановок (1 4), (2 3), (1 4)(2 3). Очевидно, что 4! делится на 4.

Можно доказать, что автоморфизмы неориентированного графа сохраняют *степени*, а ориентированого графа — *полустепени исхода* и *захода* всех его вершин. Это свойство позволяет упростить описание автоморфизмов графа.

Пример 5.13. Найдем нетривиальные (т.е. отличные от тождественной подстановки) автоморфизмы неориентированного графа, изображенного на рис. 5.36. Так как среди вершин графа лишь одна вершина v_1 имеет степень 1 и лишь одна вершина v_4 имеет степень 2, то при любом изоморфизме эти вершины перейдут в себя. Значит, при изоморфизме возможны лишь перестановки вершин v_2, v_3 и v_5.

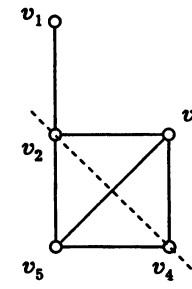


Рис. 5.36

Итак, для любого автоморфизма h этого графа $h(v_1) = v_1$. Поскольку $v_1 \mapsto v_2$, то $h(v_1) \mapsto h(v_2)$, и, следовательно, поскольку v_2 — единственная вершина, смежная с v_1, всегда $h(v_2) = v_2$, т.е. вершина v_2 переходит в себя. Таким образом, единственным нетривиальным автоморфизмом данного графа будет транспозиция (3 5) — „отражение“ квадрата $v_2v_3v_4v_5$ относительно диагонали v_2v_4. #

Иногда группу автоморфизмов графа легко найти именно из чисто геометрических соображений при удачном изображении графа. Автоморфизм есть не что иное, как преобразование графа (геометрической фигуры), при котором граф совмещается с самим собой. Поэтому группу автоморфизмов графа можно изучать, анализируя его как геометрическую фигуру.

Пример 5.14. Для графа, изображенного на рис. 5.37, из геометрических соображений вытекает, что автоморфизм φ сводится к повороту графа на 60° по часовой стрелке вокруг центра тяжести треугольника $v_1v_2v_3$.

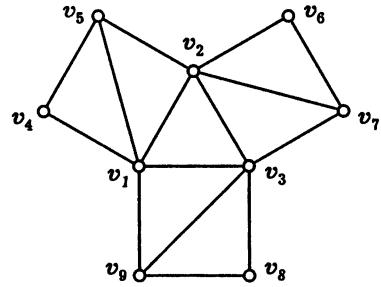


Рис. 5.37

Следовательно, группа автоморфизмов порождается подстановкой φ, являющейся композицией трех *циклов*:

$$\varphi = (1\ 2\ 3)(4\ 6\ 8)(5\ 7\ 9).$$

Заметим, что ни один цикл, входящий в φ, сам по себе не будет автоморфизмом данного графа. Так, если мы рассмотрим цикл $h = (1\ 2\ 3)$, то $h(3) = 1$, $h(4) = 4$, и $h(3) \mapsto h(4)$, но $v_3 \not\mapsto v_4$. #

В заключение сформулируем без доказательства теорему, доказанную Фрухтом в 1938 г. и характеризующую все конечные группы в терминах групп автоморфизмов конечных неориентированных графов.

Теорема 5.5 (теорема Фрухта). Каждая конечная группа изоморфна группе автоморфизмов конечного неориентированного графа.

5.8. Топологическая сортировка

Определение 5.17. ***Ориентированной сетью*** (или просто ***сетью***) называют *бесконтурный ориентированный граф* *.

Поскольку сеть является бесконтурным графом, можно показать, что существуют *вершины* (*узлы*) сети с нулевой *полустепенью исхода*, а также вершины (узлы) с нулевой *полустепенью захода*. Первые называют ***стоками*** или ***выходами сети***, а вторые — ***источниками*** или ***входами сети***.

Определение 5.18. ***Уровень вершины*** сети — это натуральное число, определяемое следующим образом:

1) если полустепень захода вершины равна 0, то ее уровень равен 0 и наоборот (т.е. нулевой уровень N_0 — это множество всех входов);

2) если множества N_i вершин уровня i определены для всех $i \leqslant k$, то уровень N_{k+1} содержит те и только те вершины, *предшественники* которых принадлежат любому из уровней с номером от 0 до k, причем существует хотя бы один предшественник уровня k, т.е.

$$N_{k+1} = \left\{v\colon \Gamma^{-1}(v) \subseteq N_1 \cup \ldots \cup N_k,\ \Gamma^{-1}(v) \cap N_k \neq \varnothing\right\},$$

*В некоторых задачах встречаются бесконтурные ориентированные графы, имеющие бесконечные множества вершин и дуг. Такие графы называют бесконечными сетями. Мы рассматриваем только конечные сети. Кроме того, в литературе по теории графов термин „сеть“ иногда используется в ином смысле — для объекта, более сложного, чем граф (см.: *Яблонский С.В.*).

где $\Gamma^{-1}(v) = \{x\colon x \to v\}$ — множество предшественников вершины v.

Уровень вершины сети можно интерпретировать как *длину* максимального *пути* от входов сети до этой вершины.

Определение 5.19. ***Порядковой функцией*** сети $G = (V, E)$ называют отображение $\mathrm{ord}\colon V \to \mathbb{N}$, сопоставляющее каждой вершине сети номер ее уровня.

Во многих прикладных задачах возникает проблема такого упорядочения вершин сети, при котором вершины, принадлежащие одному уровню, располагаются друг под другом, а дуги ориентированного графа ведут в его изображении на плоскости от вершин с меньшим уровнем к вершинам с большим уровням слева направо. Подобного рода проблема называется проблемой ***топологической сортировки***, поскольку необходимо расположить вершины графа на плоскости так, чтобы отчетливо было видно распределение вершин по уровням. Само расположение при этом может быть разным, лишь бы оно имело „слоистую" структуру, в которой каждый слой составляют вершины одного уровня. На рис. 5.38 показаны сеть и результат применения топологической сортировки сети.

Топологическая сортировка применяется в самых разных ситуациях, например при распараллеливании алгоритмов, когда по некоторому описанию алгоритма нужно составить граф зависимостей его операций и, отсортировав его топологически, определить, какие из операций являются независимыми и могут выполняться параллельно (одновременно).

Формально топологическую сортировку можно реализовать по-разному. Мы опишем один из возможных методов*.

Этот метод состоит в вычислении порядковой функции сети и известен как ***алгоритм Демукрона***. Предполагается, что вершины сети пронумерованы от 1 до n.

*Другие методы топологической сортировки см.: *Кнут Д.*; *Вирт Н.*

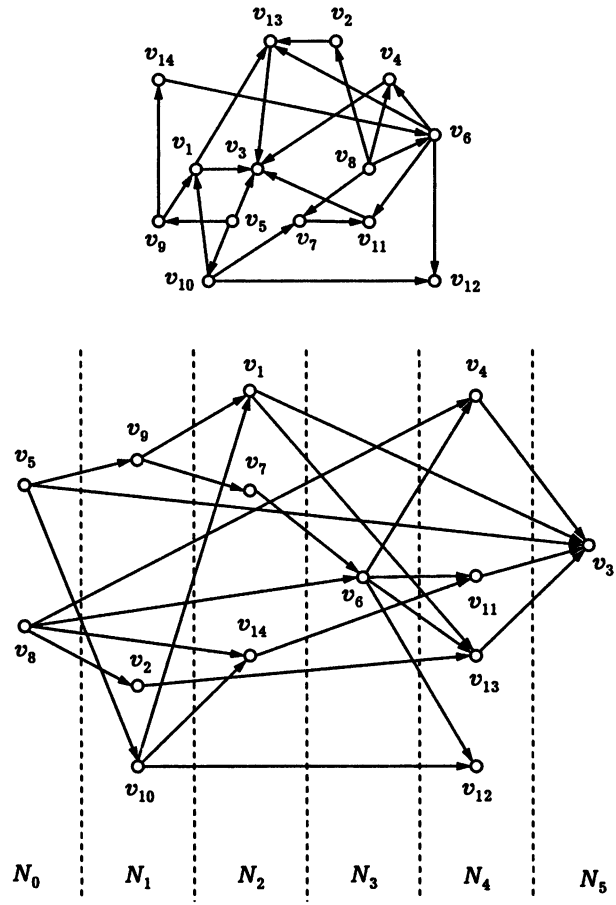


Рис. 5.38

Наглядно процесс определения уровней вершин можно представить следующим образом. Нулевой уровень образуют входы сети — вершины с *полустепенью захода*, равной 0. Удалив из сети все вершины нулевого уровня и исходящие из них дуги, вновь получим сеть, входами которой будут вершины первого уровня исходной сети. Указанный процесс „послойного“ удале-

ния вершин следует продолжать до тех пор, пока все вершины исходной сети не будут распределены по уровням.

Алгоритм Демукрона использует *матрицу смежности вершин* B типа $n \times n$ в качестве средства представления сети и основан непосредственно на определении уровня вершины и свойствах матрицы B. Можно увидеть, что сумма элементов k-го столбца матрицы B равна полустепени захода вершины v_k. Поэтому, просуммировав элементы матрицы по всем столбцам и выбрав вершины, соответствующие столбцам с нулевой суммой, получим множество вершин нулевого уровня — входы сети.

Безусловно, „физически“ удалять вершины и дуги из сети и вычеркивать из матрицы B строки, соответствующие удаляемым вершинам, нет необходимости. Процесс вычисления порядковой функции можно организовать следующим образом. Запишем суммы элементов столбцов матрицы B в вектор M длины n. При этом элемент m_k вектора M будет содержать полустепень захода вершины v_k. Пусть из сети удалены вершина v_i и все исходящие из нее дуги. Заметим, что элемент b_{ik} равен 1, если из вершины v_i идет дуга в вершину v_k, и равен 0 в противном случае. Поэтому, чтобы получить новую полустепень захода вершины v_k, необходимо из элемента m_k вектора M вычесть элемент b_{ik} матрицы B. Чтобы пересчитать полустепени захода всех вершин сети, оставшихся в ней после удаления вершины v_i, надо из вектора M вычесть i-ю строку матрицы B.

Если на очередном шаге входами сети являются вершины $v_{i_1}, \ldots, v_{i_r}$, то для определения следующего „слоя“ вершин нужно из вектора M вычесть строки матрицы B с номерами $i_1, \ldots, i_r$ и зафиксировать новые нулевые элементы вектора M, появившиеся после вычитания. Фиксировать следует именно новые нулевые элементы, поскольку элементы вектора M, соответствующие вершинам, лежащим на предыдущих уровнях, стали равными 0 на предыдущих шагах алгоритма.

Заметим, что порядковую функцию сети можно задать, указав множества вершин, принадлежащих каждому уровню, или сопоставив каждой вершине ее номер уровня. Первый способ более удобен при теоретических рассуждениях, второй — при вычислениях.

Алгоритм Демукрона вычисления порядковой функции сети

Алгоритм обрабатывает матрицу B смежности вершин графа порядка n. В результате работы алгоритма получаем массив Ord длины n, i-й элемент которого равен номеру уровня вершины v_i.

0. Сформировать множество V_1 вершин сети. Значение счетчика уровней k положить равным 0. Найти суммы элементов по всем столбцам матрицы B (полустепени захода вершин) и заполнить ими массив M.

1. Если множество V_1 не пусто, перейти на шаг **2**, если иначе, то перейти на шаг **3**.

2. Определить множество I номеров всех новых нулевых элементов массива M, т.е. таких, что соответствующие этим номерам вершины принадлежат множеству V_1.

Присвоить элементам массива Ord с номерами из множества I номер уровня k и удалить вершины с этими номерами из множества V_1 („замаскировать“ вершины). Вычесть из массива M строки матрицы B, соответствующие вершинам с номерами из множества I (т.е. вершинам последнего вычисленного уровня).

Увеличить счетчик уровней на 1 ($k = k + 1$). Вернуться на шаг **1**.

3. Закончить работу.

Пример 5.15. Применим алгоритм Демукрона к сети, представленной на рис. 5.38. Матрица смежности вершин сети

имеет следующий вид:

	1	2	3	4	5	6	7	8	9	10	11	12	13	14
1			1										1	
2													1	
3														
4			1											
5			1						1	1				
6				1							1	1	1	
7											1			
8		1		1		1	1							
9	1													1
10	1						1					1		
11			1											
12														
13			1											
14						1								

Приведем последовательность значений массива M, соответствующую итерациям алгоритма и множества N_i вершин i-го уровня. Прочерки соответствуют вершинам, не принадлежащим множеству V_1 („замаскированные“ вершины) на соответствующем этапе алгоритма.

1	2	3	4	5	6	7	8	9	10	11	12	13	14	
2	1	5	2	0	2	2	0	1	1	2	2	3	1	$N_0 = \{5, 8\}$
2	0	4	1	–	1	1	–	0	0	2	2	3	1	$N_1 = \{2, 9, 10\}$
0	–	4	1	–	1	0	–	–	–	2	1	2	0	$N_2 = \{1, 7, 14\}$
–	–	3	1	–	0	–	–	–	–	1	1	1	–	$N_3 = \{6\}$
–	–	3	0	–	–	–	–	–	–	0	0	0	–	$N_4 = \{4, 11, 12, 13\}$
–	–	0	–	–	–	–	–	–	–	–	–	–	–	$N_5 = \{3\}$

#

Алгоритм Демукрона может быть модифицирован так, чтобы он останавливался, если ориентированный граф, поданный на вход, не является сетью, и сообщал об этом. Можно увидеть,

что анализируемый граф не будет сетью тогда и только тогда, когда при очередном перевычислении массива M не появятся новые нули.

В заключение рассмотрим вопрос о связи понятий ориентированной сети и *упорядоченного множества.*

Так как сеть есть бесконтурный граф, то *отношение достижимости* в нем будет *отношением порядка.* Действительно, пусть для двух отличных друг от друга вершин u, v сети вершина u достижима из вершины v ($v \Rightarrow^* u$) и вершина v достижима из вершины u ($u \Rightarrow^* v$). Тогда существует путь ненулевой длины из v в u ($v \Rightarrow^+ u$) и из u в v ($u \Rightarrow^+ v$). Отсюда вытекает, что существует путь ненулевой длины из u в u ($u \Rightarrow^+ u$) и из v в v ($v \Rightarrow^+ v$). Следовательно, существует *контур*, связывающий вершины u и v, что невозможно.

Обратно, пусть $(A, \leqslant)$ — конечное упорядоченное множество. Сопоставим ему ориентированный граф $G = (V, E)$ так, что множество вершин V находится с A во взаимно однозначном соответствии, а множество дуг определяется следующим образом: $u \to v$ тогда и только тогда, когда v *доминирует* над u в смысле порядка $\leqslant$.

Построенный таким образом ориентированный граф G будет сетью. Действительно, предположим, что в нем существует контур, содержащий некоторую вершину u. Тогда $u \Rightarrow^+ u$, так как контур есть путь ненулевой длины. Если $u \Rightarrow^1 u$, то существует дуга (петля), ведущая из u в u, т.е. $u \to u$. Но такой петли в графе G быть не может, так как ни один элемент не может доминировать над самим собой (отношение доминирования иррефлексивно).

Если же $u \Rightarrow^n u$, где $n > 1$, то существует вершина v, отличная от u, такая, что $u \Rightarrow^+ v$ и $v \Rightarrow^+ u$, откуда, согласно построению графа G, следует, что $u < v$ и $v < u$, но это невозможно. Итак, G — бесконтурный ориентированный граф, т.е. сеть.

Входы сети G есть *минимальные элементы* исходного упорядоченного множества, а выходы — *максимальные.* Кроме

того, каждый уровень сети образует *антицепь* в упорядоченном множестве $(A, \leqslant)$.

Построенная сеть G будет обладать еще одним интересным свойством: для любых трех попарно различных вершин u, v и w из того, что $u \to v$ и $v \to w$, следует $u \nrightarrow w$. Иначе говоря, в сети G отсутствуют все „замыкающие дуги", подобные дуге $v_1 \to v_3$ для сети, изображенной на рис. 5.39. Такие сети будем называть ***простыми***.

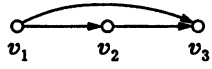


Рис. 5.39

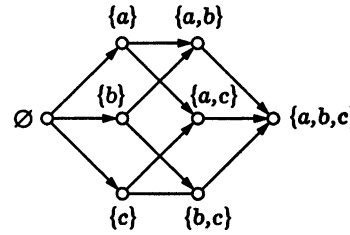


Рис. 5.40

Итак, каждому упорядоченному множеству может быть сопоставлена простая сеть согласно описанной выше процедуре. Эта простая сеть может рассматриваться как вариант наглядного изображения конечного упорядоченного множества и использоваться как *диаграмма Хассе*. На рис. 5.40 показана сеть, сопоставленная множеству всех подмножеств трехэлементного множества $\{a, b, c\}$, упорядоченного отношением включения. Полезно сравнить эту сеть с диаграммой Хассе, приведенной на рис. 1.13.

5.9. Элементы цикломатики

При обсуждении алгоритма *поиска в глубину* в *неориентированном графе* рассматривался вопрос о поиске так называемых *фундаментальных циклов* графа. При этом под фундаментальным понимался цикл, содержащий в точности одно *обратное ребро*, и между фундаментальными циклами и обратными

ребрами устанавливалось взаимно однозначное соответствие. Фундаментальные циклы возникают всякий раз, как только фиксировано произвольное разбиение всех *ребер* неориентированного графа на *древесные* (формирующие некоторый максимальный *остовный лес* исходного графа) и обратные, причем в общем случае это разбиение может быть задано совершенно независимо от алгоритма *поиска в глубину*. Поиск в глубину есть лишь один из способов реализации такого разбиения.

Теперь рассмотрим вопрос о фундаментальных циклах неориентированного графа с алгебраической точки зрения и покажем, что фундаментальные циклы являются *базисом* некоторого *линейного пространства*, которое содержит все циклы графа, а алгоритм поиска в глубину вычисляет один из базисов этого линейного пространства*. Алгебраическое изучение циклов неориентированного графа важно, в частности, при исследовании проблемы распознавания *изоморфизма* двух *графов*.

Пусть c — некоторая *цепь* графа G. Через $E(c)$ обозначим множество *ребер* этой цепи. Для любой цепи c множество ее ребер $E(c)$ определяется однозначно. Обратно, если фиксировано какое-то (конечное) множество ребер $\{e_1, \ldots, e_n\}$ графа G, то можно доказать следующий факт: если указанное выше множество ребер формирует цепь, т.е. существует такая цепь (последовательность вершин графа) v_0, v_1, ..., v_n, что $\{v_0, v_1\} = e_{i_1}$, ..., $\{v_{n-1}, v_n\} = e_{i_n}$, где $1 \leqslant i_1, \ldots, i_n \leqslant n$, то эта цепь при условии, что она не является циклом, определяется с точностью до порядка расположения ребер в последовательности (от v_0 к v_n или наоборот). Если цепь является циклом, то она определяется с точностью до порядка расположения ребер в последовательности и до выбора начальной вершины v_0. Так, на рис. 5.41 множество ребер $\{\{v_1, v_2\}, \{v_2, v_3\}, \{v_3, v_5\}, \{v_5, v_6\}\}$ формирует цепь v_1, v_2, v_3, v_5, v_6, а также цепь v_6, v_5, v_3, v_2, v_1 („инверсия“ предыдущей цепи). Множество ре-

*Отметим, что не любой базис может быть вычислен посредством поиска в глубину (см. граф на рис. 5.23).

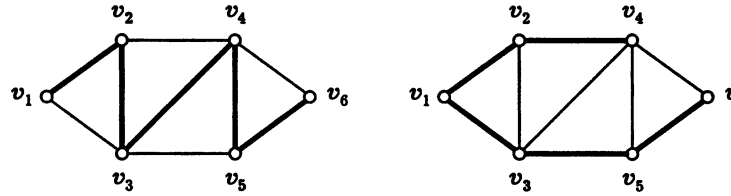


Рис. 5.41

бер $\{\{v_1, v_2\}, \{v_2, v_3\}, \{v_3, v_1\}\}$ формирует цикл v_1, v_2, v_3, v_1, а вместе с ним циклы:

v_2, v_3, v_1, v_2; v_3, v_1, v_2, v_3; v_1, v_3, v_2, v_1;

v_3, v_2, v_1, v_3; v_2, v_1, v_3, v_2.

Множество же ребер $\{\{v_1, v_2\}, \{v_2, v_3\}, \{v_3, v_5\}, \{v_4, v_6\}\}$ не формирует никакой цепи.

Тогда, говоря неформально, если не различать цепи, состоящие из одних и тех же ребер, но проходимые в противоположных направлениях, и циклы, состоящие из одних и тех же ребер, но проходимые в противоположных направлениях и от различных начальных вершин, можно отождествить цепи (и циклы) с множествами их ребер. Эти соображения позволяют ввести операцию сложения цепей как операцию над множествами их ребер.

Для двух произвольных цепей a, b графа G их сумму определим (с учетом сделанного выше замечания) так:

$$a \oplus b = E(a) \triangle E(b),$$

т.е. сложение цепей сводится к вычислению *симметрической разности* множеств их ребер.

Сумма цепей, вообще говоря, не является цепью, т.е. соответствующее множество ребер в общем случае не формирует цепь. На рис. 5.41 сумма цепей v_1, v_3, v_4, v_6 и v_2, v_3, v_4, v_5, v_6 не есть цепь. *Алгебра, системой образующих* которой служит множество всех конечных цепей данного графа G, назы-

вают ***алгеброй цепей неориентированного графа*** G. Эта алгебра является *абелевой группой* в силу известных свойств операции симметрической разности множеств (см. **1** и **2**).

Любая абелева группа $G = (M, \oplus, \mathbf{0})$ может быть рассмотрена как *линейное пространство над полем* $\mathbb{Z}_2$ (*полем вычетов по модулю* 2). Действительно, определим умножение элемента g группы на элементы поля $\mathbb{Z}_2$: $0 \cdot g = \mathbf{0}$ и $1 \cdot g = g$. Выполнение аксиом линейного пространства в этом случае можно легко проверить. Обратим внимание, что в любой *линейной комбинации* $\sum\limits_{k=1}^{n} \lambda_k \cdot e_k$ элементов $e_1, \ldots, e_k$ такого линейного пространства каждый коэффициент λ_k равен либо 0, либо 1. Тогда понятно, что нетривиальная линейная комбинация элементов $e_1, \ldots, e_n$, т.е. линейная комбинация, не все коэффициенты которой равны 0, есть не что иное, как сумма элементов непустого подмножества множества $\{e_1, \ldots, e_n\}$.

Применив указанное построение к алгебре цепей неориентированного графа, получим линейное пространство цепей данного графа. Его называют ***пространством цепей графа***. Заметим, что в этом пространстве каждый элемент противоположен себе самому, так как для каждого элемента a этого пространства $a \oplus a = \varnothing$ (в силу известных свойств операции симметрической разности).

Рассмотрим теперь *линейное подпространство* этого линейного пространства, *порожденное множеством* всех конечных циклов данного графа. Это линейное пространство будем называть ***пространством циклов*** данного ***графа***.

Пространство циклов неориентированного графа есть тем самым *линейная оболочка* в пространстве цепей данного графа множества всех его конечных циклов (причем, как видно на рис. 5.42, сумма циклов в общем случае не есть цикл). Если граф конечен, то его пространство циклов также конечно.

Теорема 5.6. Множество фундаментальных циклов, вычисляемое алгоритмом *поиска в глубину* в неориентированном графе G, есть базис пространства циклов этого графа.

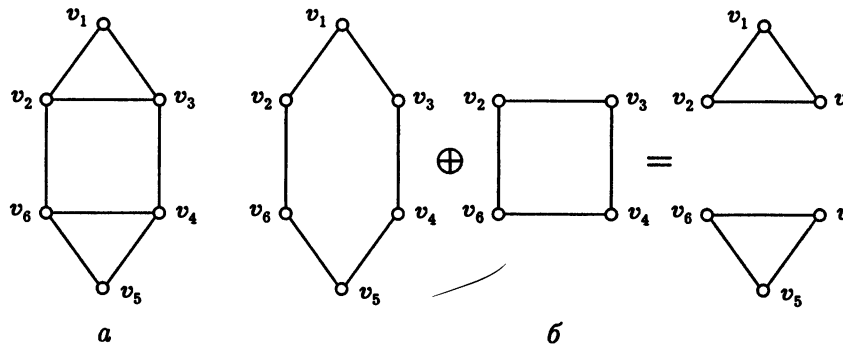


Рис. 5.42

◀ Как следует из описания алгоритма поиска в глубину, каждый вычисляемый им фундаментальный цикл содержит в точности одно обратное ребро и каждое обратное ребро принадлежит в точности одному фундаментальному циклу. Таким образом, каждый фундаментальный цикл содержит ребро, не принадлежащее остальным циклам, и, следовательно, сумма элементов любого непустого подмножества множества фундаментальных циклов отлична от нуля, т.е. фундаментальные циклы, вычисляемые алгоритмом поиска в глубину, являются *линейно независимыми.*

Теперь докажем, что любой цикл графа G есть линейная комбинация фундаментальных циклов.

Докажем сначала, что в пространстве циклов сумма любых двух отличных друг от друга циклов есть либо *замкнутая цепь* (в частности, цикл), либо непустое множество попарно не пересекающихся (т.е. не имеющих один с другим ни общих ребер, ни общих вершин) циклов*.

Пусть C_1 и C_2 — два цикла графа G. Если они не имеют ни общих вершин, ни общих ребер, то доказываемое утверждение

*Это выражение есть, разумеется, вольность речи. Строго говоря, сумму циклов нельзя назвать множеством циклов, но ей в данном случае можно однозначно сопоставить такое множество. Так, на рис. 5.42 сумме циклов соответствуют два непересекающихся между собой цикла.

тривиально. Предположим, что ребра

$$e_1 = \{a_1, b_1\}, \quad \ldots, \quad e_n = \{a_n, b_n\} \quad —$$

все ребра, общие для циклов C_1 и C_2 (рис. 5.43). В каждом из рассматриваемых циклов для каждого $i = \overline{1, n-1}$ существуют цепи, соединяющие вершины b_i и a_{i+1}. Обозначим их W_i^1 для первого и W_i^2 для второго цикла. Также существуют цепи, в первом цикле — W_0^1 и во втором — W_0^2, соединяющие вершины a_1 и b_n. Все эти цепи таковы, что не содержат ни одного из ребер e_i, $i = \overline{1, n}$. Поскольку ребра e_i, $i = \overline{1, n}$, — все ребра, общие для циклов C_1 и C_2, то для каждого $i = \overline{0, n-1}$ цепи W_i^1 и W_i^2 не имеют общих ребер и имеют две общие вершины (см. рис. 5.43). Это значит, что каждая сумма $W_i^1 \oplus W_i^2$, $i = \overline{0, n-1}$, есть цикл (в пространстве цепей графа G). Обозначим его $\varkappa_i$.

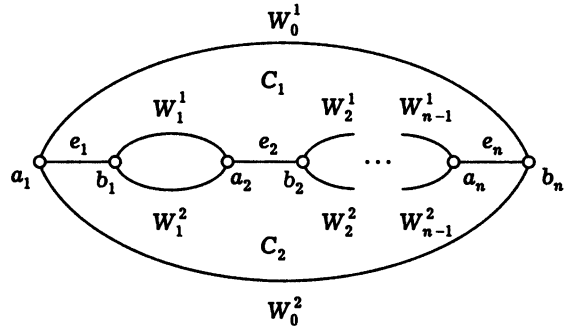


Рис. 5.43

Докажем, что циклы $\varkappa_i$ попарно не пересекаются (т.е. не имеют попарно ни общих вершин, ни общих ребер). Действительно, цепи W_i^1 и W_j^2 при $i \neq j$ не могут иметь ни общих вершин, ни общих ребер, так как вершины a_1, b_1, a_2, b_2, ..., a_n, b_n — это, по построению, все вершины, общие для циклов C_1 и C_2. В то же время (для фиксированного $s \in \{1, 2\}$) никакие две цепи W_i^s и W_j^s при $i \neq j$ не могут иметь общих вершин и

ребер, так как тогда циклы C_1 или C_2 не были бы простыми цепями.

Итак, циклы $\varkappa_i$ попарно не пересекаются. Тогда

$$C_1 \oplus C_2 = \varkappa_0 \oplus \varkappa_1 \oplus \ldots \oplus \varkappa_{n-1},$$

причем в том и только в том случае, когда множество ребер $\{e_1, \ldots, e_n\}$ формирует цепь, написанная выше сумма есть цикл.

Аналогично рассматривается случай, когда циклы C_1 и C_2, не имея общих ребер, имеют общие вершины: a_1, a_2, ..., a_n (рис. 5.44). Тогда, как можно видеть на рис. 5.44, сумма $C_1 \oplus C_2$ будет замкнутой цепью (которая в общем случае не будет циклом).

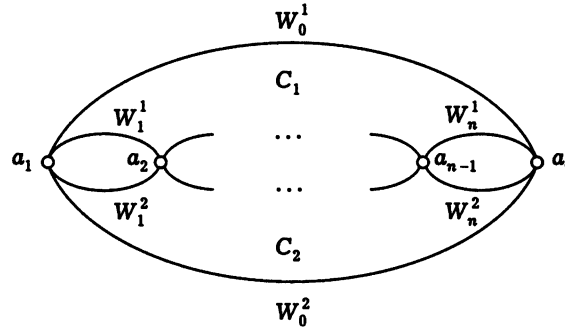


Рис. 5.44

Пусть C — произвольный цикл графа, а f_1, ..., f_m — все его обратные ребра, и пусть F_1, ..., F_m — фундаментальные циклы, содержащие ребра f_1, ..., f_m соответственно.

Рассмотрим суммы (в пространстве циклов):

$$C \oplus F_1 = C_1,$$
$$C_1 \oplus F_2 = C \oplus F_1 \oplus F_2 = C_2,$$
$$\ldots\ldots\ldots\ldots\ldots\ldots$$
$$C_{m-2} \oplus F_{m-1} = C \oplus F_1 \oplus \cdots \oplus F_{m-1} = C_{m-1}.$$

Как было доказано выше, каждая из этих сумм есть либо замкнутая цепь, либо множество попарно не пересекающихся циклов, причем сумма C_i, $1 \leqslant i \leqslant n-1$, содержит обратные ребра $f_{i+1}, \ldots, f_m$, и только их, так как при сложении C с F_1 исчезает общее для них ребро f_1, при сложении C_1 с F_2 — общее для них ребро f_2 и т.д. Следовательно, последняя из этих сумм содержит единственное обратное ребро f_m. Значит, сумма C_{m-1} есть цикл. В самом деле, если бы она была непустым множеством попарно не пересекающихся циклов, то содержала бы по крайней мере два таких цикла. Поскольку в C_{m-1} только одно обратное ребро, то по крайней мере один из этих циклов проходил бы только по древесным ребрам, что невозможно.

Аналогично, предполагая, что C_{m-1} есть замкнутая цепь, не являющаяся циклом, получим (с учетом следствия 5.1), что такая цепь представима в виде суммы по крайней мере двух циклов, не имеющих общих ребер. Тогда один из этих циклов содержит только древесные ребра, что невозможно.

Итак, сумма C_{m-1} есть цикл, содержащий только одно обратное ребро. В силу взаимно однозначного соответствия между множеством обратных ребер и множеством фундаментальных циклов цикл C_{m-1} совпадает с фундаментальным циклом F_m:

$$C_{m-1} = F_m,$$

или

$$C \oplus F_1 \oplus F_2 \oplus \cdots \oplus F_{m-1} = F_m,$$

откуда (поскольку каждый элемент пространства циклов противоположен сам себе)

$$C = F_1 \oplus F_2 \oplus \cdots \oplus F_m.$$

Таким образом, произвольный цикл C представляется в виде некоторой линейной комбинации фундаментальных циклов. ▶

Из доказанной теоремы следует, что число фундаментальных циклов, находимое алгоритмом поиска в глубину, равно

размерности линейного пространства циклов и не зависит от выбора начальной вершины, поскольку все базисы конечномерного линейного пространства состоят из одного и того же числа векторов [IV]. Это число называют ***циклическим рангом*** неориентированного графа. Разность числа всех ребер графа и его циклического ранга называется ***коциклическим рангом*** неориентированного графа. Таким образом, цикличеcкий ранг неориентированного графа — это число обратных ребер при поиске в глубину из любой вершины, а коцикличеcкий ранг — число древесных ребер при поиске в глубину из той же вершины.

Выведем формулу, связывающую циклический ранг ν, число ребер m, число вершин n и число *компонент связности* k произвольного неориентированного графа G.

Обозначим через μ коциклический ранг графа G. Тогда, поскольку в *неориентированном дереве* число ребер на единицу меньше числа вершин, получим

$$\nu = m - \mu = m - ((n_1 - 1) + \ldots + (n_k - 1)) = m - n + k,$$

где n_i — число вершин в i-й компоненте связности графа G.

Итак,

$$\nu = m - n + k.$$

Эта формула выражает циклический ранг неориентированного графа через характеристики графа, не зависящие от способа поиска в глубину, и тем самым еще раз подтверждает инвариантность циклического ранга относительно выбора начальной вершины для поиска. Выбирая различные начальные вершины в алгоритме поиска в глубину, мы получаем различные базисы пространства циклов.

Циклический ранг неориентированного графа показывает „степень цикличности“ этого графа: чем больше циклический ранг, тем больше в графе циклов. *Лес*, в частности дерево, имеет нулевой циклический ранг (нуль–мерное пространство циклов).

Пример 5.16. Рассмотрим неориентированные графы, изображенные на рис. 5.41. Предположим, что в списках смежности вершины упорядочены по возрастанию номеров. На рис. 5.41 показаны результаты поиска в глубину из двух разных вершин: v_1 в графе G_1 и v_4 в графе G_2. Древесные ребра в графах выделены. По результатам поиска циклический ранг равен 4, а коциклический ранг — 5. В графе G_1 фундаментальными являются циклы*

$$C_1\colon\ v_6 \mapsto v_5 \mapsto v_4 \mapsto v_6,$$
$$C_2\colon\ v_5 \mapsto v_3 \mapsto v_4 \mapsto v_5,$$
$$C_3\colon\ v_4 \mapsto v_3 \mapsto v_2 \mapsto v_4,$$
$$C_4\colon\ v_3 \mapsto v_1 \mapsto v_2 \mapsto v_3.$$

Циклы пронумерованы в порядке обнаружения.

Для графа G_2 фундаментальными будут циклы

$$D_1\colon\ v_3 \mapsto v_2 \mapsto v_1 \mapsto v_3,$$
$$D_2\colon\ v_3 \mapsto v_4 \mapsto v_2 \mapsto v_1 \mapsto v_3,$$
$$D_3\colon\ v_5 \mapsto v_4 \mapsto v_2 \mapsto v_1 \mapsto v_3 \mapsto v_5,$$
$$D_4\colon\ v_6 \mapsto v_4 \mapsto v_2 \mapsto v_1 \mapsto v_3 \mapsto v_5 \mapsto v_6.$$

Разложение одного и того же цикла C

$$v_2 \mapsto v_3 \mapsto v_5 \mapsto v_6 \mapsto v_4 \mapsto v_2$$

по двум указанным базисам будет следующим (рис. 5.45):

$$C = C_1 \oplus C_2 \oplus C_3, \qquad C = D_1 \oplus D_4. \quad \#$$

Можно доказать, что *изоморфные* неориентированные графы имеют *изоморфные* линейные пространства циклов, однако

*Записывая циклы (которые, по определению, есть последовательности вершин), мы ради наглядности пишем между вершинами значок $\mapsto$ непосредственной достижимости.

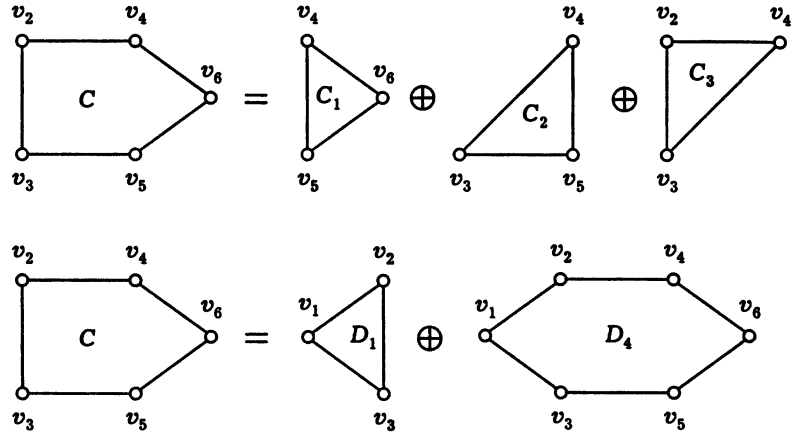


Рис. 5.45

обратное в общем случае неверно. Более того, в изоморфных графах должны совпадать „тонкие" структуры циклов: любое утверждение о циклах, истинное в одном графе, останется истинным в изоморфном ему графе.

Таким образом, анализ циклов может быть использован для доказательства неизоморфности графов.

Пример 5.17. Рассмотрим графы, представленные на рис. 5.46. Заметим, что у этих графов одинаковое количество вершин и ребер, а также одинаковые степени всех вершин. Поэтому гипотеза об их изоморфности представляется правдоподобной. Однако для доказательства изоморфизма графов необходимо явно указать биекцию множества вершин одного графа на множество вершин второго, при которой сохраняется отношение смежности. Поиск такой биекции весьма трудоемок, так как может потребовать полного перебора всех возможных вариантов. Для доказательства неизоморфности достаточно показать принципиальную невозможность установления требуемой биекции. Используем для этого анализ циклов.

Проведем поиск в глубину в каждом из графов. Если количество обратных ребер окажется различным, неизоморфность

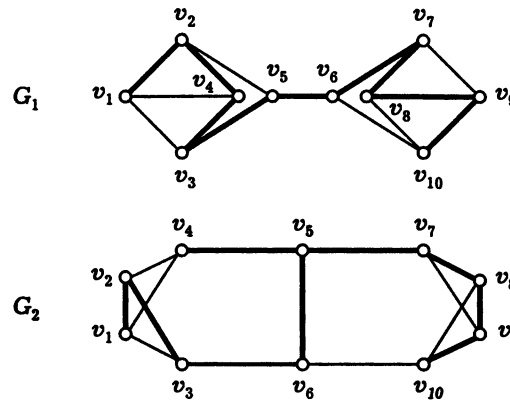


Рис. 5.46

графов будет доказана. Для определенности в каждом из графов начнем поиск с вершины v_1 и будем считать, что списки смежности каждой вершины упорядочены по возрастанию номеров вершин. На рис. 5.46 в графах выделены древесные ребра, полученные при поиске в глубину. В каждом из графов оказалось по шесть обратных ребер, и, следовательно, вывод о неизоморфности сделать нельзя.

Различие можно увидеть в „тонких" стуктурах циклов. Заметим, что в графе G_1 имеются четыре цикла длины 3:

$$v_1 \mapsto v_2 \mapsto v_4 \mapsto v_1, \qquad v_1 \mapsto v_3 \mapsto v_4 \mapsto v_1,$$
$$v_7 \mapsto v_8 \mapsto v_9 \mapsto v_7, \qquad v_8 \mapsto v_9 \mapsto v_{10} \mapsto v_8.$$

В графе G_2 таких циклов также четыре:

$$v_1 \mapsto v_2 \mapsto v_4 \mapsto v_1, \qquad v_1 \mapsto v_2 \mapsto v_3 \mapsto v_1,$$
$$v_7 \mapsto v_8 \mapsto v_9 \mapsto v_7, \qquad v_8 \mapsto v_9 \mapsto v_{10} \mapsto v_8.$$

Однако количество циклов длины 4 в графах различно. В графе G_1 таких циклов шесть:

$$v_1 \mapsto v_2 \mapsto v_4 \mapsto v_3 \mapsto v_1, \qquad v_1 \mapsto v_2 \mapsto v_5 \mapsto v_3 \mapsto v_1,$$
$$v_2 \mapsto v_4 \mapsto v_3 \mapsto v_5 \mapsto v_2, \qquad v_9 \mapsto v_7 \mapsto v_8 \mapsto v_{10} \mapsto v_9,$$
$$v_9 \mapsto v_7 \mapsto v_6 \mapsto v_{10} \mapsto v_9, \qquad v_7 \mapsto v_8 \mapsto v_{10} \mapsto v_6 \mapsto v_7,$$

а в графе G_2 — всего два:

$$v_1 \mapsto v_3 \mapsto v_2 \mapsto v_4 \mapsto v_1, \qquad v_8 \mapsto v_{10} \mapsto v_9 \mapsto v_7 \mapsto v_8.$$

Следовательно, графы неизоморфны. #

Итак, исследование структуры циклов неориентированного графа как инварианта, сохраняющегося при изоморфизмах, позволяет в некоторых случаях быстро находить ответ на вопрос об изоморфности двух заданных графов.

Заметим, что в общем случае все циклы графа можно получить, рассматривая различные линейные комбинации над $\mathbb{Z}_2$ фундаментальных циклов. При этом следует учесть, что результатом сложения может быть множество ребер, образующих несколько непересекающихся циклов.

Вопросы и задачи

5.1. Сколько существует неориентированных графов с n вершинами?

5.2. Доказать, что сумма степеней всех вершин неориентированного графа равна удвоенному числу ребер (это утверждение известно под названием „леммы о рукопожатиях“).

5.3. Доказать, что если число ребер неориентированного графа с n вершинами при $n > 2$ больше C^2_{n-1}, то он связный.

5.4. Доказать, что не существует неориентированного графа, степени всех вершин которого попарно различны.

5.5. Найти все попарно неизоморфные неориентированные графы с четырьмя вершинами и тремя ребрами.

5.6. Установить, какие из изображенных на рис. 5.47 графов изоморфны, а какие — нет.

5.7. Привести пример неориентированного графа с четырьмя вершинами, изоморфного своему дополнению (самодополнительного). Доказать, что число вершин любого самодополнительного неориентированного графа равно $4k$ или $4k+1$.

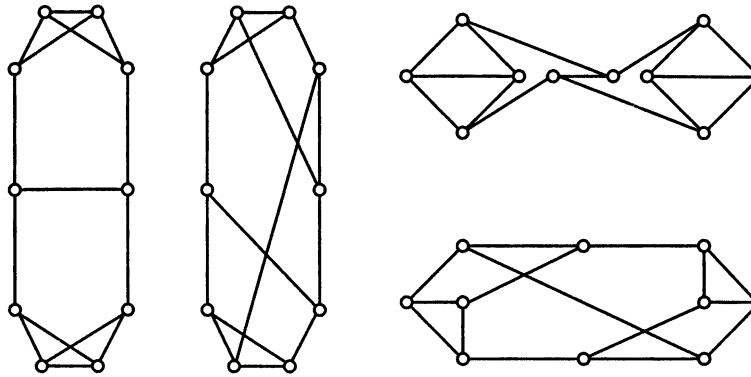
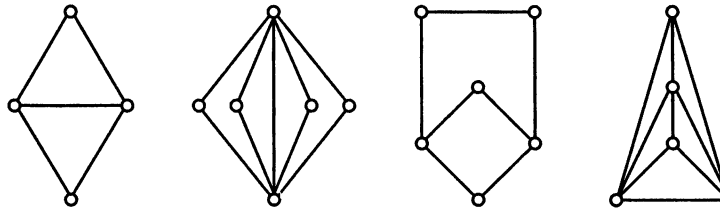


Рис. 5.47

5.8. Найти группы автоморфизмов графов, изображенных на рис. 5.48.



а *б* *в* *г*

Рис. 5.48

5.9. Пусть симметрическая матрица A порядка n, в каждой строке которой находится одинаковое нечетное число t ненулевых элементов, является матрицей смежности неориентированного графа. Показать, что число вершин графа n четное. (Учесть, что $a_{ii} = 0$.)

5.10. Доказать, что отношение взаимной достижимости в ориентированном графе есть эквивалентность.

5.11. Установить, является ли ориентированный граф, изображенный на рис. 5.49, связным. Найти все его компоненты и бикомпоненты.

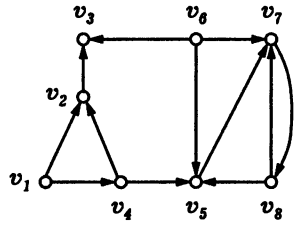


Рис. 5.49

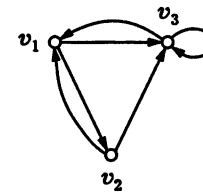


Рис. 5.50

5.12. Конденсация ориентированного графа $G = (V, E)$ есть ориентированный граф $G' = (V/\rho, E')$, где ρ — отношение взаимной достижимости (см. задачу 5.10), а для любых u', $v' \in V'$, $u' \neq v'$,

$$(u', v') \in E' \Leftrightarrow (\exists u \in u'), (\exists v \in v'), (u, v) \in E.$$

Доказать, что G — бесконтурный граф тогда и только тогда, когда G изоморфен своей конденсации G'.

5.13. Найти все ориентированные графы с двумя вершинами, являющиеся гомоморфными образами ориентированного графа, изображенного на рис. 5.50.

5.14. Установить, является ли конденсация ориентированного графа (см. задачу 5.12) его гомоморфным образом. Построить примеры.

Указание: выяснить, может ли конденсация ориентированного графа содержать петли.

5.15. Доказать, что ориентированный граф связный тогда и только тогда, когда в нем есть путь, проходящий через все вершины. Останется ли это утверждение справедливым, если потребовать, чтобы существовал простой путь?

5.16. Пусть $f\colon V \to V$ — биекция множества вершин ориентированного графа $G = (V, E)$ в себя. Сопоставим ей матрицу S с элементами

$$S_{ij} = \begin{cases} 1, & j = f(i); \\ 0, & \text{иначе.} \end{cases}$$

Доказать:

а) $S^{-1} = S^{\mathrm{T}}$ (т.е. S — ортогональная матрица);

б) если f — автоморфизм графа G, то матрица смежности B' автоморфного образа $f(G)$ есть $B' = S^{\mathrm{T}}BS$.

5.17. Доказать, что если неориентированный граф k-регулярный, т.е. степени всех его вершин одинаковы и равны k, то число k есть собственное число матрицы смежности графа.

Указание: доказать, что вектор $x = (1, 1, \ldots, 1)^{\mathrm{T}}$ размерности $n = |V|$ есть собственный вектор матрицы смежности k-регулярного графа G.

5.18. Написать алгоритм нахождения множества предшественников $\Gamma^{-1}(v)$ для каждой вершины графа по спискам смежности. Оценить сложность алгоритма.

5.19. Что является компонентами связности в ориентированном дереве?

5.20. Найти остовное дерево наименьшего веса для неориентированного графа с десятью вершинами, заданного следующим списком ребер с метками (вершина, вершина, метка):

$$(v_1, v_2, 1),\ (v_2, v_3, 5),\ (v_3, v_4, 4),\ (v_4, v_5, 8),\ (v_5, v_6, 3),$$
$$(v_6, v_8, 5),\ (v_8, v_9, 11),\ (v_9, v_{10}, 8),\ (v_{10}, v_1, 7),$$
$$(v_2, v_{10}, 5),\ (v_2, v_7, 8),\ (v_3, v_7, 7),\ (v_3, v_5, 3),$$
$$(v_4, v_{10}, 4),\ (v_6, v_7, 6),\ (v_8, v_7, 12),\ (v_9, v_7, 9).$$

5.21. Найти условия, необходимые и достаточные для единственности остовного дерева наименьшего веса.

5.22. Доказать, что сеть является простой, если любой подграф, порожденный множеством всех вершин, достижимых из некоторого корня, является деревом. Верно ли обратное?

5.23. Используя алгоритм Демукрона, найти порядковую функцию сети, заданной следующими матрицами смежности:

а)
$$\begin{pmatrix}
0&0&0&0&1&0&0&0&0&0&0\\
1&0&0&0&0&0&0&0&0&0&0\\
1&0&0&0&0&0&0&0&1&0&0\\
0&0&1&0&0&0&1&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0\\
0&1&0&0&0&0&1&1&1&0&0\\
0&1&0&0&0&0&0&1&0&0&0\\
0&0&0&0&0&0&0&0&1&1&0\\
0&0&0&0&0&0&0&0&0&1&0\\
0&0&0&0&0&0&0&0&0&0&0\\
1&0&1&0&0&1&0&0&0&0&0
\end{pmatrix};$$

б)
$$\begin{pmatrix}
0&0&1&0&0&0&0&0\\
1&0&1&1&1&1&1&0\\
0&0&0&1&1&0&0&0\\
0&0&0&0&1&0&1&0\\
0&0&0&0&0&0&1&1\\
1&0&1&0&0&0&0&0\\
0&0&0&0&0&0&0&1\\
0&0&0&0&0&0&0&0
\end{pmatrix}.$$

5.24. Составить подробное описание алгоритма вычисления порядковой функции сети, который в случае, если входной граф есть сеть, работает как алгоритм Демукрона, если же иначе, то сообщает о наличии в графе контуров и останавливается.

5.25. ***Эйлеровым циклом*** в неориентированном графе называется цикл, в котором каждое ребро графа проходится ровно один раз. Доказать, что связный неориентированный граф имеет эйлеров цикл тогда и только тогда, когда степень каждой его вершины четная.

5.26. Доказать, что ориентированный граф не содержит контуров тогда и только тогда, когда при поиске в глубину из некоторой вершины (независимо от ее выбора) множество обратных дуг пусто.

5.27. Доказать, что в связном ориентированном графе найдется вершина, при поиске в глубину из которой глубинный остовный лес является деревом. Верно ли обратное?

Указание: используйте результат задачи 5.15.

5.28. Найти все возможные глубинные остовные леса, получающиеся при поиске из вершины v_1:

а) для неориентированного графа с 10 вершинами и множеством ребер

$$E = \{\{v_1, v_2\},\ \{v_1, v_3\},\ \{v_2, v_3\},\ \{v_2, v_4\},\ \{v_4, v_5\},\ \{v_5, v_6\},\ \{v_6, v_7\},\ \{v_5, v_7\},\ \{v_7, v_8\},\ \{v_8, v_9\},\ \{v_9, v_{10}\},\ \{v_8, v_{10}\}\};$$

б) для ориентированного графа с 10 вершинами и множеством дуг

$$E = \{(v_1, v_2),\ (v_1, v_3),\ (v_3, v_2),\ (v_2, v_4),\ (v_2, v_{10}),\ (v_4, v_5),\ (v_5, v_6),\ (v_7, v_6),\ (v_7, v_5),\ (v_7, v_8),\ (v_8, v_9),\ (v_9, v_{10}),\ (v_{10}, v_8)\}.$$

Указание: рассмотреть различные порядки расположения вершин в списках смежности.

5.29. Выполнить поиск в глубину для неориентированных графов, изображенных на рис. 5.47, из различных стартовых вершин.

5.30. Выполнить поиск в ширину из вершины v_2 для ориентированного графа из задачи 5.28.

5.31. Выполнить поиск в ширину из вершины v_5 для ориентированного графа, изображенного на рис. 5.38. После остановки алгоритма продолжить поиск из вершины v_8. Сравнить стоимости прохождения со значениями порядковой функции сети, приведенными на рисунке.

5.32. Для ориентированных графов, заданных множеством дуг с указанием меток (вершина, вершина, метка), решить задачу транзитивного замыкания (в полукольце B) и задачу вычисления кратчайших расстояний (в полукольце R^+):

а) $(v_1, v_2, 8)$, $(v_1, v_1, 2)$, $(v_1, v_3, 5)$, $(v_2, v_1, 3)$, $(v_3, v_2, 2)$;

б) $(v_1, v_2, 2)$, $(v_1, v_4, 10)$, $(v_2, v_3, 3)$, $(v_3, v_5, 4)$, $(v_5, v_4, 5)$, $(v_2, v_4, 7)$, $(v_4, v_3, 6)$;

в) $(v_1, v_2, 10)$, $(v_1, v_4, 5)$, $(v_2, v_1, 6)$; $(v_2, v_3, 7)$, $(v_2, v_4, 2)$, $(v_2, v_5, 9)$, $(v_3, v_3, 8)$, $(v_3, v_4, 10)$, $(v_4, v_3, 5)$, $(v_4, v_5, 4)$, $(v_4, v_2, 7)$, $(v_5, v_1, 8)$, $(v_5, v_3, 4)$;

г) $(v_1, v_2, 2)$, $(v_1, v_3, 3)$, $(v_2, v_3, 6)$, $(v_3, v_2, 5)$, $(v_2, v_4, 6)$, $(v_3, v_5, 2)$, $(v_4, v_5, 3)$, $(v_5, v_4, 4)$, $(v_6, v_4, 1)$, $(v_7, v_5, 5)$, $(v_6, v_7, 4)$, $(v_7, v_2, 6)$;

д) $(v_1, v_2, 1)$, $(v_2, v_3, 3)$, $(v_3, v_4, 4)$, $(v_5, v_4, 5)$, $(v_5, v_6, 1)$, $(v_6, v_1, 1)$, $(v_1, v_7, 2)$, $(v_2, v_7, 1)$, $(v_4, v_7, 1)$, $(v_7, v_3, 2)$, $(v_7, v_5, 1)$, $(v_7, v_6, 1)$;

5.33. Ориентированный граф задан матрицей смежности, содержащей метки соответствующих ребер:

$$\begin{pmatrix} 0 & 7 & 8 & 9 & 10 \\ 0 & 0 & 8 & 9 & 10 \\ 0 & -2 & 0 & 9 & 10 \\ 0 & -4 & -3 & 0 & 10 \\ 0 & -7 & -6 & -5 & 0 \end{pmatrix}.$$

Решить задачу о кратчайших путях для данного ориентированного графа, обратив внимание на то, что метки некоторых дуг равны 0 или отрицательны. Выяснить, применим ли здесь алгоритм Флойда — Уоршела — Клини.

Можно ли изменить метки дуг таким образом, чтобы алгоритм Флойда — Уоршела — Клини стал неприменимым?

5.34. Установить, можно ли использовать алгоритм Флойда — Уоршела — Клини для вычисления стоимости путей наибольшей длины во взвешенном графе. Сформулировать ограничение на граф, при котором такое использование возможно.

6. БУЛЕВЫ ФУНКЦИИ

6.1. Понятие булевой функции. Булев куб

В дискретной математике большую роль играют конечные функции. ***Конечной функцией*** называют *отображение* одного конечного множества в другое. Важный класс таких функций образуют булевы функции.

Булева функция (от n переменных) — это произвольное отображение вида

$$f\colon \{0,1\}^n \to \{0,1\}, \tag{6.1}$$

т.е. булева функция определена на множестве всех n-элементных (при $n \geqslant 0$) *последовательностей* (или n-компонентных *кортежей*) нулей и единиц и принимает два возможных значения: 0 и 1.

С понятием булевой функции тесно связаны понятия булевой константы и булева переменного*.

Булева константа — это *индивидная константа* с областью значений $\{0,1\}$. Таким образом, существуют две булевы константы: 0 и 1. По определению принимается, что каждая булева константа есть также булева функция от 0 переменных (что вполне аналогично определению *нульарной операции*).

Булево переменное — это *индивидное переменное* с областью значений $\{0,1\}$, т.е. это переменное, которое может принимать только два значения: 0 и 1 (подобно тому, как действительное переменное принимает произвольное действительное

*В литературе по теории булевых функций традиционно употребляется термин „булева переменная“ (в женском роде). Но так как в данном комплексе учебников принят термин „переменное“ (среднего рода), мы пишем везде „булево переменное“.

значение, а комплексное переменное — произвольное комплексное значение). Тогда с использованием понятия булева переменного мы можем задать булеву функцию (6.1) записью $y = f(x_1,\ldots,x_n)$, в которой каждое булево переменное x_i, $i = \overline{1,n}$, и функция f принимают два возможных значения: 0 и 1. Переменные $x_1, \ldots, x_n$ называют при этом ***переменными булевой функции*** f. Фиксируя значение $\alpha_i \in \{0,1\}$ каждого переменного x_i, получаем кортеж $\tilde{\alpha} = (\alpha_1, \ldots, \alpha_n)$ из множества $\{0,1\}^n$, называемый ***набором значений переменных*** $x_1, \ldots, x_n$, и соответствующее ему значение функции $f(\alpha_1,\ldots,\alpha_n)$, которое будет значением переменного y, сопоставленным заданным значениям переменных $x_1, \ldots, x_n$.

Подчеркнем, что, если специально не оговорено противное, в выражении $y = f(x_1,\ldots,x_n)$ все переменные предполагаются попарно различными, т.е. пробегающими каждое независимо от других множество $\{0,1\}$.

Понятию булевой функции можно придать иной смысл, понимая элементы множества $\{0,1\}$ как истинностные значения, а именно понимая единицу как „истину“, а нуль как „ложь“. Такие истинностные значения могут быть, как мы знаем, сопоставлены каждому *высказыванию*. Тогда булева функция есть некоторое правило, позволяющее каждому фиксированному набору истинностных значений, т.е. набору значений булевых переменных, сопоставить то или иное истинностное значение. Так может быть вычислено, например, истинностное значение сложного высказывания, составленного по определенным правилам из простых высказываний. Подобного рода сложные высказывания составляются с помощью *логических операций*: „или“, „и“, „если ..., то ...“ и т.п. Указанная логическая интерпретация булевой функции позволяет понять, почему булево переменное часто называют логическим переменным*, а булеву функцию — логической функцией (или функцией алгебры логики).

*Традиционный термин: „логическая переменная“.

Кроме того, согласно определению, булева функция от n переменных есть отображение n-й декартовой степени множества $\{0,1\}$ в множество $\{0,1\}$, т.е. не что иное, как *n-арная операция* на множестве $\{0,1\}$. Тем самым логическая интерпретация булевой функции согласуется с ее алгебраической интерпретацией: булева функция есть операция (в алгебраическом смысле этого слова) на множестве истинностных значений. Тогда и понятие логической операции, которое было введено ранее (см. **1.1**), оказывается частным случаем общего понятия операции (см. **2.1**).

Будем обозначать через $\mathcal{P}_2$ множество всех булевых функций (для всех возможных значений n числа переменных), а через $\mathcal{P}_{2,n}$ — множество всех булевых функций от n переменных (для фиксированного n).

Из определения следует, что

$$\mathcal{P}_2 = \bigcup_{n \geqslant 0} \mathcal{P}_{2,n}.$$

Итак, областью определения любой булевой функции от n переменных является множество $\{0,1\}^n$, т.е. *булев куб** *размерности n*. Элементы булева куба $\{0,1\}^n$ называют *n-мерными булевыми векторами* (или *наборами*). Число всех элементов булева куба $\{0,1\}^n$ составляет 2^n. Элементы булева куба будем также называть его вершинами.

Булев куб $\{0,1\}^n$ является *носителем булевой алгебры* $\mathbb{B}^n$ (это же обозначение часто будем использовать и для соответствующего булева куба). Но в любой булевой алгебре $\mathcal{A} = (A, \vee, \wedge, \mathbf{0}, \mathbf{1})$ определяется, как известно, естественное *отношение порядка* $\leqslant$ так, что для произвольных $a, b \in A$ соотношение $a \leqslant b$ выполняется тогда и только тогда, когда $a \vee b = b$ (или, что равносильно, $a \wedge b = a$). Напомним (см. **3.4**), что булева алгебра $\mathbb{B}^n$ является не чем иным, как n-й *декартовой степенью двухэлементной булевой алгебры* $\mathbb{B} = (\{0,1\}, \vee, \wedge, 0, 1)$.

*Употребляются также термины: ***единичный куб размерности n***, ***n-мерный единичный куб***; вместо слова „куб“ говорят также „гиперкуб“.

Согласно общему принципу распространения отношения порядка на *декартово произведение множеств* (см. **4.5**), для произвольных двух наборов $\widetilde{\alpha} = (\alpha_1, \ldots, \alpha_n)$ и $\widetilde{\beta} = (\beta_1, \ldots, \beta_n)$ из $\mathbb{B}^n$ имеет место $\widetilde{\alpha} \leqslant \widetilde{\beta}$ тогда и только тогда, когда $\alpha_i \leqslant \beta_i$ для каждого $i = \overline{1, n}$, т.е. $\alpha_i \vee \beta_i = \beta_i$.

Другими словами, $\widetilde{\alpha} \leqslant \widetilde{\beta}$ тогда и только тогда, когда $\alpha_i = \beta_i$ или $\alpha_i = 0$, а $\beta_i = 1$ для каждого $i = \overline{1, n}$. Если существует хотя бы одно i, для которого выполняется $\alpha_i = 0$, $\beta_i = 1$, то имеет место строгое неравенство $\widetilde{\alpha} < \widetilde{\beta}$. В частности, если существует ровно одно такое i, то набор $\widetilde{\beta}$ *доминирует* над набором $\widetilde{\alpha}$, так как ясно, что в этом случае нельзя найти такой набор $\widetilde{\gamma}$, что $\widetilde{\alpha} < \widetilde{\gamma} < \widetilde{\beta}$.

Пример 6.1. В булевом кубе $\mathbb{B}^4$

$$(0, 0, 1, 1) < (1, 0, 1, 1) < (1, 1, 1, 1),$$

причем второй из этих наборов доминирует над первым, а третий — над вторым (но, естественно, третий уже не доминирует над первым, а лишь строго больше его). Наборы же (0, 1, 0, 1) и (1, 0, 1, 1) — *несравнимые элементы*, так как первая компонента второго набора больше первой компоненты первого набора, но зато вторая компонента первого набора больше второй компоненты второго набора. Подчеркнем также, что описанное сравнение наборов возможно только для фиксированной размерности и никак нельзя сравнивать наборы разных размерностей. #

Рассмотренное отношение порядка на $\mathbb{B}^n$ будем называть ***булевым порядком***.

Булев куб как *упорядоченное множество*, можно изобразить в виде *диаграммы Хассе*. На рис. 6.1 приведены диаграммы Хассе для булевых кубов размерностей от 0 до 4.

Другой способ наглядного изображения булева куба основан на том, что диаграмма Хассе любого конечного упорядоченного множества A может быть задана в виде *ориентированной сети*, так что *дуга* из *вершины*, сопоставленной элементу $x \in A$,

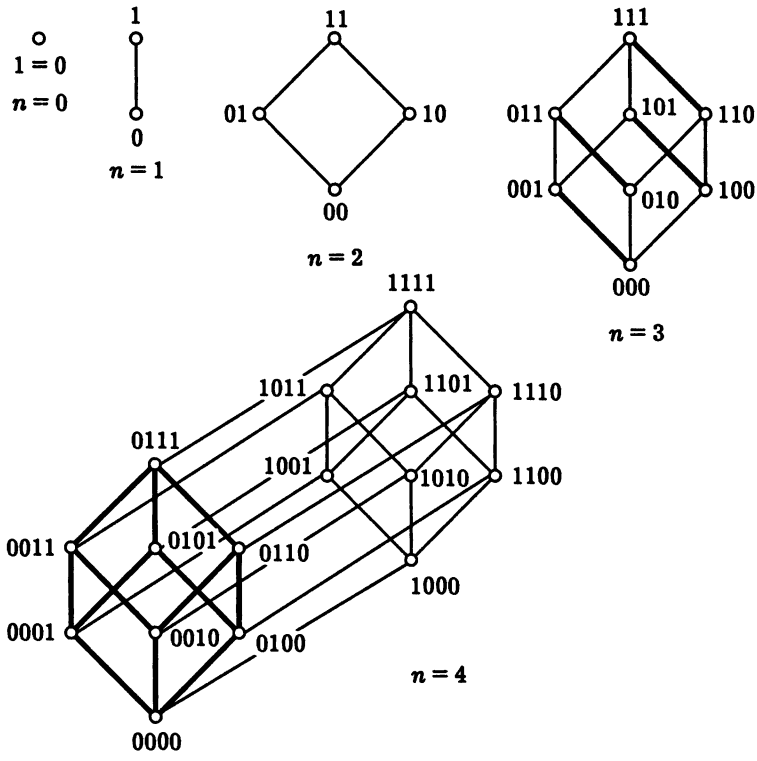


Рис. 6.1

ведет в вершину, сопоставленную элементу $y \in A$, тогда и только тогда, когда y доминирует над x и каждый *уровень сети* состоит из вершин, сопоставленных попарно несравнимым элементам множества A (т.е. элементам некоторой антицепи в A). *Входы сети* сопоставлены ***минимальным***, а *выходы* — ***максимальным элементам*** A.

Каждый уровень представляющей булев куб $\mathbb{B}^n$ сети состоит из всех вершин, соответствующих наборам, у которых ровно k $(0 \leqslant k \leqslant n)$ компонент отличны от нуля (множество всех таких наборов для фиксированного k называют ***k-слоем*** булева куба $\mathbb{B}^n$).

Сеть, служащую изображением булева куба размерности n, будем называть ***булевой n-сетью*** или просто ***булевой сетью***, если упоминание о размерности опускается. Так как булев куб имеет *наименьший элемент — нулевой набор* и *наибольший элемент — единичный набор*, то каждая булева сеть имеет единственный вход (помеченный нулевым набором) и единственный выход (помеченный единичным набором).

На рис. 6.2 приведено изображение булева куба $\mathbb{B}^4$ в виде сети.

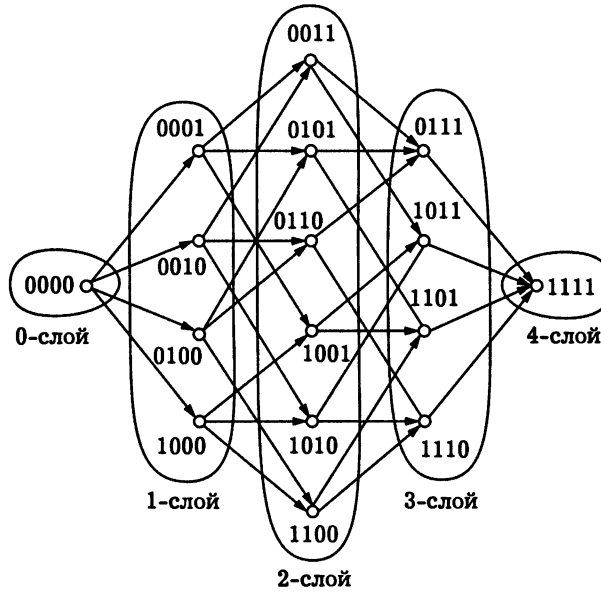


Рис. 6.2

Помимо булева порядка полезно также ввести на булевом кубе другое отношение порядка, которое мы будем называть ***лексикографическим порядком***, используя обозначение $\preceq$.

Пусть $\widetilde{\alpha}, \widetilde{\beta} \in \mathbb{B}^n$ (для произвольного фиксированного n). По определению, $\widetilde{\alpha} \preceq \widetilde{\beta}$, если

$$\sum_{i=1}^{n} \alpha_i \cdot 2^{n-i} \leqslant \sum_{i=1}^{n} \beta_i \cdot 2^{n-i}. \tag{6.2}$$

Каждая из сумм в неравенстве (6.2) есть не что иное, как представление некоторого натурального числа (включая и нуль) в двоичной системе счисления (при числе разрядов, равных фиксированной размерности n). На каждый булев вектор можно смотреть как на такое представление (двоичный код) натурального числа, и лексикографический порядок на булевом кубе $\mathbb{B}^n$ есть не что иное, как *естественный числовой порядок* на подмножестве $\{0, 1, \ldots, 2^{n-1}\}$ множества $\mathbb{N} \cup \{0\}$ (при условии, что числа заданы в двоичной системе счисления)*.

Заметим, что отношение лексикографического порядка является, в отличие от булева порядка, отношением *линейного порядка*.

Пример 6.2. Набор (1, 0, 1) как двоичный код числа $5 = 1 \cdot 2^2 + 0 \cdot 2^1 + 1 \cdot 2^0$ лексикографически больше набора (0, 1, 1), служащего двоичным кодом числа 3, но при этом указанные наборы не сравнимы по отношению булева порядка. #

Однако лексикографический порядок при изучении булевых кубов играет вспомогательную роль. В частности, при изображении булевых кубов (в виде диаграмм Хассе или в виде сети) принято располагать вершины каждого k-слоя в лексикографическом порядке (по возрастанию — слева направо или сверху вниз). Везде в дальнейшем, рассуждая о булевом кубе как об упорядоченном множестве, мы имеем в виду булев порядок.

Гранью булева куба размерности $n - k$, где n — размерность булева куба, называют множество наборов, имеющих не менее k одинаковых компонент.

Грань размерности $n - k$ в $\mathbb{B}^n$ будет определена, если фиксировать k номеров $1 \leqslant i_1 \leqslant \ldots \leqslant i_k \leqslant n$ и k констант $\sigma_1, \ldots, \sigma_k$ из множества $\{0, 1\}$. Тогда грань, обозначаемая как $\mathbb{B}^{n, i_1, \ldots, i_k}_{\sigma_1, \ldots, \sigma_k}$, есть множество всех таких $\alpha \in \mathbb{B}^n$, что $\alpha_{i_1} = \sigma_1, \ldots, \alpha_{i_k} = \sigma_k$. При этом кортеж номеров $(i_1, \ldots, i_k)$ называют ***направлени-***

*Более строго: упорядоченное множество $(\mathbb{B}^n, \preceq)$ *изоморфно* подмножеству $\{0, 1, \ldots, 2^{n-1}\} \subset \mathbb{N} \cup \{0\}$ с естественным числовым порядком.

ем грани $\mathbb{B}^{n,i_1,\ldots,i_k}_{\sigma_1,\ldots,\sigma_k}$. Если число $n-k$, определяющее размерность грани, равно 1 (т.е. $k=n-1$), то такую грань называют ***ребром булева куба*** $\mathbb{B}^n$. Таким образом, ребро — это множество, состоящее из двух наборов, один из которых доминирует над другим (в смысле булева порядка). Другими словами, эти два набора отличаются друг от друга значением единственной компоненты. Для ребра булева куба будем использовать обозначение $[\widetilde{\alpha}, \widetilde{\beta}]$, полагая, что второй набор доминирует над первым. Любая вершина булева куба считается гранью размерности 0.

Можно показать, что число всех граней размерности $n-k$ составляет $2^k \cdot C_n^k$.

Пример 6.3. В четырехмерном булевом кубе $\mathbb{B}^4$ направление трехмерной грани задается фиксированием одного из номеров от 1 до 4 и фиксированием значения соответствующей компоненты наборов, принадлежащих этой грани. На рис. 6.1 выделены ребра грани $\mathbb{B}^{4,1}_0$. Эта грань состоит из восьми наборов:

$$(0,0,0,0),\quad (0,0,0,1),\quad (0,0,1,0),\quad (0,0,1,1),$$
$$(0,1,0,0),\quad (0,1,0,1),\quad (0,1,1,0),\quad (0,1,1,1).$$

На рис. 6.1 выделены все ребра булева куба $\mathbb{B}^3$, имеющие направление (1, 2). #

Грани булева куба, имеющие одно и то же направление, называют ***параллельными***. Две параллельные грани $\mathbb{B}^{n,i_1,\ldots,i_k}_{\sigma_1,\ldots,\sigma_k}$ и $\mathbb{B}^{n,i_1,\ldots,i_k}_{\gamma_1,\ldots,\gamma_k}$ называют ***соседними***, если один из наборов $(\sigma_1, \ldots, \sigma_k)$ и $(\gamma_1, \ldots, \gamma_k)$ доминирует над другим. На рис. 6.1 грани $\mathbb{B}^{4,1}_0$ и $\mathbb{B}^{4,1}_1$ соседние, равно как и грани (ребра) $\mathbb{B}^{3,1,2}_{0,0}$ и $\mathbb{B}^{3,1,2}_{0,1}$. Но ребра $\mathbb{B}^{3,1,2}_{1,0}$ и $\mathbb{B}^{3,1,2}_{0,1}$ не являются соседними.

Нетрудно догадаться, что каждая грань размерности $n-k$ булева куба $\mathbb{B}^n$ сама является булевым кубом размерности $n-k$ и содержит, следовательно, 2^{n-k} вершин. Точнее говоря, эта грань вместе с *отношением порядка, индуцированным* булевым

порядком на $\mathbb{B}^n$, будет упорядоченным множеством, *изоморфным* булеву кубу $\mathbb{B}^{n-k}$ (с булевым порядком на нем). Поэтому часто грани булева куба называют его ***подкубами***.

В то же время булев куб $\mathbb{B}^n$ изоморфно вкладывается (несколькими способами) в булев куб большей размерности, а именно булев куб $\mathbb{B}^n$ вкладывается (как грань) в булев куб $\mathbb{B}^{n+k}$, $k \geqslant 1$, столькими способами, сколько существует разных n-мерных граней в кубе размерности $n+k$, т.е. $2^k \cdot C_{n+k}^k$ способами. Так, одномерный куб $\mathbb{B}$ вкладывается в двумерный $\mathbb{B}^2$ четырьмя способами — как одна из его четырех одномерных граней (т.е. как одно из его четырех ребер, см. рис. 6.1).

Договоримся впредь записывать конкретные наборы (элементы булева куба соответствующей размерности) без скобок и запятых, т.е. будем писать не (0, 1, 0, 1), а 0101 (если, конечно, это не ведет к недоразумениям).

В заключение найдем *мощность множества* всех булевых функций от n переменных для фиксированного n. Поскольку каждая булева функция отображает множество из 2^n элементов в двухэлементное множество, а мощность множества всех отображений из n-элементного множества в m-элементное равна, как известно, m^n, то мощность множества $\mathcal{P}_{2,n}$ равна 2^{2^n}. В частности, при $n = 0$ получаем две булевы константы: 0 и 1.

Замечание 6.1. Поскольку булева функция от n переменных является в то же время и n-арной операцией на множестве $\{0, 1\}$, то при $n = 0$ получаем *нульарную операцию*. Ясно, что на множестве $\{0, 1\}$ существуют две нульарные операции: 0 и 1, которые есть не что иное, как нуль и единица двухэлементной булевой алгебры.

6.2. Таблицы булевых функций

Булева функция от n переменных может быть задана таблицей, состоящей из двух столбцов и 2^n строк. В первом столбце перечисляются все наборы из $\mathbb{B}^n$ в лексикографиче-

ском порядке, а во втором — значения функции на наборах. Форма таблицы произвольной булевой функции приведена ниже (табл. 6.1).

Таблица 6.1

$x_1 \dots x_n$	$f(x_1, \dots, x_n)$
$0 \dots 0$	$f(0, \dots, 0)$
$\dots$	$\dots$
$(\alpha_{k,1} \dots \alpha_{k,n})$	$f(\alpha_{k,1}, \dots, \alpha_{k,n})$
$\dots$	$\dots$
$1 \dots 1$	$f(1, \dots, 1)$

В $(k+1)$-й строке таблицы расположен набор

$$\widetilde{\alpha}_k = (\alpha_{k,1} \dots \alpha_{k,n}),$$

являющийся двоичным кодом числа k (при $0 \leqslant k \leqslant 2^n - 1$).

Рассмотрим некоторые примеры булевых функций, которые будем задавать посредством таблиц.

При $n = 1$ имеем четыре булевы функции (табл. 6.2).

Таблица 6.2

x	$f_1(x)$	$f_2(x)$	$f_3(x)$	$f_4(x)$
0	0	0	1	1
1	1	0	1	0

Функцию f_1 называют ***тождественной функцией***, а функцию f_4 — ***отрицанием***. Функции f_2 и f_3 являются функциями (от одного переменного), принимающими постоянное значение (0 и 1 соответственно). Их также зачастую называют константой 0 и константой 1. Постоянные функции, разумеется, могут быть определены и при любом (большем 1) числе переменных.

В табл. 6.3 указаны семь (из $2^{2^2} = 16$) наиболее важных для дальнейшего изложения булевых функций от двух переменных.

Таблица 6.3

x_1	x_2	f_1	f_2	f_3	f_4	f_5	f_6	f_7
0	0	0	0	0	1	1	1	1
0	1	1	0	1	1	0	1	0
1	0	1	0	1	0	0	1	0
1	1	1	1	0	1	1	0	0

Поскольку каждая булева функция от двух переменных есть одновременно и бинарная операция на множестве $\{0,1\}$, то естественно для таких функций использовать запись, принятую для бинарных операций: $x\omega y$ вместо $\omega(x,y)$.

Для функций, записанных в табл. 6.3, принимаются следующие обозначения:

$$
\begin{aligned}
&f_1(x_1,x_2) = x_1 \vee x_2,\\
&f_2(x_1,x_2) = x_1 \cdot x_2 \quad (\text{или } f_2(x_1,x_2) = x_1 \wedge x_2),\\
&f_3(x_1,x_2) = x_1 \oplus x_2,\\
&f_4(x_1,x_2) = x_1 \to x_2,\\
&f_5(x_1,x_2) = x_1 \sim x_2,\\
&f_6(x_1,x_2) = x_1 \mid x_2,\\
&f_7(x_1,x_2) = x_1 \downarrow x_2.
\end{aligned}
$$

Функцию f_1 называют ***дизъюнкцией***, f_2 — ***конъюнкцией***, f_3 — ***сложением по модулю 2*** (mod 2), f_4 — ***импликацией***; f_5 — ***эквивалентностью***, f_6 — ***штрихом Шеффера***, f_7 — ***стрелкой Пирса***.

Дизъюнкция и конъюнкция, как видно, — это операции *двухэлементной булевой алгебры — объединение* и *пересечение* соответственно (тогда как функция отрицания есть не что иное, как *дополнение* в этой булевой алгебре). В то же время дизъюнкция и конъюнкция — это не что иное, как одноименные *логические операции*, введенные в **1.1** (в свете логической интерпретации булевой функции, см. **6.1**). Очевидным образом с логическими связками, помимо отрицания, соотносятся импликация и эквивалентность (хотя их обозначения как булевых

функций несколько отличаются от введенных в **1.1**). Таблицы для введенных булевых функций являются тогда не чем иным, как формой представления *таблиц истинности*.

Далее, можно заметить, что сложение по модулю 2 совпадает с операцией сложения *кольца вычетов* $\mathbb{Z}_2$ по модулю 2, штрих Шеффера есть отрицание конъюнкции, а стрелка Пирса — отрицание дизъюнкции, т.е.

$$x_1 \mid x_2 = \overline{x_1 \cdot x_2}, \quad x_1 \downarrow x_2 = \overline{x_1 \vee x_2}.$$

Приведем для примера таблицу булевой функции от трех переменных (табл. 6.4).

Таблица 6.4

	x_1	x_2	x_3	$f(x_1, x_2, x_3)$
0	0	0	0	0
1	0	0	1	0
2	0	1	0	0
3	0	1	1	1
4	1	0	0	0
5	1	0	1	1
6	1	1	0	1
7	1	1	1	1

Эта функция называется ***мажоритарной функцией*** или ***функцией голосования***. Заметим, что в первом столбце табл. 6.4 для каждого набора из $\{0,1\}^3$ указан его номер, т.е. число, двоичным кодом которого служит данный набор.

Теоретически таблицей можно задать любую булеву функцию, но при большом числе переменных этот способ практически не применим. Далее (см. **6.4**) мы рассмотрим способ задания булевых функций в виде формул, аналогичный аналитическому заданию элементарных функций в анализе [I]. С точки зрения логической интерпретации булевой функции ее

таблица есть таблица истинности некоторого сложного *высказывания*. Переменными функции являются входящие в сложное высказывание простые высказывания.

Кроме того, полезно иметь в виду, что, записывая таблицу булевой функции от n переменных, нет необходимости каждый раз перечислять все наборы длины n — достаточно записать ***вектор значений булевой функции***, понимая, что i-я компонента этого вектора есть значение функции на i-м наборе (двоичном коде числа i).

Тогда мажоритарная функция f может быть задана так:

$$f = (0,0,0,1,0,1,1,1).$$

Можно также перечислить номера тех наборов, на которых функция принимает значение 1:

$$f = \{3,5,6,7\}.$$

Обобщением понятия булевой функции служит понятие булева оператора. ***Булев оператор*** — это произвольное отображение вида

$$f\colon \mathbb{B}^n \to \mathbb{B}^m \tag{6.3}$$

для произвольных n, $m \in \mathbb{N} \cup \{0\}$.

Булев оператор (6.3) может быть задан посредством семейства булевых функций в виде

$$\begin{cases} y_1 = f_1(x_1,\ldots,x_n), \\ \ldots\ldots\ldots\ldots \\ y_m = f_m(x_1,\ldots,x_n). \end{cases} \tag{6.4}$$

Функции f_i в (6.4) будем называть ***координатными функциями булева оператора*** (6.3). Если ввести векторы переменных $\boldsymbol{y} = (y_1, \ldots, y_m)$ и $\boldsymbol{x} = (x_1, \ldots, x_n)$, то булев оператор (6.3), заданный семейством координатных функций (6.4), можно записать в таком виде:

$$\boldsymbol{y} = f(\boldsymbol{x}). \tag{6.5}$$

6.3. Фиктивные переменные. Равенство булевых функций

Ключевым понятием в теории *булевых функций* является понятие равных булевых функций. Для функций от одного и того же числа переменных n нет необходимости рассматривать какое-то специальное определение равенства, ибо такие функции равны, если они совпадают как *отображения булева куба* $\mathbb{B}^n$ в $\mathbb{B}$. Проблема состоит в том, чтобы определить равенство булевых функций независимо от числа переменных.

Пример 6.4. Рассмотрим булевы функции $f(x,y) = x \vee y$ и $g(x,y,z) = xz \vee x\overline{z} \vee yz \vee y\overline{z}$.

Можно заметить, используя тождества *булевой алгебры*, что

$$g(x,y,z) = (x \vee y)(z \vee \overline{z}),$$

а поскольку $z \vee \overline{z} = 1$, то

$$g(x,y,z) = (x \vee y) = f(x,y),$$

и функции f и g естественно рассматривать как равные, несмотря на то что они зависят от разного числа переменных. #

В примере 6.4 функция g определена как функция от трех переменных, но значение переменного z не влияет на значение функции. Обобщая ситуацию примера, можно ввести понятие фиктивного переменного булевой функции.

Определение 6.1. Переменное x_i называют ***фиктивным переменным*** булевой функции $f(x_1,\ldots,x_n)$, если значение функции не зависит от значения этого переменного, т.е. если для любых значений переменных x_1, ..., x_{i-1}, x_{i+1}, ..., x_n

$$f(x_1,\ldots,x_{i-1},0,x_{i+1},\ldots,x_n) = f(x_1,\ldots,x_{i-1},1,x_{i+1},\ldots,x_n).$$

Будем называть переменное x, не являющееся фиктивным переменным функции f, ***существенным переменным*** данной функции и говорить, что функция f существенно зависит от переменного x.

Пусть дана булева функция $y = f(x_1, \ldots, x_n)$ от n переменных. Пусть существенными переменными этой функции являются переменные $x_{i_1}, \ldots, x_{i_r}$, где $r \leqslant n$ и $1 \leqslant i_1 < \ldots < i_r \leqslant n$. Присваивая каждому фиктивному переменному функции f произвольное значение, получим функцию $\widetilde{f}$, такую, что она есть функция от r переменных, т.е. есть отображение из $\mathbb{B}^r$ в $\mathbb{B}$, и для любого *набора* $\alpha_{i_1}, \ldots, \alpha_{i_r}$ *значений переменных* $x_{i_1}, \ldots, x_{i_r}$ имеет место равенство

$$\widetilde{f}(\alpha_{i_1}, \ldots, \alpha_{i_r}) = f(x_1, \ldots, \alpha_{i_1}, \ldots, \alpha_{i_r}, \ldots, x_n)$$

независимо от значений фиктивных переменных функции f (т.е. переменных, отличных от $x_{i_1}, \ldots, x_{i_r}$). Тем самым функция $\widetilde{f}$ оказывается уже функцией, определенной на некоторой r-мерной *грани булева куба* $\mathbb{B}^n$.

Договоримся только что описанный переход от функции f к функции $\widetilde{f}$, уже не имеющей фиктивных переменных, называть удалением фиктивных переменных (функции f).

Замечание 6.2. В качестве упомянутой выше r-мерной грани может быть взята любая грань с *направлением*, заданным номерами фиктивных переменных. Фиксация грани означает фиксацию конкретного набора значений фиктивных переменных. Тогда соответствующая этой грани функция $\widetilde{f}$ получается из исходной функции f в результате подстановки вместо каждого фиктивного переменного того конкретного значения, которое задано указанным выше набором.

Вернемся к примеру 6.4. Удаляя фиктивное переменное z, вместо функции g получим либо функцию $g(x, y, 0)$, которая определена на (двумерной) грани $\mathbb{B}_0^{3,3}$ булева куба $\mathbb{B}^3$, либо функцию $g(x, y, 1)$, определенную на грани $\mathbb{B}_1^{3,3}$. Направлением каждой из этих граней будет однокомпонентный кортеж (3), определенный номером фиктивного переменного z. #

С использованием понятий фиктивного и существенного переменного можно дать следующее определение равных булевых функций.

Определение 6.2. Булевы функции f и g называют ***равными***, если их существенные переменные соответственно равны и на каждом наборе значений этих переменных функции f и g принимают равные значения.

Чтобы распознать по таблице булевой функции, является ли переменное x_i фиктивным, нужно рассмотреть все наборы с фиксированным значением i-й компоненты (один раз фиксировав это значение как 0, другой раз — как 1). Из определения 6.1 ясно, что переменное x_i фиктивно тогда и только тогда, когда для любых двух наборов, отличающихся только значением i-й компоненты, функция принимает равные значения.

Пример 6.5. а. Из табл. 6.4 следует, что ***мажоритарная функция*** не имеет фиктивных переменных, так как, например, $f(0,0,1)=0$, а $f(1,0,1)=1$, т.е. переменное x_1 существенное. Далее, $f(1,0,0)=0$, но $f(1,1,0)=1$, что означает существенность переменного x_2; для x_3 имеем $f(1,0,0)=0$, но $f(1,0,1)=1$, что означает существенность и этого переменного.

б. Ниже приведена таблица функции g от четырех переменных (табл. 6.5). Рекомендуется проверить, что переменное x_2 является фиктивным и что остальные переменные существенны. Более того, анализ таблицы показывает, что эта функция есть не что иное, как мажоритарная функция, существенно зависящая от переменных x_1, x_3 и x_4. #

Кроме процедуры удаления фиктивных переменных используют и процедуру добавления к множеству переменных булевой функции одной или нескольких фиктивных переменных. Так, если дана функция $f(x_1,\ldots,x_n)\in\mathcal{P}_{2,n}$, то можно ввести новое фиктивное переменное y, определив новую, равную исходной, согласно определению 6.2, функцию от $n+1$ переменного таким образом:

$$\widehat{f}(x_1,\ldots,x_n,y)=f(x_1,\ldots,x_n)\cdot(y\vee\overline{y})$$

(см. пример 6.4).

Таблица 6.5

	x_1	x_2	x_3	x_4	$g(x_1,x_2,x_3,x_4)$
0	0	0	0	0	0
1	0	0	0	1	0
2	0	0	1	0	0
3	0	0	1	1	1
4	0	1	0	0	0
5	0	1	0	1	0
6	0	1	1	0	0
7	0	1	1	1	1
8	1	0	0	0	0
9	1	0	0	1	1
10	1	0	1	0	1
11	1	0	1	1	1
12	1	1	0	0	0
13	1	1	0	1	1
14	1	1	1	0	1
15	1	1	1	1	1

Следует заметить, что фиктивное переменное можно (в новой функции $\widehat{f}$) „поставить на любое место“. Или, говоря точнее, можно определить семейство функций $\widehat{f}_i$, $i = \overline{1,n}$, с фиктивным переменным y так, что

$$\widehat{f}_i(x_1,\ldots,y,x_i,\ldots,x_n) = f(x_1,\ldots,x_n)\cdot(y\vee\overline{y}).$$

Понятие фиктивного переменного позволяет также произвольные две булевы функции рассматривать как функции от одного и того же числа переменных.

Действительно, пусть функция $f(x_1,\ldots,x_n)$ есть функция от n переменных, а функция $g(y_1,\ldots,y_m)$ — функция от m переменных. Обозначим множества переменных функции f и g через X и Y соответственно. Расширим множество переменных функции f до $X\cup Y$, вводя переменные из $Y\setminus X$ (если они

существуют) как фиктивные. Точно так же поступим с функцией g, добавляя фиктивные переменные из множества $X \setminus Y$ (если, конечно, оно не пусто). Тогда получим функции $\widehat{f}$ и $\widehat{g}$, равные, согласно определению 6.2, функциям f и g соответственно и определенные как функции от одного и того же числа переменных, составляющего $|X \cup Y|$.

Нетрудно распространить описанную конструкцию на произвольное конечное множество булевых функций и считать тем самым все функции этого множества функциями от одного и того же числа переменных.

В заключение рассмотрим понятие проектирующей функции.

Функцию pr_i от n переменных, такую, что

$$\mathrm{pr}_i(x_1, \ldots, x_i, \ldots, x_n) = x_i,$$

называют (i-й) ***проектирующей функцией***. В общем случае нумерация множества переменных может быть явно не задана, и тогда при определении проектирующей функции следует указывать не номер соответствующего переменного, а само переменное. В этом случае проектирующая функция pr_x^X с множеством переменных X этой функции и выделенным переменным $x \in X$ определяется так:

$$\mathrm{pr}_x^X(\ldots, x, \ldots) = x$$

(за многоточиями „скрыты“ переменные проектирующей функции, отличные от переменного x).

Из определения следует, что проектирующая функция pr_x^X имеет единственное *существенное переменное*, а именно переменное x. Все остальные переменные проектирующей функции являются *фиктивными*. Поэтому любые две проектирующие функции pr_x^X и pr_x^Y *равны*, согласно определению 6.2, при любых множествах переменных X и Y, содержащих переменное x.

Вместе с тем для двух различных переменных x и y проектирующие функции pr_x^X и pr_y^X — разные функции. Так,

$\mathrm{pr}_1(x_1,x_2)$ — функция, отличная от функции $\mathrm{pr}_2(x_1,x_2)$, поскольку, например, $\mathrm{pr}_1(1,0)=1$, $\mathrm{pr}_2(1,0)=0$.

Впредь договоримся любую из множества равных между собой проектирующих функций pr_x^X обозначать символом x ее единственного существенного переменного.

Замечание 6.3. Такое обозначение проектирующих функций есть условность и определенная вольность, состоящая в том, что проектирующая функция pr_x^X как бы отождествляется с самим переменным x. Однако отождествление функции и переменного недопустимо, так как понятие переменного, хоть и связано с понятием функции, никак не есть частный случай понятия функции. Переменное — только имя, некое обозначение, которое используется при аналитическом задании функции, но никак не сама функция. Тем не менее ради краткости, мы сохраним обозначение x как обозначение любой из проектирующих функций pr_x^X и будем использовать иногда даже выражение „функция x“, понимая под этим любую из указанного семейства проектирующих функций.

6.4. Формулы и суперпозиции

Табличный способ задания *булевой функции* не является эффективным. Им практически нельзя воспользоваться при большом числе переменных. Помимо этого способа существует способ представления булевых функций в виде формул. Этот способ аналогичен аналитическому способу задания функций действительного переменного [I].

Как известно, в математическом анализе мы исходили из определенного множества элементарных функций и строили из них сложные функции, записывая их в виде формул, например: $y=\sin(\mathrm{tg}\,x)$, $y=\ln(\cos x)$, $y=e^{-\cos(x+\mathrm{tg}\,x)}$ и т.п. Аналогично обстоит дело для булевых функций, но только вместо элементарных функций математического анализа мы используем эле-

ментарные булевы функции — главным образом, те функции от одной и от двух переменных, которые мы определили в **6.1**.

Но в отличие от математического анализа в теории булевых функций ставится задача представления любой булевой функции такой формулой, которая содержала бы строго определенное конечное множество элементарных булевых функций. Эти функции назовем пока условно „базисными функциями“. Множества таких базисных функций могут быть разными, но, так или иначе, мы хотим иметь нечто вроде функционального базиса (или множества таких базисов), через элементы которого можно было бы выразить любую булеву функцию. Аналогичная задача не может быть решена для функций действительного переменого. Для булевых же функций задача оказывается разрешимой, и это обусловлено прежде всего тем, что булева функция является *конечной функцией*.

Чтобы математически точно сформулировать и решить поставленную выше задачу, нам необходимо уточнить понятие формулы. В анализе, поскольку там не возникала задача подобного рода, мы могли ограничиться интуитивной идеей формулы как некоего способа представления функции. В теории булевых функций мы хотим доказывать утверждения вида „любая булева функция может быть представлена формулой над заданным множеством базисных функций F“. Но тогда нам нужно дать математическое определение „формулы над заданным множеством базисных функций F“, а также уточнить, что значит „булева функция представлена некоторой формулой“. Кроме того, формулы обретают „самостоятельную жизнь“ еще и потому, что одну и ту же булеву функцию можно представить, вообще говоря, разными формулами (как над одним и тем же базисом, так и над разными базисами). Тогда необходимо иметь механизм эквивалентного преобразования формул, т.е. перехода от заданной формулы, представляющей некоторую булеву функцию, к новой, скажем, более простой формуле, которая представляет ту же самую функцию.

Определение формулы основано на понятии сложной функции, или суперпозиции.

Пусть булева функция f есть функция от n переменных, а булевы функции $g_1, \ldots, g_n$ — произвольные (и не обязательно различные) функции от одного и того же числа переменных, которое обозначим m.

Определим функцию $f(g_1,\ldots,g_n)$, называемую ***суперпозицией функций*** $f, g_1, \ldots, g_n$ так, что для любого $\widetilde{\alpha} \in \mathbb{B}^m$ имеет место равенство

$$f(g_1,\ldots,g_n)(\widetilde{\alpha}) = f(g_1(\widetilde{\alpha}),\ldots,g_n(\widetilde{\alpha})).$$

Таким образом, суперпозиция $f(g_1,\ldots,g_n)$ есть не что иное, как *композиция булевых операторов* $g \circ f$, где булев оператор g задается семейством координатных функций g_i, $i = \overline{1,n}$.

Для суперпозиции $f(g_1,\ldots,g_n)$ используется также обозначение $\mathcal{S}(f,g_1,\ldots,g_n)$. Предположение о том, что все функции g_i, $i = \overline{1,n}$, — функции от одного и того же числа переменных, не ограничивает общности, поскольку, как было показано (см. **6.3**), любое конечное множество булевых функций всегда можно рассматривать как множество функций от одного и того же числа переменных.

Замечание 6.4. Обратим внимание на то, что в общем случае „уравнивание“ числа переменных функций g_i, $i = \overline{1,n}$, связано с добавлением *фиктивных переменных*, а его, как известно, можно осуществлять разными способами. Поэтому суперпозиция $f(g_1,\ldots,g_n)$, вообще говоря, определена однозначно лишь с точностью до равенства булевых функций согласно определению 6.2. #

Пусть дано некоторое множество булевых функций F. Тогда ***формулой над множеством*** F мы считаем любую константу из F (если она там есть) и любое булево переменное. Далее, если известно, что $\Phi_1, \ldots, \Phi_n$ $(n \geqslant 1)$ — формулы над множеством F, а f — функция из F от n переменных, то выра-

жение $f(\Phi_1,\dots,\Phi_n)$ есть формула над множеством F. Никаких других формул над множеством F, кроме определенных выше, не существует.

Замечание 6.5. 1. Строго говоря, в формуле $f(\Phi_1,\dots,\Phi_n)$ фигурирует не сама булева функция из множества F, а ее обозначение, т.е. *индивидная константа* с областью значений $\mathcal{P}_2$. Но мы, чтобы не усложнять терминологию, будем отождествлять обозначения базисных функций, т.е. функций из заданного множества F, с самими базисными функциями.

2. Обычно предполагают, что рассматриваются переменные из некоторого заранее фиксированного (и не более чем счетного) множества переменных X.

Пример 6.6. а. Пусть $F = \{\vee, \cdot, {}^{-}\}$. Это множество, состоящее из функций *дизъюнкции*, *конъюнкции* и *отрицания*, называют ***стандартным базисом***. Формулами над стандартным базисом будет любое переменное: x, y, ..., x_1, ..., x_n и т.д. Далее, из переменных x, y как формул и функции $\vee$ можно построить новую формулу, например $\vee(x,y)$ или $\cdot(x,y)$. Эти формулы, однако, естественно записывать несколько иначе. Поскольку каждая булева функция от двух переменных (каковы, в частности, дизъюнкция и конъюнкция) является одновременно *бинарной операцией* на множестве $\mathbb{B} = \{0,1\}$, то формулы с такими функциями обычно записывают в „инфиксной форме“, т.е. как $(x \vee y)$, $(x \cdot y)$ и т.п. Аналогично для отрицания используют запись $\overline{x}$, а не ${}^{-}(x)$.

Кроме того, в формулах над стандартным базисом, во-первых, опускают скобки, используя *ассоциативность булевых операций* $\vee$ и $\cdot$, т.е. вместо $((x \vee y) \vee z)$ пишут $(x \vee y \vee z)$; во-вторых, опускают, как правило, внешние скобки, записывая формулу, аналогичную последней, просто как $x \vee y \vee z$; в-третьих, используют соглашение о „старшинстве“ (или о приоритете) операций, полагая, что самый высокий приоритет имеет операция отрицания (т.е. она всегда выполняется перед конъюнкцией

и дизъюнкцией), затем идет конъюнкция и после нее — дизъюнкция.

С учетом сказанного формула $(\overline{(x \vee y)} \vee ((y \cdot z) \cdot u))$ может быть переписана так:

$$\overline{(x \vee y)} \vee y \cdot z \cdot u. \tag{6.6}$$

Согласно определению формулы, можно представить процесс построения формулы (6.6) следующим образом. Из переменных x, y и функции $\vee$ строим формулу $\Phi_1 = (x \vee y)$, затем из нее и функции отрицания получаем формулу $\Phi_2 = \overline{\Phi}_1$, т.е. формулу $\overline{(x \vee y)}$. Далее из переменных y, z и функции $\cdot$ строим формулу $(y \cdot z)$, а из нее, переменного u и опять функции $\cdot$ — формулу $\Phi_3 = (y \cdot z) \cdot u$, которую записываем как $y \cdot z \cdot u$. И наконец, из формул Φ_2, Φ_3 и функции $\vee$ строим формулу $\Phi_2 \vee \Phi_3$, т.е. формулу (6.6).

Описанный процесс можно наглядно изобразить в виде *ориентированного дерева* (рис. 6.3). *Листья* этого дерева помечены переменными или константами формулы, а каждый *узел*, не являющийся листом, — одной из функций из множества F (на рисунке эти узлы изображены в виде кружочков, внутри которых указано имя функции).

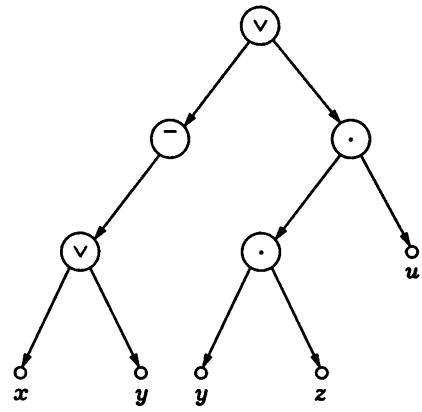


Рис. 6.3

б. Рассмотрим множество булевых функций $\{\oplus, \cdot, 1\}$, которое называют ***базисом Жегалкина***. При записи формул над базисом Жегалкина используют те же принципы, что и при записи формул над стандартным базисом. Приоритет операции конъюнкции считается выше приоритета операции *сложения по модулю* 2. Так как последняя ассоциативна, то при записи формул с этой операцией соответствующим образом опускают скобки. Так, формулами над базисом Жегалкина будут:

$$xy \oplus x \oplus y, \quad x \oplus 1, \quad xyz \oplus xy \oplus xz \oplus yz \oplus x \oplus y \oplus z \oplus 1.$$

Процесс построения первых двух формул изображен в виде деревьев на рис. 6.4.

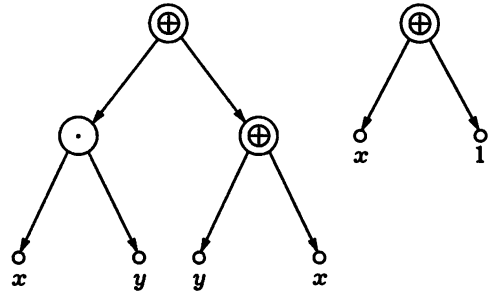


Рис. 6.4

в. Пусть множество базисных функций F состоит из единственной функции $|$ (*штрих Шеффера*). Как бинарная операция, эта функция не ассоциативна, т.е. булева функция $(x|y)|z$ не равна булевой функции $x|(y|z)$. Поэтому при записи формул над множеством $\{|\}$ следует заботиться о расстановке скобок. Примеры формул над множеством $\{|\}$:

$$(x|x)|(y|y), \quad (x|y)|(x|y), \quad (x|x)|(x|x).$$

Внешние скобки мы опускаем и в этом случае. Дерево, показывающее процесс построения первой из записанных выше формул, изображено на рис. 6.5. #

Мы будем использовать запись $\Phi(x_1,\ldots,x_n)$, указывая тем самым, что формула Φ содержит переменные $x_1,\ldots, x_n$, и только их. Множество переменных формулы Φ будем обозначать через $Var(\Phi)$.

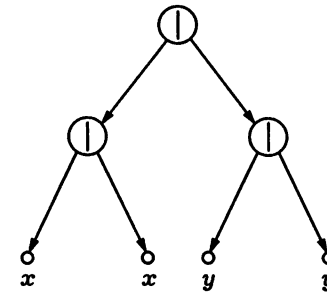


Рис. 6.5

Нам понадобится также понятие ***подформулы.***

Из определения и рассмотренных примеров следует, что процесс построения формулы есть процесс определения некоторой сложной булевой функции, т.е. суперпозиции. Формула „собирается“ из „элементарных формул“, т.е. переменных и базисных функций, так, что на каждом шаге из уже полученных формул строится новая, более сложная формула. Естественно назвать эти „промежуточные“ формулы подформулами рассматриваемой формулы. Так, в примере 6.6.а формулы Φ_1, Φ_2, Φ_3 (и, конечно, переменные и базисные функции) — это подформулы формулы (6.6).

Строго понятие подформулы может быть введено следующим образом. Пусть Ψ — формула над F. Если $\Psi \in F$ или Ψ есть переменное, то ее единственной подформулой является она сама. Если Ψ имеет вид $f(\Phi_1,\ldots,\Phi_n)$, где f — функция из F от n переменных, а Φ_i, $i=\overline{1,n}$, суть формулы над F, то подформулами формулы Ψ будут: 1) все формулы Φ_i; 2) для каждого $i=\overline{1,n}$ все подформулы формулы Φ_i.

В дереве, изображающем процесс построения формулы, каждое *поддерево*, все листья которого являются также и листьями всего дерева, определяет некоторую подформулу.

Каждому набору значений переменных, входящих в заданную формулу, можно определенным образом сопоставить ***значение*** этой ***формулы.*** Вычисление этого значения в точности соответствует процессу построения формулы из подформул (в конечном счете из переменных и базисных функций).

Пример 6.7. Полагая в формуле (6.6) $x=1$, $y=0$, $z=u=1$, получим значение формулы (6.6), равное

$$\overline{(1\vee 0)}\vee 0\cdot 1\cdot 1=0. \quad \#$$

Таким образом, по каждому набору значений переменных формулы можно по определенному алгоритму вычислить значение формулы. Это значит, что каждая формула определяет (или представляет) некоторую булеву функцию. Введем понятие ***функции, представляемой формулой над множеством*** F. Мы полагаем, что:

1) любая константа из F представляет саму себя;

2) любое переменное x из X представляет *проектирующую функцию* x (точнее, любую из множества равных между собой проектирующих функций *существенного переменного* x, см. замечание 6.3);

3) если формулы $\Phi_1, \ldots, \Phi_n$ над множеством F представляют соответственно функции $g_1, \ldots, g_n$, а f — функция из F от n переменных ($n \geqslant 1$), то формула $f(\Phi_1,\ldots,\Phi_n)$ представляет суперпозицию функций $f(g_1,\ldots,g_n)$;

4) других булевых функций, представляемых формулами над множеством F, кроме тех, которые могут быть получены согласно пп. 1–3 данного определения, не существует.

Функцию, представляемую некоторой формулой над множеством F, называют ***суперпозицией над множеством*** F. Таким образом, суперпозиция над множеством F — это любая суперпозиция функций вида $f(g_1,\ldots,g_n)$, где $f\in F$, а каждая из функций $g_1, \ldots, g_n$ есть либо элемент F, либо переменное (точнее, проектирующая функция), либо некоторая суперпозиция над F. Множество всех суперпозиций над F будем обозначать $[F]$ и называть ***замыканием множества*** F ***булевых функций***.

Понятия формулы и суперпозиции взаимно предполагают друг друга. Суперпозиция над множеством F есть некоторая сложная функция, которая определенным образом построена из

базисных функций — функций фиксированного множества F (и проектирующих функций). Само „строение“ суперпозиции, т.е. то, из каких именно базисных функций и в какой последовательности образуется результирующая сложная функция (суперпозиция), и есть формула.

Если булева функция $f(x_1,\dots,x_n)$ представляется формулой $\Phi(x_1,\dots,x_n)$, то будем писать $f(x_1,\dots,x_n)=\Phi(x_1,\dots,x_n)$, или, короче, $f=\Phi(x_1,\dots,x_n)$.

Определение 6.3. Множество булевых функций F называют:

1) ***замкнутым***, если любая формула над F представляет некоторую функцию из F;

2) ***полным***, если любая булева функция может быть представлена некоторой формулой над F.

Определение 6.3 равносильно следующему (на „языке суперпозиций“): множество F функций замкнуто, если каждая суперпозиция над F есть функция из F, т.е. $[F]=F$, и полно, если всякая булева функция есть некоторая суперпозиция над F, т.е. $[F]=\mathcal{P}_2$.

Замечание 6.6. Можно заметить, что определение формулы и суперпозиции над заданным множеством булевых функций похоже на определение *Ω-замыкания* множества в *алгебрах* (см. **4.2**). Эти определения, равно как и основанные на них определения замкнутости и полноты множеств булевых функций, могут быть переведены полностью на алгебраический язык. Тогда замкнутое множество булевых функций, согласно определению 6.3, окажется не чем иным, как *замкнутым подмножеством* некоторой алгебры булевых функций, а полное множество булевых функций — *системой образующих* этой алгебры. Однако при попытке чисто алгебраической интерпретации определения 6.3 возникают некоторые технические трудности, обсуждение которых выходит за рамки этой книги*.

*См., например: *Матросов В.Л., Стеценко В.А.*

Пример 6.8. Для каждой из определенных в **6.1** функций от двух переменных мы можем записать следующие формулы над стандартным базисом:

$$x_1 \oplus x_2 = x_1 \cdot \overline{x}_2 \vee \overline{x}_1 x_2,$$
$$x_1 \to x_2 = \overline{x}_1 \vee x_2,$$
$$x_1 \sim x_2 = (\overline{x}_1 \vee x_2) \cdot (\overline{x}_2 \vee x_1),$$
$$x_1 | x_2 = \overline{x_1 \cdot x_2},$$
$$x_1 \downarrow x_2 = \overline{x_1 \vee x_2}.$$

Если мы пополним стандартный базис функцией $\to$ (импликацией), то формула для эквивалентности примет вид

$$x_1 \sim x_2 = (x_1 \to x_2) \cdot (x_2 \to x_1). \quad \#$$

Тот факт, что одна и та же функция (в данном случае эквивалентность) может быть представлена по крайней мере двумя разными формулами над одним и тем же множеством, а именно над $\{\vee, \cdot, \bar{}, \to\}$, показывает, что соответствие между формулами над фиксированным множеством и представляемыми ими функциями не является взаимно однозначным. Эта ситуация до некоторой степени аналогична разложению по *базису векторов конечномерного линейного пространства* [IV]. Формула, представляющая некоторую булеву функцию, выражает „разложение“ этой функции по фиксированному „функциональному базису“. Одна и та же функция может иметь несколько таких разложений. В отличие от линейной алгебры в этом случае возникает ситуация, когда возможны различные разложения заданной функции по одному и тому же базису. Например, формулы $\overline{(x \vee y)}$ и $\overline{x} \cdot \overline{y}$ над стандартным базисом представляют одну и ту же функцию.

Назовем ***эквивалентными*** формулы, которые представляют равные функции. ***Эквивалентным*** (или ***тождественным***) ***преобразованием формулы*** Φ называют переход (по определенным правилам) к любой формуле Ψ, эквивалентной

формуле Φ. Необходимо сделать несколько замечаний относительно правил, согласно которым осуществляются эквивалентные преобразования формул.

Введем понятие тождества. ***Тождеством*** (над множеством $F \subseteq \mathcal{P}_2$) называют выражение

$$\Phi(x_1,\dots,x_n) = \Psi(y_1,\dots,y_m), \tag{6.7}$$

где формулы Φ и Ψ — эквивалентные формулы над F. Формула Φ называется при этом ***левой***, а формула Ψ — ***правой частью тождества*** (6.7).

Левая и правая части тождества представляют равные булевы функции. Поэтому пересечение множеств переменных $Var(\Phi) = \{x_1,\dots,x_n\}$ и $Var(\Psi) = \{y_1,\dots,y_m\}$ должно содержать все *существенные переменные* функций $f(x_1,\dots,x_n)$ и $g(y_1,\dots,y_m)$, представляемых формулами Φ и Ψ соответственно. В частности, если это пересечение пусто, то обе функции равны некоторой константе.

Пример 6.9. В тождествах $x \vee \overline{x} = y \vee \overline{y}$, $x \vee \overline{x} = 1$ пересечение множеств переменных в левых и правых частях пусто, причем во втором тождестве правая часть вообще не содержит переменных. В тождестве $(x \vee y)\cdot(z \vee \overline{z}) = x \vee y \vee v \cdot \overline{v}$ указанное пересечение равно $\{x, y\}$.

Все записанные в примере 6.8 выражения являются тождествами над множеством $\{\vee, \cdot, \bar{}, \oplus, \to, \sim, |, \downarrow\}$, причем во всех этих тождествах множества переменных в левой и правой частях тождества совпадают. Такого же рода тождества — *аксиомы булевой алгебры** (кроме тождеств $x \vee \overline{x} = 1$ и $x \wedge \overline{x} = 0$) и вытекающие из них тождества (подобные, например, *законам де Моргана*). #

Без доказательства сформулируем следующие правила тождественных преобразований.

*Поскольку все переменные, фигурирующие в этих тождествах, есть булевы переменные, то речь здесь идет об аксиомах булевой алгебры применительно к частному случаю — двухэлементной булевой алгебре $\mathbb{B}$.

Теорема 6.1. 1. Если в тождестве (6.7) некоторые переменные заменить произвольными формулами (над множеством F), то тождество сохранится, т.е. полученные в результате такой замены новые формулы останутся эквивалентными.

2. Если в формуле Φ произвольную ее подформулу заменить любой эквивалентной ей, то получится формула, эквивалентная формуле Φ. #

Чтобы использовать сформулированные в теореме 6.1 правила, нужно фиксировать какую-то систему исходных тождеств. Тогда возникает вопрос: можно ли утверждать, что при надлежащем выборе исходных тождеств с помощью правил 1 и 2, сформулированных в теореме 6.1, можно из формулы Φ получить эквивалентную ей формулу Ψ, каковы бы ни были эти формулы, или, говоря неформально, любые ли две эквивалентные формулы над заданным множеством F можно „трансформировать" друг в друга, используя фиксированную систему основных тождеств над F и правила, сформулированные в теореме 6.1?

Рассмотрение этого вопроса выходит за рамки учебника. Ответ на него зависит от того, какое множество булевых функций F и какая система исходных тождеств над F выбраны. Отметим, что для стандартного базиса ответ на вопрос положителен, причем в качестве исходных тождеств используются аксиомы булевой алгебры.

В свете изложенного может быть поставлена задача поиска наиболее простой (в определенном смысле) формулы, среди всех эквивалентных между собой формул, представляющих данную булеву функцию. Решение этой задачи для некоторого класса формул над стандартным базисом будет рассмотрено в **6.6**.

Понятие формулы позволяет также взглянуть по-новому на логическую интерпретацию булевой функции. В силу установленного в **6.2** взаимно однозначного соответствия между логическими связками $\vee$, $\wedge$, $\neg$, $\Rightarrow$, $\Leftrightarrow$ и булевыми функциями $\vee$, $\cdot$, $^-$, $\to$, $\sim$ любому сложному высказыванию, составленно-

му из некоторых „простых“ высказываний с использованием указанных выше логических связок, однозначно сопоставляется формула над множеством $L = \{\vee, \wedge, \bar{}, \to, \sim\}$, т.е. каждому простому высказыванию сопоставляется булево переменное (так что разным высказываниям сопоставляются и разные переменные), а связки $\vee$, $\wedge$, $\neg$, $\Rightarrow$, $\Leftrightarrow$ заменяются соответствующими функциями из множества L. Тогда, например, высказыванию $(P \Rightarrow Q) \wedge (Q \Rightarrow P)$ (читается: „если P, то Q, и если Q, то P“) будет сопоставлена формула $(x \to y) \cdot (y \to x)$.

Таким образом, с логической точки зрения формула над множеством L есть высказывание. Поскольку мы имеем возможность вычислять значения формул и проводить их эквивалентные преобразования (используя, в частности, аксиомы булевой алгебры), мы получаем алгебраический аппарат для упрощения сложных высказываний (путем эквивалентных преобразований) и вычисления их истинностных значений.

Но тогда возникают вопросы: как практически построить для любой наперед заданной булевой функции представляющую ее формулу над фиксированным множеством базисных функций F? Каковы условия полноты множества F?

Далее (см. **6.5** и **6.6**) мы рассмотрим вопрос о представлении любой булевой функции над стандартным базисом и вопрос о поиске минимальной (в уточняемом ниже смысле), наиболее простой формулы над стандартным базисом, представляющей данную функцию.

6.5. Дизъюнктивные и конъюнктивные нормальные формы

*Любая **формула** вида x или $\overline{x}$ над стандартным базисом*, где x — произвольное переменное, называется ***литералом***. Таким образом, литерал есть обозначение либо самого переменного x, либо его отрицания. Часто используют такое обозначение: для $\sigma \in \{0, 1\}$ пишут x^σ, понимая под этим само переменное x, если

$\sigma = 1$, и отрицание x, если $\sigma = 0$, т.е.

$$x^\sigma = \begin{cases} x, & \sigma = 1; \\ \overline{x}, & \sigma = 0. \end{cases} \tag{6.8}$$

Подставляя в (6.8) 0 и 1 вместо x, получаем

$$0^\sigma = \begin{cases} 0, & \sigma = 1; \\ 1, & \sigma = 0, \end{cases} \qquad 1^\sigma = \begin{cases} 1, & \sigma = 1; \\ 0, & \sigma = 0. \end{cases}$$

Часто используют также обозначение $\widetilde{x}$, понимая под этим любой из двух литералов — x или $\overline{x}$.

Формула вида $\widetilde{x}_1\widetilde{x}_2\ldots\widetilde{x}_m$ (соответственно вида $\widetilde{x}_1 \vee \widetilde{x}_2 \vee \ldots \vee \widetilde{x}_m$), где все фигурирующие в ней переменные попарно различны, называется ***элементарной конъюнкцией*** (соответственно ***элементарной дизъюнкцией***).

Определение 6.4. ***Дизъюнктивная нормальная форма*** (***ДНФ***) от переменных $x_1, \ldots, x_n$ — это формула вида $K_1 \vee \ldots \vee K_m$, где K_i, $i = \overline{1,m}$, — элементарная конъюнкция, содержащая некоторые из литералов $x_1, \ldots, x_n$. В том случае, когда в каждую конъюнкцию K_i для каждого номера $j = \overline{1,n}$ входит в точности один из литералов $\widetilde{x}_j$, ДНФ называется ***совершенной дизъюнктивной нормальной формой*** (***СДНФ***).

Двойственным образом, т.е. с использованием *принципа двойственности для булевых алгебр*, определяются ***конъюнктивная нормальная форма*** (***КНФ***) и ***совершенная конъюнктивная нормальная форма*** (***СКНФ***).

Теорема 6.2. Любая булева функция, отличная от константы 0 (соответственно от константы 1) представима в виде СДНФ (соответственно в виде СКНФ).

◀ Докажем первое из двух (взаимно двойственных) утверждений теоремы. Для функции $f \in \mathcal{P}_{2,n}$, не равной тождественно 0, рассмотрим множество $C_f^1 = \{\widetilde{\alpha}\colon f(\widetilde{\alpha}) = 1\}$. Каждый набор из C_f^1 называется ***конституентой единицы*** функции f. Так

как по условию $f \neq 0$ тождественно, то множество C_f^1 не пусто. Каждому набору $\tilde{\alpha} \in C_f^1$ поставим в соответствие элементарную конъюнкцию $K_{\tilde{\alpha}} = x_1^{\alpha_1} x_2^{\alpha_2} \dots x_n^{\alpha_n}$, которую также называют конституентой единицы функции f. Ясно, что $K_{\tilde{\alpha}}$ обращается в единицу только на наборе $\tilde{\alpha}$. Тогда искомой СДНФ для функции f будет

$$f = \bigvee_{\tilde{\alpha} \in C_f^1} K_{\tilde{\alpha}}.$$

Согласно принципу двойственности, СКНФ для той же функции будет иметь вид

$$f = \bigwedge_{\tilde{\alpha} \in C_f^0} D_{\tilde{\alpha}},$$

где множество наборов $C_f^0 = \{\tilde{\alpha}\colon f(\tilde{\alpha}) = 0\}$, и каждый набор $\tilde{\alpha}$ из C_f^0 называют ***конституентой нуля*** функции f; при этом элементарная дизъюнкция $D_{\tilde{\alpha}}$ сопоставляется конституенте нуля $\tilde{\alpha}$ по следующему правилу:

$$D_{\tilde{\alpha}} = x_1^{\overline{\alpha}_1} \vee x_2^{\overline{\alpha}_2} \vee \dots \vee x_n^{\overline{\alpha}_n},$$

т.е. если в наборе i-я компонента равна 0, то в $D_{\tilde{\alpha}}$ ставим само переменное x_i, если иначе — отрицание переменного x_i (таким образом, дизъюнкция $D_{\tilde{\alpha}}$ обращается в нуль только на наборе $\tilde{\alpha}$). ▶

Из доказанного следует, что любая булева функция может быть представлена в виде формулы над стандартным базисом (СДНФ или СКНФ), и, значит, стандартный базис есть полное множество булевых функций.

Рассмотрим в качестве примера построение СДНФ и СКНФ для *мажоритатрной функции* (см. **6.1**). Конституентами единицы для нее служат наборы: $\tilde{\alpha}_1 = (0, 1, 1)$, $\tilde{\alpha}_2 = (1, 0, 1)$, $\tilde{\alpha}_3 = (1, 1, 0)$ и $\tilde{\alpha}_4 = (1, 1, 1)$. Им соответствуют элементарные конъюнкции: $K_{\tilde{\alpha}_1} = \overline{x}_1 x_2 x_3$, $K_{\tilde{\alpha}_2} = x_1 \overline{x}_2 x_3$, $K_{\tilde{\alpha}_3} = x_1 x_2 \overline{x}_3$,

$K_{\tilde{\alpha}_4} = x_1x_2x_3$. Тогда СДНФ, представляющая мажоритарную функцию, имеет вид

$$\overline{x}_1x_2x_3 \vee x_1\overline{x}_2x_3 \vee x_1x_2\overline{x}_3 \vee x_1x_2x_3. \qquad (6.9)$$

Для получения СКНФ для той же функции выпишем все конституенты нуля данной функции: (0, 0, 0), (0, 0, 1), (0, 1, 0), (1, 0, 0). Сопоставим им элементарные дизъюнкции: $x_1 \vee x_2 \vee \vee x_3$, $x_1 \vee x_2 \vee \overline{x}_3$, $x_1 \vee \overline{x}_2 \vee x_3$ и $\overline{x}_1 \vee x_2 \vee x_3$ соответственно.

В результате получим СКНФ для мажоритарной функции в виде

$$(x_1 \vee x_2 \vee x_3)(x_1 \vee x_2 \vee \overline{x}_3)(x_1 \vee \overline{x}_2 \vee x_3)(\overline{x}_1 \vee x_2 \vee x_3). \qquad (6.10)$$

Заметим, что если в формуле СКНФ (6.10) мы раскроем скобки и преобразуем полученное выражение согласно законам булевой алгебры, проведя тем самым *эквивалентные преобразования* СКНФ, то придем к формуле СДНФ (6.9) .

6.6. Построение минимальных ДНФ

СДНФ, которая строится по *таблице булевой функции*, зачастую оказывается весьма сложной, т.е. она содержит достаточно много *элементарных конъюнкций* и *литералов*. Необходимо уметь находить в определенном смысле минимальную ДНФ, *представляющую* исходную функцию.

Уточним задачу.

Определение 6.5. Булеву функцию g называют ***импликантой**** булевой функции f, если для любых *наборов значений переменных* из $g = 1$ следует $f = 1$.

Замечание 6.7. Напомним, что функции f и g можно рассматривать как функции от одного и того же числа переменных (см. **6.3**). Обозначая это число через n, можно так уточнить

*Иногда употребляют и термин „импликант“ (мужского рода).

понятие импликанты: функция $g \in \mathcal{P}_{2,n}$ есть импликанта функции $f \in \mathcal{P}_{2,n}$, если для каждого *набора* $\widetilde{\alpha} \in \mathbb{B}^n$ из $g(\widetilde{\alpha}) = 1$ следует $f(\widetilde{\alpha}) = 1$. Термин „импликанта" естественным образом ассоциируется и с логической связкой, называемой импликацией, и с одноименной булевой функцией. Действительно, если g импликанта f, то из $(g \to f) = 1$ и $g = 1$ следует, что и $f = 1$, т.е. истинно высказывание

$$(\forall \widetilde{\alpha} \in \mathbb{B}^n)((g(\widetilde{\alpha}) = 1) \Rightarrow (f(\widetilde{\alpha}) = 1)). \quad \#$$

Если функция f представлена СДНФ, то любая ее элементарная конъюнкция (*конституента единицы* функции f) будет ее импликантой. Полезно заметить также, что если g_1 и g_2 — импликанты f, то дизъюнкция $g_1 \vee g_2$ также является импликантой f. Действительно, если $g_1 \vee g_2 = 1$, то $g_1 = 1$ или $g_2 = 1$. Но тогда, поскольку каждая из этих функций есть импликанта f, и $g_1 \vee g_2$ есть импликанта f.

Из определения 6.5 и понятия *равных булевых функций* (см. определение 6.2) следует, что булевы функции f и g равны, если и только если каждая из них служит импликантой другой: $f = 1 \Leftrightarrow g = 1$.

Определение 6.6. ДНФ называют ***минимальной***, если она содержит наименьшее число *литералов* среди всех ДНФ, эквивалентных ей.

Обратим внимание на то, что под числом литералов в ДНФ понимают число всех подформул этой ДНФ, которые являются литералами. Так, СДНФ (6.9) содержит 12 литералов (по три литерала в каждой из четырех элементарных конъюнкций).

Пример 6.10. ДНФ $x_1x_2 \vee \overline{x}_1x_2$ не является минимальной, так как ее можно преобразовать к эквивалентной ДНФ, не содержащей ни одного из литералов $\widetilde{x}_1$:

$$x_1x_2 \vee \overline{x}_1x_2 = (x_1 \vee \overline{x}_1)x_2 = x_2.$$

Вместо четырех литералов в исходной ДНФ получаем ДНФ, состоящую из одного литерала.

Определение 6.7. ***Длиной ДНФ*** называют число входящих в нее элементарных конъюнкций.

ДНФ называют ***кратчайшей***, если она имеет наименьшую длину среди всех эквивалентных ей ДНФ.

Заметим, что кратчайшая ДНФ не обязана быть в то же время минимальной среди всех ДНФ, эквивалентных исходной функции. Но поиск минимальных ДНФ, как мы сейчас увидим, проводится среди кратчайших ДНФ.

Наша задача состоит в том, чтобы описать метод построения минимальной ДНФ, эквивалентной заданной булевой функции. Мы рассмотрим простейший метод такого рода, основанные на ***алгоритме Квайна — Мак-Клоски***. Этот алгоритм исходит обязательно из СДНФ, которая строится по таблице функции так, как это было описано ранее (см. **6.5**).

Опишем последовательно этапы, составляющие алгоритм Квайна — Мак-Клоски.

1. Склейка. Пусть K_1 и K_2 — две элементарные конъюнкции, входящие в исходную СДНФ Ф, которая представляет функцию f, причем для некоторого переменного x и некоторой элементарной конъюнкции K выполняются равенства $K_1 = xK$ и $K_2 = \overline{x}K$.

Тогда имеем, согласно тождествам *булевой алгебры*,

$$K_1 \vee K_2 = xK \vee \overline{x}K = (x \vee \overline{x})K = K.$$

Мы получаем элементарную конъюнкцию K, которая содержит на один литерал меньше, чем K_1 и K_2, и является, как и обе конъюнкции K_1 и K_2, импликантой f. Образно говоря, мы „склеили“ две импликанты в одну, в которой число литералов на единицу меньше.

Операцию получения K по K_1 и K_2, описанную выше, можно провести и для любых двух элементарных конъюнкций подоб-

ного вида, составляющих любую ДНФ, эквивалентную исходной функции. Такую операцию называют ***простой склейкой*** импликант K_1 и K_2 по переменному x.

Установим геометрический смысл простой склейки* (с точки зрения структуры, или „геометрии", *булева куба*, см. **6.1**).

Из доказательства теоремы о представлении булевой функции в виде ДНФ (см. теорему 6.2) мы знаем, что существует *взаимно однозначное соответствие* между множеством элементарных конъюнкций СДНФ, представляющей функцию f, и множеством C_f^1 ее конституент единицы. Это соответствие, напомним, таково, что каждому набору $\tilde{\alpha} = (\alpha_1, \ldots, \alpha_n) \in C_f^1$ отвечает элементарная конъюнкция $K_{\tilde{\alpha}} = x_1^{\alpha_1} \cdot \ldots \cdot x_n^{\alpha_n}$, принимающая значение 1 только на наборе $\tilde{\alpha}$. Тогда простая склейка может быть применена только к таким двум элементарным конъюнкциям $K_{\tilde{\alpha}}$ и $K_{\tilde{\beta}}$, соответствующим наборам $\tilde{\alpha}$, $\tilde{\beta} \in C_f^1$, что для некоторого i $(1 \leqslant i \leqslant n)$

$$\tilde{\alpha} = (\alpha_1, \ldots, \alpha_{i-1}, \alpha_i, \alpha_{i+1}, \ldots, \alpha_n),$$
$$\tilde{\beta} = (\alpha_1, \ldots, \alpha_{i-1}, \overline{\alpha}_i, \alpha_{i+1}, \ldots, \alpha_n).$$

Это значит, что наборы $\tilde{\alpha}$, $\tilde{\beta}$ таковы, что один из них *доминирует* над другим (они различаются значением только одной компоненты), т.е. они образуют *ребро* булева куба $\mathbb{B}^n$.

Следовательно, простой склейке, применяемой к элементарным конъюнкциям исходной СДНФ, представляющей функцию f, подлежат те и только те элементарные конъюнкции, которые соответствуют элементам какого-либо ребра булева куба, на котором функция f принимает единичное значение. Образно говоря, две соседние вершины куба, на которых функция равна 1, „склеиваются" в ребро, их „соединяющее".

С алгебраической же точки зрения мы из двух элементарных конъюнкций $K_{\tilde{\alpha}}$ и $K_{\tilde{\beta}}$ получаем новую элементарную конъюнкцию $x_1^{\alpha_1} \ldots x_{i-1}^{\alpha_{i-1}} x_{i+1}^{\alpha_{i+1}} \ldots x_n^{\alpha_n}$, лишенную литерала $x_i^{\alpha_i}$.

*Подробно о геометрической сути минимизации см.: *Яблонский С.В.*

Итак, применяя простую склейку к исходной СДНФ Ф, получаем новую ДНФ Φ_1; к ней также применяем простую склейку — получаем ДНФ Φ_2; продолжаем выполнять эту операцию до тех пор, пока не окажется, что для некоторого k в ДНФ Φ_k уже нельзя склеить никакие две элементарные конъюнкции. Такое k всегда найдется, так как СДНФ Ф состоит из конечного числа элементарных конъюнкций, а они, в свою очередь, состоят из конечного числа литералов. Полученную в результате ДНФ Φ_k называют ***сокращенной ДНФ*** функции f, а ее элементарные конъюнкции — ***простыми импликантами*** булевой функции f.

Замечание 6.8. Понятие простой импликанты определено через процедуру многократного повторения простой склейки. Иногда простую импликанту булевой функции f определяют независимо от понятия о склейке как такую элементарную конъюнкцию в составе некоторой ДНФ, представляющей функцию f, что удаление из нее любого литерала лишает ее свойства „быть импликантой“. Например, конъюнкция $x_1x_2\overline{x}_3$ не является простой импликантой *мажоритарной функции*, так как из ее СДНФ (6.9) можно удалить литерал $\overline{x}_3$ и получить конъюнкцию x_1x_2, которая будет снова импликантой функции, но уже, как будет показано далее, простой.

Можно доказать, что эти два определения простой импликанты равносильны. #

Геометрия описанного выше многократного повторения простой склейки, как можно показать, состоит в дальнейшем „склеивании“ каждой пары *соседних* ребер (*граней* размерности 1), на которых значение функции равно 1, в грани размерности 2, соседних граней размерности 2 в грани размерности 3 и т.д. Разбираемый ниже пример поясняет эту идею.

Пример 6.11. Зададим функцию f от трех переменных следующей СДНФ:

$$f = \overline{x}_1\overline{x}_2\overline{x}_3 \vee \overline{x}_1\overline{x}_2x_3 \vee x_1\overline{x}_2\overline{x}_3 \vee x_1\overline{x}_2x_3. \qquad (6.11)$$

Подвергнем простой склейке первую и третью, а также вторую и четвертую элементарные конъюнкции в (6.11):

$$f = \overline{x}_2\overline{x}_3 \vee \overline{x}_2 x_3. \qquad (6.12)$$

С геометрической точки зрения склейка первой и третьей конъюнкций в формуле (6.11) означает, что функция f принимает единичное значение на ребре [000, 100] (рис. 6.6), а склейка второй и четвертой конъюнкций точно так же определяет ребро [001, 101]. Эти ребра являются соседними, и, кроме того, оказывается, что функция f принимает единичное значение и на другой паре соседних ребер: [000, 001] и [100, 101]. Здесь сказывается существенное отличие „геометрии" булева куба от классической: в булевом кубе ребро — это пара вершин, между которыми нет никаких „точек". Тогда любая пара соседних ребер образует грань размерности 2, любая пара соседних граней размерности 2 образует грань размерности 3 и т.д. Таким образом, если функция принимает единичное значение на двух соседних ребрах булева куба, то она равна 1 в любой точке образуемой ими грани размерности 2, если она равна 1 на двух параллельных соседних гранях размерности 2, то она равна 1 на соответствующей грани размерности 3 и т.д.

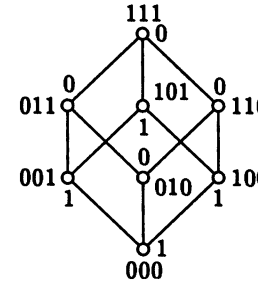


Рис. 6.6

Применяя простую склейку к (6.12) (по переменному x_3), получаем $f(x_1, x_2, x_3) = \overline{x}_2$. Побочным результатом склейки явилось и удаление *фиктивных переменных* функции x_1 и x_3.

Пример 6.12. Рассмотрим СДНФ мажоритарной функции (6.9).

Имеем следующие склейки:

$$\overline{x}_1 x_2 x_3 \vee x_1 x_2 x_3 = x_2 x_3,$$
$$x_1 \overline{x}_2 x_3 \vee x_1 x_2 x_3 = x_1 x_3,$$
$$x_1 x_2 \overline{x}_3 \vee x_1 x_2 x_3 = x_1 x_2.$$

В данном случае сразу получаем сокращенную ДНФ:

$$\Phi_1 = x_1x_2 \vee x_1x_3 \vee x_2x_3. \quad \#$$

Для булевых функций от трех и четырех переменных процедура склейки наглядно и просто выполняется на так называемых ***картах Карно***. Форма карт Карно, представляющих собой прямоугольные таблицы, для функции от трех переменных показана на рис. 6.7, а для функции от четырех переменных — на рис. 6.8. На рис. 6.7 строки отмечены наборами значений переменного x_1, а столбцы — x_2, x_3, а на рис. 6.8 строки — наборами значений переменных x_1, x_2, а столбцы — x_3, x_4.

x_1 \ x_2x_3	00	01	11	10
0				
1				

Рис. 6.7

x_1x_2 \ x_3x_4	00	01	11	10
00				
01				
11				
10				

Рис. 6.8

Карта Карно есть не что иное, как форма таблицы для определения булевой функции. Каждая клетка карты задается своим набором значений переменных, причем в клетках, соответствующих конституентам единицы данной функции, ставится единица, тогда как остальные клетки остаются пустыми. Карта Карно устроена так, что наборы, определяющие любые две соседние клетки, различаются в точности в одной позиции (т.е. различаются значениями ровно одной компоненты), причем клетки (одной и той же строки или одного и того же столбца), примыкающие к противоположным сторонам прямоугольника, также являются соседними в только что определенном смысле. Это можно представить себе так, что карта

„закручивается“ в „цилиндр“ по обоим направлениям, т.е. в „тор“.

С геометрической точки зрения карта Карно есть способ изображения булева куба (размерностей 3 и 4)*. Любая пара соседних клеток (с учетом „закрученности“ карты) определяет некоторое ребро булева куба, а любой прямоугольник, состоящий из 2^k клеток (или, как говорят, прямоугольник с площадью 2^k) для некоторого k, определяет грань размерности k.

Пусть булева функция f задана таблицей, представленной в форме карты Карно. Описанный выше итерационный процесс склейки, в результате которого получается сокращенная ДНФ, представляющая функцию f, проводится на карте Карно так: любые две соседние клетки, содержащие единицы, обводятся, и „поглотивший“ их прямоугольник (он и есть обозначение результата склейки на карте) представляется словом, содержащим „0“, „1“ и „×“ („крестик“), причем „крестик“ занимает позицию того переменного, по которому произведена склейка (рис. 6.9).

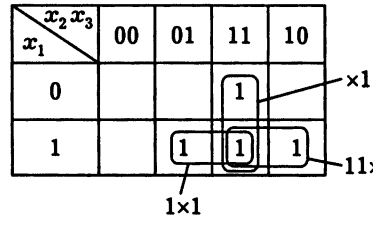


Рис. 6.9

С геометрической точки зрения такой прямоугольник площади 2 соответствует ребру булева куба, в каждой вершине которого функция принимает значение 1. Запись прямоугольника в виде слова можно понимать как обозначение соответствующе-

*Можно построить карты Карно и для размерностей 5 и 6, но они используются весьма редко. Может быть построена и простейшая карта Карно для функции от двух переменных, но для таких функций не возникает нетривиальных задач построения минимальной ДНФ.

го ребра. Так, на карте, показанной на рис. 6.9, прямоугольник $11\times$ обозначает ребро [110, 111], прямоугольники же 1×1 и $\times11$ — ребра [101, 111] и [011, 111] соответственно.

По таким обозначениям легко получить и ту импликанту, которая является результатом простой склейки: для этого достаточно записать литерал x_i (соответственно $\overline{x}_i$), если в i-й позиции стоит 1 (соответственно 0), и пропустить литерал x_i, если в i-й позиции стоит „крестик“. Так, по слову 1×0 получим импликанту $x_1\overline{x}_3$.

Наличие на карте Карно двух прямоугольников площади 2, находящихся в соседних столбцах или строках, показывает, что функция принимает значение 1 на некоторой паре соседних ребер, т.е. на некоторой грани размерности 2. Тогда они могут быть объединены в один большой прямоугольник площади 4 (рис. 6.10).

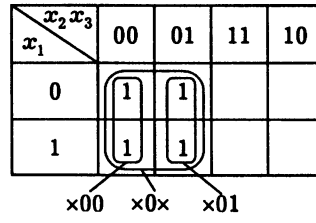


Рис. 6.10

Этот прямоугольник можно записать в виде слова $\times0\times$, показывая тем самым, что соответствующая грань (размерности 2) образована любой из двух пар соседних ребер: $(\times00, \times01)$ (два вертикальных прямоугольника площади 2) или $(00\times, 10\times)$ (два горизонтальных прямоугольника площади 2).

Точно так же можно объединять в один прямоугольник площади 8 два соседних прямоугольника площади 4 (рис. 6.11).

Если такие большие прямоугольники находить сразу, то „поглощаемые“ ими меньшие прямоугольники уже не рассматриваются. Тем самым, находя на карте Карно прямоугольники максимальной площади и не содержащиеся друг в друге, мы

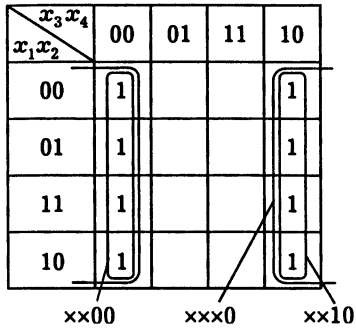


Рис. 6.11

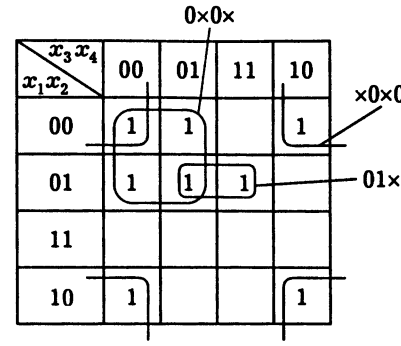


Рис. 6.12

находим грани максимальных размерностей и максимальные по включению, такие, на которых заданная функция принимает единичное значение. Поскольку грань размерности k имеет 2^k вершин, то выделяемые описанным способом прямоугольники могут состоять только из 2^k клеток (для некоторого k, не превышающего числа переменных). Так, на карте, приведенной на рис. 6.12, получим два прямоугольника площади 4: $\times 0 \times 0$ и $0 \times 0 \times$, соответствующие граням размерности 2, и один прямоугольник 01×1, отвечающий ребру, которое не содержится ни в одной из указанных выше граней. Подчеркнем еще раз, что соседство клеток, прямоугольников и само выделение прямоугольников на карте Карно производится с учетом ее „закрученности“. В этой связи интересен „прямоугольник“ на карте, приведенной на рис. 6.12, обозначенный $\times 0 \times 0$. Он образован двумя парами противоположных угловых клеток.

Таким образом, если на карте Карно сразу выделять все максимальные (в указанном выше смысле) прямоугольники площади 2^k (для некоторого $k \geqslant 0$ и не превышающего числа переменных), то тем самым мы „геометрически“ реализуем описанный ранее алгебраический итерационный процесс склейки и в результате получаем все простые импликанты исходной функции (составляющие сокращенную ДНФ). Эти импликанты восстанавливаются по записям прямоугольников точно так

же, как описано выше для простой склейки. Так, для карты, приведенной на рис. 6.12, получим сокращенную ДНФ в виде $\overline{x}_2\overline{x}_4 \vee \overline{x}_1\overline{x}_3 \vee \overline{x}_1x_2x_4$.

2. Определение ядра. Говорят, что элементарная конъюнкция K покрывает элементарную конъюнкцию L (и пишут $K \succ L$), если любой литерал, входящий в K, входит в L. Так, $x_1x_2 \succ x_1x_2x_3$, $x_1x_3 \succ x_1\overline{x}_2x_3$, но $x_1x_3 \not\succ x_1x_2\overline{x}_3$, поскольку вторая конъюнкция содержит литерал $\overline{x}_3$, отсутствующий в первой конъюнкции. Легко понять, что если $K \succ L$, то $K \vee L = K$ (согласно *тождествам поглощения*).

Каждая входящая в сокращенную ДНФ простая импликанта покрывает некоторую элементарную конъюнкцию исходной СДНФ. На карте Карно этому отвечает прямоугольник, „закрывающий“ соответствующую единицу.

Простую импликанту называют ***ядровой***, если она покрывает некоторую элементарную конъюнкцию исходной СДНФ, не покрываемую никакой другой простой импликантой. На карте Карно прямоугольник, соответствующий ядровой импликанте, отыскивается очень просто: это такой прямоугольник, удалив который получим единицу, не закрытую никаким другим прямоугольником. Тогда ни одна ядровая импликанта не может быть удалена из искомой минимальной ДНФ исходной функции, т.е. все ядровые импликанты обязательно войдут в минимальную ДНФ.

Множество всех ядровых импликант (склеек) сокращенной ДНФ называют ***ядром***.

Пример 6.13. а. У мажоритарной функции все импликанты являются ядровыми. Напротив, у функции, изображенной на карте Карно на рис. 6.13, ядро пусто, т.е. ядровых импликант нет вовсе.

б. На карте Карно на рис. 6.14 в ядро попадают склейки $0\times\times1$, $0\times1\times$, 1×00. #

Если все простые импликанты оказались в ядре, то сокращенная ДНФ и есть единственная минимальная и кратчайшая

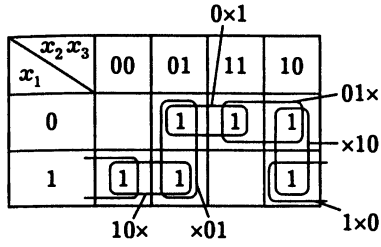


Рис. 6.13

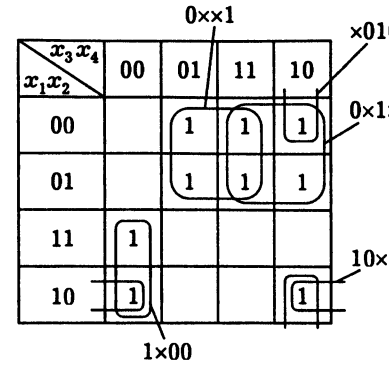


Рис. 6.14

ДНФ для данной функции. Именно так обстоит дело с мажоритарной функцией (см. пример 6.12). В противном случае смотрят, не эквивалентна ли ДНФ, построенная как дизъюнкция всех ядровых импликант, исходной СДНФ. Это будет иметь место тогда и только тогда, когда ядровые импликанты покрывают в совокупности все элементарные конъюнкции исходной СДНФ. На карте Карно тогда каждая клетка, содержащая единицу, должна быть закрыта прямоугольником, отвечающим некоторой ядровой импликанте. Если это так, то ДНФ, построенная по ядру, как описано выше, есть минимальная и кратчайшая (склейки ядра закрыли все единицы карты Карно). При этом импликанты, не попавшие в ядро, все оказываются „избыточными“, т.е. их удаление из сокращенной ДНФ не приводит к нарушению эквивалентности этой последней с исходной СДНФ.

В остальных случаях переходят к отысканию так называемых тупиковых ДНФ.

3. Перечисление тупиковых ДНФ. Простую импликанту называют ***избыточной*** (относительно некоторой ДНФ, содержащей только простые импликанты и эквивалентной исходной СДНФ), если ее можно удалить из этой ДНФ без потери эквивалентности ее исходной СДНФ. Так, сокращенная ДНФ

(см. рис. 6.14) содержит избыточные импликанты: импликанта, соответствующая прямоугольнику 10×0, или импликанта, соответствующая прямоугольнику $\times010$, может быть удалена (но не обе сразу!). Это значит, что каждая из этих импликант является избыточной относительно сокращенной ДНФ, но удаление одной из них приводит к новой ДНФ, относительно которой вторая из упомянутых импликант уже не будет избыточной. В том случае, когда каждую элементарную конъюнкцию исходной СДНФ покрывает некоторая ядровая импликанта, импликанты, не вошедшие в ядро, можно удалить одновременно.

Тогда можно представить процесс пошагового удаления избыточных импликант, начиная с сокращенной ДНФ, в результате которого получится некоторая ДНФ, уже не содержащая ни одной избыточной склейки.

Любую ДНФ, эквивалентную исходной СДНФ, содержащую все ядровые импликанты и не содержащую ни одной избыточной импликанты, называют ***тупиковой***.

Заметим, что в силу конечности множества всех импликант тупиковая ДНФ обязательно существует, т.е. в упомянутом выше процессе мы рано или поздно доберемся до такого момента, когда удаление хотя бы одной склейки приведет к тому, что „откроется“ какая-то единичная клетка на карте Карно и тем самым будет потеряна эквивалентность полученной таким образом ДНФ исходной СДНФ.

Для СДНФ, карта Карно которой приведена на рис. 6.14, имеются две тупиковые ДНФ:

$$\overline{x}_1x_4 \vee \overline{x}_1x_3 \vee x_1\overline{x}_3\cdot\overline{x}_4 \vee \overline{x}_2x_3\overline{x}_4,$$
$$\overline{x}_1x_4 \vee \overline{x}_1x_3 \vee x_1\overline{x}_3\cdot\overline{x}_4 \vee x_1\overline{x}_2\cdot\overline{x}_4.$$

Первые три конъюнкции соответствуют ядру.

В общем случае для перечисления всех тупиковых ДНФ может быть использован следующий алгоритм. Мы изложим его в терминах карт Карно и, допуская вольность речи, будем отождествлять максимальные прямоугольники на карте Карно с соответствующими простыми импликантами.

Присвоим каждой простой импликанте сокращенной ДНФ некоторое имя: т.е. обозначим их, например, как K_1, K_2, ..., K_m. Для любой единицы карты Карно, не покрываемой ядром, перечислим все простые импликанты, которые ее покрывают, записав их в виде *элементарной дизъюнкции*, в которой переменными считаются введенные выше имена простых импликант. Переменное, именующее данную простую импликанту, принимает, по определению, значение 1, если данная простая импликанта выбирается для покрытия рассматриваемой единицы карты Карно.

Записав все элементарные дизъюнкции, составим из них *КНФ*. Рассмотрим карту Карно на рис. 6.13. Обозначив

$$K_1 = x_1\overline{x}_2\,(10\times), \quad K_2 = \overline{x}_2x_3\,(\times 01), \quad K_3 = \overline{x}_1x_3\,(0\times 1),$$
$$K_4 = \overline{x}_1x_2\,(01\times), \quad K_5 = x_2\overline{x}_3\,(\times 10), \quad K_6 = x_1\overline{x}_3\,(1\times 0),$$

получим

$$(K_1 \vee K_6) \wedge (K_1 \vee K_2) \wedge (K_2 \vee K_3) \wedge \\ \wedge (K_3 \vee K_4) \wedge (K_4 \vee K_5) \wedge (K_5 \vee K_6). \quad (6.13)$$

Тем самым мы образуем вспомогательную функцию (представленную КНФ вида (6.13)), называемую ***функцией Патрика***. Раскрывая скобки в КНФ (6.13) и используя тождества булевой алгебры (в частности, тождество поглощения), получим ДНФ, в которой каждая элементарная конъюнкция соответствует некоторой тупиковой ДНФ и, наоборот, каждой тупиковой ДНФ может быть сопоставлена одна из этих конъюнкций.

Для нашего примера поступим так: вычислим конъюнкцию первой и второй скобки в выражении (6.13), а также третьей и четвертой, пятой и шестой скобок, после чего получим

$$(K_1 \vee K_1K_2 \vee K_6K_1 \vee K_6K_2) \wedge \\ \wedge (K_2K_3 \vee K_2K_4 \vee K_3 \vee K_3K_4) \wedge \\ \wedge (K_4K_5 \vee K_4K_6 \vee K_5 \vee K_5K_6). \quad (6.14)$$

Используя тождества поглощения, в первой скобке в формуле (6.14) мы можем удалить все члены, содержащие K_1, во второй скобке — все члены, содержащие K_3, в третьей скобке — все члены, содержащие K_5. Проделав это, раскрыв все три скобки и применив еще раз поглощение, окончательно получим

$$K_1K_3K_5 \vee K_1K_3K_4K_6 \vee \\ \vee K_1K_2K_4K_5 \vee K_2K_3K_5K_6 \vee K_2K_4K_6. \quad (6.15)$$

Элементарные конъюнкции в (6.15) определяют тупиковые ДНФ. Более того, так как в данном случае отсутствуют ядровые импликанты, найденные конъюнкции исчерпывают тупиковые ДНФ. Первая тупиковая ДНФ состоит из конъюнкций K_1, K_3 и K_5, т.е. имеет вид

$$x_1\overline{x}_2 \vee \overline{x}_1x_3 \vee x_2\overline{x}_3.$$

Точно так же определяются остальные тупиковые ДНФ.

Обоснование описанного выше алгоритма может быть получено из следующих соображений. Функция Патрика, представленная КНФ, принимает значение 1 тогда и только тогда, когда каждая элементарная дизъюнкция принимает значение 1. А элементарная дизъюнкция принимает значение 1 в том и только в том случае, когда хотя бы одно ее переменное принимает значение 1. Согласно определению функции Патрика, это значит, что хотя бы одна простая импликанта выбрана для покрытия соответствующей единицы на карте Карно. Поскольку таким образом перебираются все не покрываемые ядром единицы карты Карно, то гарантируется эквивалентность искомой ДНФ исходной СДНФ. Однако, когда функция Патрика представлена ДНФ и мы выбираем в точности одну из ее элементарных конъюнкций, полагая, что все входящие в нее переменные равны 1, мы тем самым из всех возможных вариантов покрытия каждой единицы на карте Карно выбираем в точности один вариант. Значит, полученная в результате такого выбора ДНФ

для исходной (минимизируемой) СДНФ действительно будет тупиковой.

Но нужно заметить, что перечисление тупиковых ДНФ является самым неприятным и трудоемким этапом всего алгоритма минимизации. Если число единичных клеток карты Карно, не покрываемых ядром, достаточно велико, то функция Патрика будет весьма сложной и ее упрощение сопоставимо по трудоемкости со всем процессом минимизации.

4. Отыскание среди тупиковых ДНФ кратчайших и минимальных. Среди найденных тупиковых ДНФ находят кратчайшие и минимальные. Можно легко показать, что минимальная ДНФ всегда является кратчайшей, но обратное неверно. Так, $x_1x_2 \vee \overline{x}_2 = x_1 \vee \overline{x}_2$ и первая ДНФ кратчайшая, но не минимальная. Действительно, легко сообразить, что вторая из записанных ДНФ минимальна. Следовательно, представляемую ею функцию нельзя представить ДНФ, содержащей менее двух элементарных конъюнкций. Но в первой ДНФ три литерала, а во второй — два. Из пяти тупиковых ДНФ, соответствующих функции Патрика (6.15), кратчайшими являются две. Каждая из них минимальна, так как обе они имеют одинаковое число литералов.

Пример 6.14. Рассмотрим карту Карно на рис. 6.15.

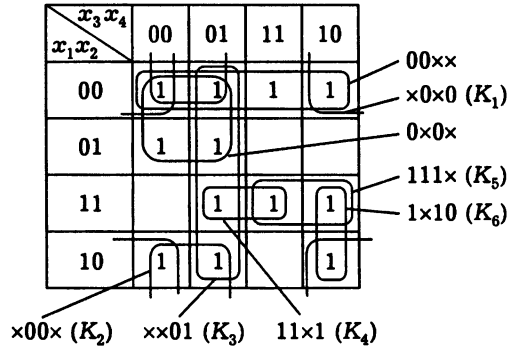


Рис. 6.15

В результате проведения склейки получим следующую сокращенную ДНФ*:

$$\overline{x}_1\overline{x}_3 \vee \overline{x}_1\overline{x}_2 \vee \overline{x}_2\overline{x}_4 \vee \overline{x}_2\overline{x}_3 \vee \overline{x}_3x_4 \vee x_1x_2x_4 \vee x_1x_2x_3 \vee x_1x_3\overline{x}_4.$$

Ядро составляют склейки (простые импликанты) $\overline{x}_1\overline{x}_3$ и $\overline{x}_1\overline{x}_2$.

Шесть клеток, содержащих единицу, на карте Карно остаются непокрытыми ядровыми склейками. Для неядровых склеек (обозначенных $K_1, \ldots, K_6$) составляем функцию Патрика в виде

$$(K_3 \vee K_4)(K_4 \vee K_5)(K_5 \vee K_6)(K_1 \vee K_2)(K_2 \vee K_3)(K_1 \vee K_6).$$

Преобразуя ее аналогично функции (6.13), получаем

$$K_1K_3K_5 \vee K_2K_4K_6 \vee K_2K_3K_5K_6 \vee K_1K_2K_4K_5 \vee K_1K_3K_4K_6.$$

Имеем, следовательно, пять тупиковых ДНФ. Запишем их, для наглядности, так:

$$\underbrace{\overline{x}_1\overline{x}_3 \vee \overline{x}_1\overline{x}_2}_{\text{ядро}} \vee \begin{cases} \overline{x}_2\overline{x}_4 \vee \overline{x}_3x_4 \vee x_1x_2x_3, \\ \overline{x}_2\overline{x}_3 \vee x_1x_2x_4 \vee x_1x_3\overline{x}_4, \\ \overline{x}_2\overline{x}_3 \vee \overline{x}_3x_4 \vee x_1x_2x_3 \vee x_1x_3\overline{x}_4, \\ \overline{x}_2\overline{x}_4 \vee \overline{x}_2\overline{x}_3 \vee x_1x_2x_4 \vee x_1x_2x_3, \\ \overline{x}_2\overline{x}_4 \vee \overline{x}_3x_4 \vee x_1x_2x_4 \vee x_1x_3\overline{x}_4. \end{cases}$$

Из этих пяти тупиковых ДНФ кратчайшими являются первая и вторая. Из них, в свою очередь, минимальной является первая, так как она содержит на один литерал меньше.

В итоге получаем минимальную ДНФ в виде

$$\overline{x}_1\overline{x}_3 \vee \overline{x}_1\overline{x}_2 \vee \overline{x}_2\overline{x}_4 \vee \overline{x}_3x_4 \vee x_1x_2x_3.$$

*Обратим еще раз внимание на то, что каждый выделяемый прямоугольник на карте Карно имеет площадь, равную некоторой степени двойки. Поэтому, например, три соседние единичные клетки не могут быть объединены в один прямоугольник, а их „накроют“ два прямоугольника площадью 2, пересекающиеся по одной клетке.

В данном случае минимальная ДНФ оказалась единственной, хотя, как это мы видели в ранее разобранных примерах, в общем случае могут существовать несколько минимальных ДНФ. #

Техника карт Карно является удобным и наглядным (при определенных ограничениях на число переменных минимизируемой функции) способом реализации алгоритма Квайна — Мак-Клоски. Но существуют и другие способы проведения склейки, т.е. получения сокращенной ДНФ для исходной функции. Одним из таких способов является чисто алгебраический ***метод Блейка***, состоящий в том, что к любой ДНФ, представляющей функцию, применяются следующие тождества:

$$\begin{cases} xK_1 \vee \overline{x}K_2 = xK_1 \vee \overline{x}K_2 \vee K_1K_2, \\ K_1 \vee K_1K_2 = K_1. \end{cases} \tag{6.16}$$

Первое из тождеств (6.16) называют ***тождеством*** (или ***правилом***) ***обобщенного склеивания***, второе — ***тождеством*** (или ***правилом***) ***поглощения***.

„Технология“ использования метода Блейка такова: применяют тождество обобщенного склеивания до тех пор, пока не перестанут появляться новые элементарные конъюнкции (вида K_1K_2). После этого применяют тождество поглощения*.

Как только сокращенная ДНФ тем или иным способом найдена, приступают к нахождению ядра. Ядро можно определить (без использования карты Карно) с помощью так называемой ***таблицы Квайна***. Столбцы этой таблицы соответствуют элементарным конъюнкциям исходной СДНФ, а строки — простым импликантам сокращенной ДНФ. На пересечении строки и столбца проставляется знак „+“ (плюс), если простая импликанта данной строки покрывает элементарную конъюнкцию данного столбца. Ядро вычисляется так: отмечаем столбцы с

*См.: *Гаврилов Г.П., Сапоженко А.А.*

единственным знаком „+“, тогда простые импликанты тех и только тех строк, в которые попал этот знак, образуют ядро.

Для примера 6.13.б (см. рис. 6.14) получим таблицу Квайна, изображенную на рис. 6.16.

	0001	0011	0010	0101	0111	0110	1100	1000	1010
0××1	+	+		+	+				
0×1×		+	+		+	+			
10×0								+	+
×010			+						+
1×00							+	+	
	*			*		*	*		

Рис. 6.16

(В целях экономии места элементарные конъюнкции в таблице заменены цифровыми обозначениями соответствующих вершин и граней булева куба — точно так же как при обозначении прямоугольников на картах Карно. Ядровые импликанты выделены жирным шрифтом.)

По таблице Квайна можно составить и функцию Патрика для перечисления тупиковых ДНФ. Для этого нужно отметить все столбцы таблицы, в которых на пересечении со строками, соответствующими ядровым импликантам, не стоит знак „+“. Для разбираемого примера таковым является только последний столбец. Чтобы покрыть соответствующую элементарную конъюнкцию СДНФ, можно выбрать одну из двух простых импликант: $x_1\overline{x}_2\overline{x}_4$ или $\overline{x}_2x_3\overline{x}_4$.

В заключение рассмотрим очень кратко применение карт Карно к построению минимальных ДНФ ***частичных булевых функций***, т.е. *частичных отображений* из множества $\{0,1\}^n$ в множество $\{0,1\}$.

Частичная булева функция может быть задана посредством карты Карно, в которой кроме клеток с единицами и пустых клеток будут клетки, заполненные прочерками (–). Такой прочерк означает, что на соответствующем наборе функция не определена.

Склейка для частичной функции (заданной картой Карно) проводится таким образом, что выделяются прямоугольники максимальной площади (содержащие 2^k клеток, для некоторого k), каждая клетка которых содержит либо единицу, либо прочерк, причем существует по крайней мере одна единичная клетка.

Пример 6.15. Пусть частичная функция $f(x_1,x_2,x_3)$ задана картой Карно, приведенной на рис. 6.17. Прямоугольник максимальной площади (равной 4), состоящий из единицы и прочерков, записывается как $0\times\times$. Следовательно, минимальная ДНФ для заданной функции будет $\overline{x}_1$. #

x_1 \ x_2x_3	00	01	11	10
0	1	–	–	–
1	–			

0××

Рис. 6.17

По поводу рассмотренного примера возникает такой вопрос: почему не принят во внимание другой прямоугольник (площади 2), содержащий клетку с единицей и клетку с прочерком: $\times 00$? Связано это вот с чем. Перед тем как выделять упомянутые выше прямоугольники, мы на самом деле доопределяем исходную частичную функцию (получая обычную булеву функцию) так, чтобы в максимальном числе клеток, в которых стоят прочерки (но не нули!), появились единицы. Точнее говоря, среди прямоугольников (с прочерками), содержащих данную единицу, выбирают для замены прочерков единицами такой, который имеет максимальную площадь. Прочерки же в остальных прямоугольниках заменяют нулями.

В примере 6.15 мы доопределяем исходную функцию так, что получается функция f_1, задаваемая картой Карно, приве-

x_1 \ x_2x_3	00	01	11	10
0	1	1	1	1
1				

0××

Рис. 6.18

x_1 \ x_2x_3	00	01	11	10
0	1			
1	1			

×00

Рис. 6.19

денной на рис. 6.18. Эта функция имеет минимальную ДНФ $\overline{x}_1$. Следовательно, и частичная исходная функция может быть представлена такой ДНФ, поскольку на всех наборах, на которых она определена, она принимает такое же значение, как и функция f_1.

Конечно, мы могли бы доопределить функцию f по-другому, так, чтобы получилась функция f_2, заданная картой Карно, приведенной на рис. 6.19. Ясно, что $f_2 = \overline{x}_2\overline{x}_3$, поэтому и частичная функция f может быть определена такой ДНФ. Но эта ДНФ не минимальна для данной частичной (именно частичной!) функции, поскольку первый способ доопределения дал ДНФ, содержащую лишь один литерал.

Таким образом, в отличие от минимизации булевых функций при минимизации частичных булевых функций не следует выделять все максимальные прямоугольники с прочерками, содержащие данную единичную клетку карты Карно, достаточно выбрать произвольно любой из таких прямоугольников. Но, конечно, не нужно забывать о том, что каждая единица на карте должна быть покрыта некоторой склейкой.

Пример 6.16. Для карты на рис. 6.20 следует взять обе склейки на четыре позиции: 00×× и ××00, получив для заданной этой картой частичной функции минимальную ДНФ в виде $\overline{x}_1\overline{x}_2 \vee \overline{x}_3\overline{x}_4$.

Заметим, что без использования склеек с прочерками мы вообще не могли бы минимизировать данную функцию. Нуж-

но также отметить, что не всегда использование „частичности" функции позволяет получить минимальную ДНФ для нее. Так, на представленной на рис. 6.20 карте в случае, если мы переместим нижнюю единицу на строку выше, обычная склейка на две позиции дает лучший результат: $\overline{x}_1\overline{x}_3\overline{x}_4$, а записанная выше ДНФ уже не будет минимальной (и даже кратчайшей).

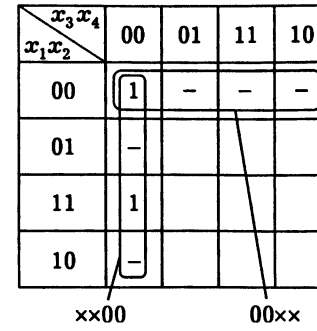


Рис. 6.20

6.7. Теорема Поста

В силу теоремы о *представлении* любой *булевой функции дизъюнктивной* или *конъюнктивной нормальной формой стандартный базис* $\{\vee, \cdot, \overline{\ }\}$ является *полным множеством.* Поскольку, согласно *законам де Моргана*, можно выразить *конъюнкцию* через *дизъюнкцию* и *отрицание*, равно как и дизъюнкцию можно выразить через конъюнкцию и отрицание, то при удалении из стандартного базиса одной функции, дизъюнкции или конъюнкции, при сохранении отрицания, получим снова полное множество.

Прежде чем рассматривать другие примеры полных множеств, установим один важный факт.

Теорема 6.3. Пусть F и G — некоторые множества булевых функций, причем F — полное множество. Тогда, если каждая функция из F может быть представлена некоторой *формулой над множеством* G, то G — полное множество.

◀ Из условия теоремы следует, что каждая функция $f \in F$ может быть представлена некоторой формулой Ψ над G, т.е.

$$f(x_1, \ldots, x_n) = \Psi(x_1, \ldots, x_n). \tag{6.17}$$

Докажем, что всякая формула над F *эквивалентна* некоторой формуле над G, т.е. всякая функция φ, представляемая формулой над F, может быть представлена также и некоторой формулой над G. Доказательство проведем по такой схеме: сначала убедимся в справедливости утверждения для „базисных“ формул, т.е. для переменных и констант из F, а затем в предположении, что оно уже доказано для формул $\Phi_1, \ldots, \Phi_n$, где $n \geqslant 1$, докажем его для любой формулы вида $f(\Phi_1, \ldots, \Phi_n)$, где $f \in F$. Такой метод доказательства называют доказательством индукцией по построению формулы.

Пусть φ — какая-то формула над F. Если $\varphi = x$, где x — *булево переменное* из множества X, то, поскольку каждое переменное есть, по определению, и формула над G, функция φ представляется формулой над G.

Если φ есть константа из F, то представляющая φ формула над G существует ввиду (6.17).

Рассмотрим формулу над F вида $f(\Phi_1, \ldots, \Phi_n)$, где $n > 0$, $f \in F$, а $\Phi_1, \ldots, \Phi_n$ — формулы над F. Согласно предположению индукции, каждая формула Φ_i, $i = \overline{1, n}$, может быть заменена эквивалентной ей формулой Θ_i над G (т.е. имеет место *тождество* $\Phi_i = \Theta_i$ над $F \cup G$). Тогда, используя правила *эквивалентных преобразований* формул (см. теорему 6.1), получим тождество

$$f(\Phi_1, \ldots, \Phi_n) = f(\Theta_1, \ldots, \Theta_n),$$

а в соответствии с (6.17) будем иметь

$$f(\Theta_1, \ldots, \Theta_n) = \Psi(\Theta_1, \ldots, \Theta_n). \tag{6.18}$$

Правая часть тождества (6.18) и есть формула над G, эквивалентная исходной формуле над F.

Поскольку в силу полноты множества F любая булева функция может быть представлена некоторой формулой над F, а любая такая формула, как мы только что доказали, эквивалентна некоторой формуле над G, то любая булева функция

может быть представлена некоторой формулой над G, что и доказывает полноту множества G. ▶

Пример 6.17. Рассмотрим *базис Жегалкина* $\{\oplus, \cdot, 1\}$. Чтобы доказать полноту этого множества, заметим, что

$$x \vee y = x \cdot y \oplus x \oplus y, \quad \overline{x} = x \oplus 1,$$

т.е. каждый элемент стандартного базиса может быть представлен формулой над базисом Жегалкина. Отсюда и следует (ввиду полноты стандартного базиса и теоремы 6.3) полнота базиса Жегалкина. #

Любую формулу над базисом Жегалкина называют ***полиномом Жегалкина***. Полином Жегалкина от n переменных может быть записан в виде

$$P(x_1, \ldots, x_n) = \sum_{\{i_1, i_2, \ldots, i_m\} \subseteq \{1,2,\ldots,n\}} (\mathrm{mod}\, 2)\, a_{i_1 i_2 \ldots i_m} x_{i_1} x_{i_2} \ldots x_{i_m},$$

где коэффициенты полинома $a_{i_1 i_2 \ldots i_m} \in \{0, 1\}$ индексированы всеми возможными подмножествами множества $\{1, 2, \ldots, n\}$ (коэффициент a_0 соответствует пустому множеству). В частности, при $n = 3$ будем иметь:

$$a_{123} x_1 x_2 x_3 \oplus a_{12} x_1 x_2 \oplus a_{13} x_1 x_3 \oplus \\ \oplus a_{23} x_2 x_3 \oplus a_1 x_1 \oplus a_2 x_2 \oplus a_3 x_3 \oplus a_0 \quad (6.19)$$

(общий вид полинома Жегалкина от трех переменных). Формула вида

$$\sum_{i=1}^{n} (\mathrm{mod}\, 2)\, a_i x_i \oplus a_0 \quad (6.20)$$

называется ***полиномом Жегалкина первой степени*** от переменных. В таком полиноме отсутствуют „нелинейные“ слагаемые, т.е. все коэффициенты, индексированные более чем одноэлементными подмножествами, равны 0 (и вместе с ними равны 0 все слагаемые, содержащие конъюнкции переменных).

Можно доказать следующее достаточно простое, но важное утверждение.

Теорема 6.4. Полином Жегалкина для любой булевой функции определен однозначно.

Для функций от небольшого числа переменных (не превышающего 4) можно использовать ***метод неопределенных коэффициентов***, позволяющий получить полином Жегалкина данной функции. Проиллюстрируем этот метод на примере.

Пример 6.18. Пусть *вектор значений булевой функции* f равен 11001011. Найдем полином Жегалкина, представляющий f. Поскольку размерность вектора значений f равна $2^3 = 8$, то f задана как функция от трех переменных. Тогда она представляется некоторым полиномом Жегалкина третьей степени, общий вид которого дает формула (6.19). Наша задача — найти такие значения коэффициентов этого полинома, при которых он представляет функцию f. Ясно, что значение функции f на наборе 000 равно коэффициенту a_0 в формуле (6.19). Но, согласно заданному вектору значений, оно равно 1. Следовательно, $a_0 = 1$. Далее,

$$f(0,0,1) = a_3 \oplus a_0 = a_3 \oplus 1 = 1,$$

откуда, решая уравнение относительно a_3 в *поле* $\mathbb{Z}_2$, получим $a_3 = 0$;

$$f(0,1,0) = a_2 \oplus 1 = 0,$$

т.е. $a_2 = 1$;

$$f(1,0,0) = a_1 \oplus 1 = 1,$$

и $a_1 = 0$. Чтобы найти коэффициенты a_{12}, a_{13} и a_{23}, нужно рассмотреть значения функции на наборах 110, 101 и 011 соответственно. Так, для первого набора получим

$$f(1,1,0) = a_{12}x_1x_2 \oplus a_1x_1 \oplus a_2x_2 \oplus a_0 = \\ = a_{12} \oplus a_2 \oplus a_0 = a_{12} \oplus 1 \oplus 1 = a_{12}$$

(сумма по модулю 2 любого четного числа равных слагаемых равна 0). Поскольку в то же время $f(1,1,0)=1$, то $a_{12}=1$. Аналогично

$$f(1,0,1)=a_{13}\oplus a_0=0,$$

откуда $a_{13}=1$;

$$f(0,1,1)=a_{23}\oplus a_2\oplus a_0=0,$$

и, так как $a_2=a_0=1$, $a_{23}=0$. Наконец,

$$f(1,1,1)=a_{123}\oplus a_{12}\oplus a_{13}\oplus a_2\oplus a_0=a_{123}=1.$$

Итак,

$$f=x_1x_2x_3\oplus x_1x_2\oplus x_1x_3\oplus x_2\oplus 1. \quad \#$$

Как видно из примера 6.18, суть метода неопределенных коэффициентов состоит в следующем. Записывая полином Жегалкина сначала в общем виде, с неопределенными коэффициентами, мы выражаем значение полинома на фиксированных наборах через коэффициенты и приравниваем его заданному значению функции. Начинаем с нулевого набора и находим коэффициент a_0, равный значению заданной функции на *нулевом наборе*. Зная его, из рассмотрения значений фукнции на наборах, содержащих в точности одну единицу, находим коэффициенты a_i при „одиночных“ переменных (коэффициенты „линейной части“ полинома). Зная их, из рассмотрения значений функции на наборах, содержащих в точности две единицы, находим коэффициенты при конъюнкциях двух переменных и т.д. При этом выполняются вычисления и решаются простейшие линейные уравнения в поле вычетов по модулю 2.

Пример 6.19. а. Рассмотрим множество $\{|\}$, состоящее из единственной функции (*штриха Шеффера*). Полнота этого множества следует из легко проверяемых тождеств

$$x\cdot y=(x|y)|(x|y), \qquad \overline{x}=(x|x).$$

б. Полнота множества $\{\downarrow\}$, единственным элементом которого является *стрелка Пирса*, проверяется аналогично. #

Теперь мы сформулируем и докажем критерий (необходимое и достаточное условие) полноты для произвольного множества булевых функций. Для этого нам потребуется сначала рассмотреть некоторые специальные множества функций.

Определение 6.8. Функцию f называют ***функцией, сохраняющей константу*** **0** (соответственно ***константу*** **1**), если $f(\widetilde{0}) = 0$ (соответственно: $f(\widetilde{1}) = 1$), где $\widetilde{0}$ — нулевой, а $\widetilde{1}$ — *единичный наборы значений переменных функции* f.

Например, *мажоритарная функция* является функцией, сохраняющей и константу 0, и константу 1. Отрицание не сохраняет ни 0, ни 1, а *эквивалентность* сохраняет 1, но не сохраняет 0. Множество всех функций, сохраняющих константу 0 (константу 1), обозначается T_0 (соответственно T_1).

Наборы $\widetilde{\alpha}$ и $\overline{\widetilde{\alpha}}$ из *булева куба* $\mathbb{B}^n = \{0,1\}^n$ (для произвольного фиксированного n) будем называть ***взаимно противоположными***, говоря при этом также, что набор $\overline{\widetilde{\alpha}}$ есть ***инверсия*** (или ***отрицание***) ***набора*** $\widetilde{\alpha}$ (в силу единственности *дополнения* любого элемента *булевой алгебры* набор $\widetilde{\alpha}$ будет, очевидно, инверсией набора $\overline{\widetilde{\alpha}}$).

Определение 6.9. Функцию $g \in \mathcal{P}_{2,n}$ называют ***двойственной к функции*** $f \in \mathcal{P}_{2,n}$, если для всякого $\widetilde{\alpha} \in \{0,1\}^n$ $(n > 0)$ имеет место

$$g(\widetilde{\alpha}) = \overline{f}(\overline{\widetilde{\alpha}}).$$

Полагаем также, что константа 0 является двойственной к константе 1 и наоборот.

Пример 6.20. а. Стрелка Пирса есть функция, двойственная к штриху Шеффера, так как

$$x \downarrow y = \overline{x \vee y} = \overline{\overline{\overline{x} \cdot \overline{y}}} = \overline{x|y}.$$

6. Сумма по модулю 2 двойственна к *эквивалентности*, так как

$$x \sim y = \overline{x \oplus y} = \overline{\overline{x} \oplus \overline{y}}.$$

Эти две функции являются и отрицанием друг друга, но неверно в общем случае, что функция g, будучи отрицанием функции f, двойственна к f: штрих Шеффера не есть функция, двойственная к конъюнкции, а стрелка Пирса не есть функция, двойственная к дизъюнкции, но конъюнкция двойственна к дизъюнкции и наоборот, а стрелка Пирса двойственна к штриху Шеффера и наоборот. #

В общем случае в силу уже упомянутого свойства единственности дополнения в булевой алгебре функция h, двойственная к функции g, которая двойственна к f, *равна* f.

Определение 6.10. Функцию $f \in \mathcal{P}_{2,n}$ называют ***самодвойственной***, если она двойственна к себе самой, т.е.

$$(\forall \widetilde{\alpha} \in \{0,1\}^n)(f(\widetilde{\alpha}) = \overline{f}(\overline{\widetilde{\alpha}})),$$

или

$$(\forall \widetilde{\alpha} \in \{0,1\}^n)(f(\overline{\widetilde{\alpha}}) = \overline{f}(\widetilde{\alpha})).$$

Таким образом, функция самодвойственна тогда и только тогда, когда на взаимно противоположных наборах она принимает взаимно противоположные значения. Следовательно, для того чтобы убедиться в несамодвойственности заданной функции f, достаточно найти хотя бы одну пару взаимно противоположных наборов $\widetilde{\alpha}$ и $\overline{\widetilde{\alpha}}$, таких, что значения функции на них совпадают, т.е. $f(\widetilde{\alpha}) = f(\overline{\widetilde{\alpha}})$. Так, мажоритарная функция является самодвойственной, а эквивалентность — нет, поскольку при $\widetilde{\alpha} = (0,\ 0)$ $0 \sim 0 = 1$ и $1 \sim 1 = 1$.

Множество всех самодвойственных функций (при всех $n \geqslant 1$) обозначим S.

Определение 6.11. Функцию $f \in \mathcal{P}_{2,n}$ называют ***монотонной***, если для любых наборов $\widetilde{\alpha}$, $\widetilde{\beta} \in \mathbb{B}^n$, таких, что $\widetilde{\alpha} \leqslant \widetilde{\beta}$, имеет место $f(\widetilde{\alpha}) \leqslant f(\widetilde{\beta})$.

Другими словами, функция монотонна тогда и только тогда, когда для любого набора $\widetilde{\alpha}$ имеет место следующее свойство: если значение функции на наборе $\widetilde{\alpha}$ равно 1, то оно равно 1 и на всех наборах, *строго больших* (по отношению *булева порядка* на $\mathbb{B}^n$) набора $\widetilde{\alpha}$. Любой *минимальный* (относительно того же порядка) набор $\widetilde{\alpha}$, для которого значение $f(\widetilde{\alpha})$ монотонной функции f равно 1, называют ***нижней единицей функции*** f. Очевидно, что *вектор значений* монотонной *булевой функции* полностью определяется множеством ее нижних единиц*. Мажоритарная функция монотонна, и множество ее нижних единиц есть $\{011, 101, 110\}$. Штрих Шеффера — немонотонная функция, так как $00 < 11$, но $0|0 = 1$, а $1|1 = 0$. Множество всех монотонных функций принято обозначать через M.

Определение 6.12. Функцию $f \in \mathcal{P}_{2,n}$ называют ***линейной***, если она может быть представлена полиномом Жегалкина первой степени от n переменных, т.е. формулой вида (6.20).

Множество всех линейных функций принято обозначать через L.

Любая *булева константа* и любая *проектирующая функция* x являются линейными функциями. Такова, разумеется, сумма по модулю 2. Отрицание также линейно, ибо $\overline{x} = x \oplus 1$. Конъюнкция и дизъюнкция не являются линейными функциями, так как не могут быть представлены полиномом Жегалкина первой степени (см. теорему 6.4).

Определение 6.13. Множества функций T_0, T_1, S, M, L называются ***классами Поста***.

Замечание 6.9. Каждый класс Поста состоит из функций с соответствующим свойством для любого числа переменных.

*Нетрудно понять, что множество нижних единиц монотонной функции f есть множество всех минимальных элементов множества C_f^1 — *конституент единицы* функции f.

Можно доказать также, что если функция f принадлежит какому-то классу Поста C, то и любая функция, равная функции f, принадлежит этому же классу. Другими словами, добавление или удаление *фиктивных переменных* не выводит за пределы любого из классов Поста.

Полезно еще заметить, что любая проектирующая функция x принадлежит одновременно всем пяти классам Поста. Действительно, если $f(x)=x$, то $f(0)=0$ и $f(1)=1$, т.е. $f\in T_0\cap T_1$. Отсюда же вытекает и самодвойственность функции x. Монотонность следует из того, что $0<1$ и $f(0)=0<f(1)=1$. Линейность очевидна.

В то же время существуют функции, не принадлежащие ни одному из классов Поста. Таков, например, штрих Шеффера. Все свойства, кроме нелинейности, следуют прямо из таблицы этой функции. Нелинейность же доказывается выводом полинома Жегалкина для штриха Шеффера:

$$x|y=\overline{x\cdot y}=x\cdot y\oplus 1,$$

что не есть полином Жегалкина первой степени. #

Фундаментальным свойством каждого класса Поста является его *замкнутость* (в смысле определения 6.3). Это означает для любого из классов Поста C, что всякая *суперпозиция над* C снова есть элемент C.

Теорема 6.5. Каждый класс Поста замкнут.

◀ Нужно для каждого класса Поста $C\in\{T_0,T_1,S,M,L\}$ доказать, что *замыкание* $[C]$ *множества булевых функций* C совпадает с C. Пусть $f(g_1,\ldots,g_n)$ — какая-то суперпозиция над C. Обозначим ее через φ. Без ограничения общности можно считать, что все функции f, g_1, …, $g_n\in\mathcal{P}_{2,n}$ (для некоторого n).

Рассуждаем, используя индукцию по определению суперпозиции. Если для каждого $i=\overline{1,n}$ функция $g_i=x_i$, где x_i — переменное, то $\varphi=f(x_1,\ldots,x_n)\in C$. Предположим, что в суперпозиции φ все функции g_i есть элементы класса Поста C

(в частности, это может быть и соответствующая проектирующая функция, которая ввиду замечания 6.9 принадлежит всем классам Поста). Докажем, что и $\varphi = f(g_1,\dots,g_n) \in C$.

1. Если $C = T_0$, то $\varphi(\widetilde{0}) = f(g_1(\widetilde{0}),\dots,g_n(\widetilde{0})) = f(0,\dots,0) = 0$, так как f, g_1, ..., $g_n \in T_0$. Следовательно, $\varphi \in T_0$.

2. При $C = T_1$ рассуждаем точно так же.

3. Пусть $C = S$. Фиксируем произвольно набор $\widetilde{\alpha} \in \{0,1\}^n$. Вычислим (используя самодвойственность всех функций):

$$\varphi(\overline{\widetilde{\alpha}}) = f(g_1(\overline{\widetilde{\alpha}}),\dots,g_n(\overline{\widetilde{\alpha}})) = f(\overline{g_1}(\widetilde{\alpha}),\dots,\overline{g_n}(\widetilde{\alpha})) = \\ = \overline{f}(g_1(\widetilde{\alpha}),\dots,g_n(\widetilde{\alpha})) = \overline{\varphi}(\widetilde{\alpha}).$$

Следовательно, $\varphi \in S$.

4. $C = M$. Берем произвольно наборы $\widetilde{\alpha}$ и $\widetilde{\beta}$ так, что $\widetilde{\alpha} \leqslant \widetilde{\beta}$. Докажем, что $\varphi \in M$. Имеем

$$\varphi(\widetilde{\alpha}) = f(g_1(\widetilde{\alpha}), \quad \dots, \quad g_n(\widetilde{\alpha})) \leqslant f(g_1(\widetilde{\beta}), \quad \dots, \quad g_n(\widetilde{\beta})) = \varphi(\widetilde{\beta}),$$

так как все функции g_i, $i = \overline{1,n}$, монотонны и тем самым вектор $(g_1(\widetilde{\alpha}),\dots,g_n(\widetilde{\alpha}))$ не больше вектора $(g_1(\widetilde{\beta}),\dots,g_n(\widetilde{\beta}))$, а функция f также монотонна. Тогда ясно, что $\varphi \in M$.

5. Если же $C = L$, то очевидно, что при подстановке в линейную функцию (полином Жегалкина первой степени) вместо ее переменных произвольных линейных функций получится снова линейная функция. Итак, мы доказали замкнутость каждого класса Поста. ▶

Докажем теперь теорему, характеризующую одно важное свойство немонотонных функций.

Теорема 6.6. Если функция f не является монотонной, т.е. $f \notin M$, то найдутся два таких набора $\widetilde{\alpha}$, $\widetilde{\beta}$, что

$$\widetilde{\alpha} = (\alpha_1, \dots, \alpha_{i-1}, 0, \alpha_{i+1}, \dots, \alpha_n),$$
$$\widetilde{\beta} = (\alpha_1, \dots, \alpha_{i-1}, 1, \alpha_{i+1}, \dots, \alpha_n),$$

и $f(\widetilde{\alpha})=1$, $f(\widetilde{\beta})=0$, т.е. эти два набора различаются значениями в точности одной компоненты, а значение функции равно 0 на большем наборе и равно 1 на меньшем.

◀ Так как функция f не является монотонной, то найдутся такие два набора $\widetilde{\gamma}$ и $\widetilde{\delta}$, что $\widetilde{\gamma}<\widetilde{\delta}$, но $f(\widetilde{\gamma})=1$, $f(\widetilde{\delta})=0$. Строгое неравенство $\widetilde{\gamma}<\widetilde{\delta}$ означает, что найдутся такие номера (не меньше одного) $1\leqslant i_1<i_2<\ldots<i_k\leqslant n$, что

$$\widetilde{\gamma}=(\gamma_1,\ldots,\gamma_{i_1-1},0,\gamma_{i_1+1},\ldots,\gamma_{i_2-1},0,\ldots,\gamma_{i_2+1},\ldots,\gamma_{i_k-1},0,\gamma_{i_k+1},\ldots,\gamma_n),$$

а

$$\widetilde{\delta}=(\gamma_1,\ldots,\gamma_{i_1-1},1,\gamma_{i_1+1},\ldots,\gamma_{i_2-1},1,\ldots,\gamma_{i_2+1},\ldots,\gamma_{i_k-1},1,\gamma_{i_k+1},\ldots,\gamma_n),$$

т.е. все компоненты наборов, кроме выделенных, с номерами i_1, i_2, ..., i_k соответственно совпадают, а все компоненты с выделенными номерами у меньшего набора равны 0, а у большего равны 1. Построим монотонно возрастающую последовательность наборов $\widetilde{\gamma}=\widetilde{\gamma}_0<\widetilde{\gamma}_1<\widetilde{\gamma}_2<\ldots<\widetilde{\gamma}_k=\widetilde{\delta}$ так, что для каждого $s\in\{1,2,\ldots,k\}$ набор $\widetilde{\gamma}_s$ получается из набора $\widetilde{\gamma}_{s-1}$ заменой нулевого значения компоненты с номером i_s единичным. Поскольку $f(\widetilde{\gamma})=1$, а $f(\widetilde{\delta})=0$, то обязательно найдется такое $s\in\{1,2,\ldots,k\}$, что $f(\widetilde{\gamma}_{s-1})=1$, а $f(\widetilde{\gamma}_s)=0$ (на наборе $\widetilde{\gamma}_{s-1}$ значение функции еще равно 1, а на наборе $\widetilde{\gamma}_s$ оно уже равно 0). Полагая $\widetilde{\alpha}=\widetilde{\gamma}_{s-1}$ и $\widetilde{\beta}=\widetilde{\gamma}_s$, получим доказываемое. ▶

Теорема 6.7 (*критерий Поста*). Множество F булевых функций полно тогда и только тогда, когда оно не содержится целиком ни в одном из классов Поста.

◀ Необходимость условия теоремы доказывается просто: если бы для некоторого класса Поста C выполнялось $F\subseteq C$, то всякая суперпозиция над F, согласно теореме 6.5, снова лежала бы в C. Между тем существуют функции, не содержащиеся ни в

одном из классов Поста, например штрих Шеффера (см. пример 6.9). Таким образом, нашлась бы функция, которую нельзя представить в виде суперпозиции над F, что противоречит предположению о полноте F.

Переходим к доказательству достаточности условия теоремы. Для доказательства полноты множества F, удовлетворяющего условию теоремы, построим формулы над F для функций отрицания и конъюнкции, поскольку, как было замечено выше, множество, образованное этими функциями, полно. Тогда в силу теоремы 6.3 будет полным и множество F.

Для немонотонной функции $f_M \in F \setminus M$, согласно теореме 6.6, найдутся два таких набора $\widetilde{\alpha}$ и $\widetilde{\beta}$, что

$$\widetilde{\alpha} = (\alpha_1, \ldots, \alpha_{i-1}, 0, \alpha_{i+1}, \ldots, \alpha_n),$$
$$\widetilde{\beta} = (\alpha_1, \ldots, \alpha_{i-1}, 1, \alpha_{i+1}, \ldots, \alpha_n),$$

$f_M(\widetilde{\alpha}) = 1$, а $f_M(\widetilde{\beta}) = 0$.

Тогда $\overline{x} = f_M(\alpha_1, \ldots, \alpha_{i-1}, x, \alpha_{i+1}, \ldots, \alpha_n)$, т.е. отрицание может быть получено из немонотонной функции подстановкой вместо некоторого ее переменного x_i переменного x, а вместо остальных переменных констант $\alpha_1, \ldots, \alpha_{i-1}, \alpha_{i+1}, \ldots, \alpha_n$, определяемых выбранными выше наборами $\widetilde{\alpha}$ и $\widetilde{\beta}$. Но, вообще говоря, система F может и не содержать констант. Поэтому константы 0 и 1 необходимо представить некоторыми формулами над F.

Рассмотрим два случая.

Первый случай. В F существует функция f_0, не сохраняющая константу 0, которая сохраняет константу 1; или существует функция f_1, не сохраняющая константу 1, но сохраняющая константу 0.

Если существует функция $f_0 \notin T_0$ и $f_0 \in T_1$, то константа 1 представляется формулой

$$1 = f_0(x, \ldots, x),$$

так как $f_0(0,\ldots,0)=1$ и $f_0(1,\ldots,1)=1$. Чтобы выразить константу 0, используем любую функцию $g\in F$, не сохраняющую константу 1:

$$0=g(1,\ldots,1)=g(f_0(x,\ldots,x),\ldots,f_0(x,\ldots,x)).$$

Если же указанная функция f_0 не существует, но найдется функция $f_1\in T_0\setminus T_1$, то константы представляем формулами над F аналогично (двойственным образом).

Второй случай. Любая функция из F, не сохраняющая константу 0, не сохраняет и константу 1, и, наоборот, любая функция из F, не сохраняющая константу 1, не сохраняет и константу 0.

Пусть $f_0(0,\ldots,0)=1$, а $f_0(1,\ldots,1)=0$. Тогда мы получаем возможность представить отрицание следующей формулой:

$$\overline{x}=f_0(x,\ldots,x).$$

Переходим к построению формул для констант, так как они потребуются нам далее при построении формулы для конъюнкции. Чтобы представить константы формулами над F, воспользуемся несамодвойственной функцией f_S из F. Для этой функции найдется такой набор $\widetilde{\alpha}$, что $f_S(\widetilde{\widetilde{\alpha}})=f_S(\widetilde{\alpha})$. Введем функцию от одного переменного $h(x)=f_S(x^{\alpha_1},\ldots,x^{\alpha_n})$ (в предположении, что $\widetilde{\alpha}=(\alpha_1,\ \ldots,\ \alpha_n)$). Легко видеть, что $h(0)=$ $=f_S(\widetilde{\widetilde{\alpha}})=f_S(\widetilde{\alpha})=h(1)$, так как для любого $\sigma\in\{0,1\}$

$$0^\sigma=\begin{cases}1, & \sigma=0;\\ 0, & \sigma=1,\end{cases}\qquad 1^\sigma=\begin{cases}1, & \sigma=1;\\ 0, & \sigma=0.\end{cases}$$

Итак, значение $h(x)$ есть константа. Если она равна 0, то константу 1 представляем через функцию, не сохраняющую константу 0; если же $h(x)\equiv 1$, то константу 0 представляем через функцию, не сохраняющую константу 1.

Опишем построение формулы для конъюнкции. Берем нелинейную функцию f_L из F. В полиноме Жегалкина для этой

функции выберем произвольное нелинейное слагаемое, содержащее наименьшее число переменных; пусть это будет слагаемое $x_{i_1}, \ldots, x_{i_k}$ при $2 \leqslant k \leqslant n$. Вместо каждого переменного x_m функции f_L, где $m \notin \{i_1, \ldots, i_k\}$, подставим константу 0, т.е. заменим нулем все переменные, которые не вошли в выбранную ранее конъюнкцию. Получим „редуцированную“ функцию

$$f'_L = x_{i_1} \ldots x_{i_k} \oplus a_{i_1} x_{i_1} \oplus \ldots \oplus a_{i_k} x_{i_k} \oplus a_0 = \\ = f_L(0, \ldots, 0, x_{i_1}, 0, \ldots, 0, x_{i_k}, 0, \ldots, 0).$$

Заметим, что коль скоро мы уже представили константы формулами над F, то функция f'_L тоже представлена формулой над F. Разобьем теперь множество переменных $\{x_{i_1}, \ldots, x_{i_k}\}$ произвольно на две части: $\{x_{i_1}, \ldots, x_{i_m}\}$ и $\{x_{i_{m+1}}, \ldots, x_{i_k}\}$, где $1 \leqslant m \leqslant k-1$, и заменим все переменные первой части переменным x, а переменные второй части — переменным y. В результате получим функцию от двух переменных

$$\chi(x, y) = xy \oplus ax \oplus by \oplus c,$$

где $a = a_{i_1} \oplus \ldots \oplus a_{i_m}$, $b = a_{i_{m+1}} \oplus \ldots \oplus a_{i_k}$, $c = a_0$.

Ясно также, что функция χ может быть представлена такой формулой над F:

$$\chi(x, y) = f_L(0, \ldots, 0, \underbrace{x}_{i_1}, 0, \ldots, 0, \underbrace{x}_{i_m}, 0, \ldots, 0, \\ \underbrace{y}_{i_{m+1}}, 0, \ldots, 0, \underbrace{y}_{i_k}, 0, \ldots, 0),$$

т.е. функция χ получена из нелинейной функции $f_L \in F$ путем подстановки на место ее переменных с номерами $i_1, \ldots, i_m$ переменного x, на место переменных с номерами $i_{m+1}, \ldots, i_k$ — переменного y, а на место всех остальных переменных — константы 0, уже представленной формулой над F (см. выше).

Определим функцию

$$\psi(x, y) = \chi(x \oplus b, y \oplus a) \oplus ab \oplus c.$$

Выразив эту функцию из полинома Жегалкина для χ, получим

$$\begin{aligned}\psi(x,y) &= \chi(x\oplus b, y\oplus a)\oplus ab\oplus c = \\ &= (x\oplus b)(y\oplus a)\oplus a(x\oplus b)\oplus b(y\oplus a)\oplus c\oplus ab\oplus c = \\ &= xy\oplus ax\oplus by\oplus ab\oplus ax\oplus ab\oplus by\oplus ab\oplus c\oplus ab\oplus c = xy,\end{aligned}$$

поскольку сумма по модулю 2 любого четного числа равных слагаемых равна 0. Итак, функция ψ и есть конъюнкция. Так как прибавление к любой функции константы по модулю 2 есть либо сама исходная функция, либо ее отрицание, а отрицание уже представлено формулой над базисом F, то тем самым и конъюнкция представлена такой формулой. ▶

Чтобы исследовать полноту конкретного множества функций $F = \{f_1, f_2, \ldots, f_n\}$, нужно построить так называемую ***критериальную таблицу*** (табл. 6.6).

Таблица 6.6

	T_0	T_1	S	M	L
f_1					
f_2					
$\vdots$					
f_n					

Строки таблицы соответствуют функциям исследуемого множества, а столбцы — классам Поста. В результате анализа функций множества F клетки таблицы заполняются: если функция f_i принадлежит некоторому классу Поста C, то в соответствующей клетке критериальной таблицы ставится знак „+“ (плюс), а если нет, то — знак „−“ (минус). Множество F тогда полно, если и только если в каждом столбце таблицы стоит хотя бы один знак „−“.

Пример 6.21. а. Пусть $F = \{\sim, \vee, 0\}$. Ниже приведены результаты исследования элементов этого множества на принадлежность к классам Поста, а также заполненная критериальная таблица (табл. 6.7).

Таблица 6.7

	T_0	T_1	S	M	L
$\sim$	−	+	−	−	+
$\vee$	+	+	−	+	−
0	+	−	−	+	+

При этом $1 = x \sim x$, так как функция $\sim$ (эквивалентность) не сохраняет константу 0, но сохраняет константу 1 (т.е. имеет место

первый случай из доказательства достаточности условия теоремы Поста). Константа 0 принадлежит самому множеству F. Поскольку $\overline{x} = x \sim 0$, то ввиду полноты множества $\{\vee, \overline{\ }\}$ будет полным и рассматривамое множество. Конъюнкцию можно представить формулой над F, следуя доказательтву теоремы Поста. Берем единственную нелинейную функцию данного множества, дизъюнкцию, и записываем для нее полином Жегалкина:

$$x_1 \vee x_2 = x_1 \cdot x_2 \oplus x_1 \oplus x_2.$$

Видно, что этот полином есть не что иное, как функция $\chi(x_1, x_2)$ при $a = b = 1$ и $c = 0$. Следовательно,

$$x_1 \cdot x_2 = \chi(x_1 \oplus 1, x_2 \oplus 1) \oplus 1.$$

Но так как $\overline{x} = x \sim 0$, то $x_1 \cdot x_2 = ((x_1 \sim 0) \vee (x_2 \sim 0)) \sim 0$. Этот же результат (в данном конкретном случае) можно получить и гораздо проще:

$$x_1 \cdot x_2 = \overline{\overline{x}_1 \vee \overline{x}_2} = ((x_1 \sim 0) \vee (x_2 \sim 0)) \sim 0.$$

Итак, исходное множество является полным.

Заметим, что это полное множество двойственно к базису Жегалкина в том смысле, что каждая из его функций двойственна к соответствующей функции базиса Жегалкина: эквивалентность двойственна к сумме по модулю 2, дизъюнкция — к конъюнкции, константа 0 — к константе 1. Полезно заметить также, что никакое собственное подмножество заданного множества не будет полным.

б. Функция f_1 задана *картой Карно* (рис. 6.21), а вектор значений функции f_2 есть $(0,1,0,0)$.

Для функции f_2 очень просто находятся ДНФ и полином Жегалкина:

$$f_2 = \overline{x}_1 x_2 = (x_1 \oplus 1) x_2 = x_1 x_2 \oplus x_2,$$

откуда следует нелинейность функции f_2. Легко показать также, что эта функция принадлежит лишь классу T_0.

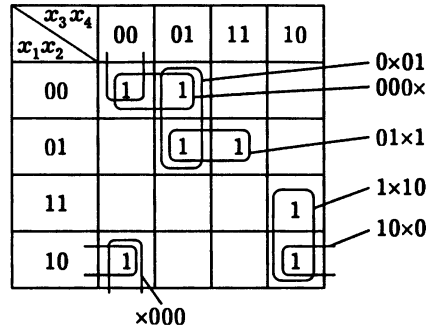


Рис. 6.21

Так как $f_1(0,0,0,0)=1$, а $f_1(1,1,1,1)=0$, то функция f_1 не сохраняет ни константу 0, ни константу 1. Далее, эта функция несамодвойственна, поскольку $f_1(0,1,1,1)=f_1(1,0,0,0)=1$; немонотонность ее следует из того, что, например, $f_1(0,0,0,0)=1$, но $f_1(0,1,0,0)=0$. Вопрос о нелинейности функции f_1 оставим пока открытым, так как даже из частично заполненной критериальной таблицы (табл. 6.8) вытекает, что множество $\{f_1, f_2\}$ полно.

Таблица 6.8

	T_0	T_1	S	M	L
f_1	−	−	−	−	?
f_2	+	−	−	−	−

Формулы для отрицания и констант находятся легко:

$$\overline{x} = f_1(x,x,x,x),$$
$$0 = f_2(x,x),$$
$$1 = f_1(f_2(x,x), f_2(x,x), f_2(x,x), f_2(x,x)).$$

Формулу для конъюнкции проще всего построить прямо по ДНФ для функции f_2:

$$x_1 \cdot x_2 = f_2(\overline{x}_1, x_2) = f_2(f_1(x_1,x_1,x_1,x_1), x_2).$$

Вернемся теперь к отложенному вопросу о нелинейности функции f_1. По приведенной на рис. 6.21 карте Карно можно построить одну из минимальных ДНФ, представляющих эту функцию в виде

$$f_1 = \overline{x}_1 x_2 x_4 \vee x_1 x_3 \overline{x}_4 \vee \overline{x}_1 \overline{x}_2 \overline{x}_3 \vee x_1 \overline{x}_2 \overline{x}_4.$$

Подставляя в последнюю формулу 0 вместо x_1, x вместо x_2, 1 вместо x_3 и y вместо x_4, получаем

$$x \cdot y = f_1(0, x, 1, y),$$

что доказывает нелинейность функции f_1 и одновременно полноту множества $\{f_1\}$.

Константы же можно представить формулами над базисом $\{f_1\}$, используя несамодвойственность функции f_1 (см. разбор второго случая в доказательстве достаточности условия теоремы Поста): мы уже видели, что $f_1(0,1,1,1) = f_1(1,0,0,0) = 1$, т.е. набор $\widetilde{\alpha}$ из упомянутого места в доказательстве теоремы Поста есть 0111, и тогда функция h, фигурирующая в том же месте доказательства и равная в данном случае константе 1, будет иметь вид

$$h(x) = f_1(\overline{x}, x, x, x).$$

Но так как $\overline{x} = f_1(x,x,x,x)$, то получаем

$$1 = f_1(f_1(x,x,x,x), x, x, x),$$

$$0 = f_1(f_1(f_1(x,x,x,x),x,x,x), f_1(f_1(x,x,x,x),x,x,x), f_1(f_1(x,x,x,x),x,x,x), f_1(f_1(x,x,x,x),x,x,x)).$$

Подставив эти формулы в написанную выше формулу для конъюнкции, получим окончательно формулу над базисом $\{f_1\}$ для конъюнкции. Мы не выписываем эту формулу ввиду ее громоздкости.

6.8. Схемы из функциональных элементов

Представлению *булевых функций формулами* можно придать следующий „инженерно-конструктивный“ смысл. Будем рассматривать формулу $\Phi(x_1,\ldots,x_n)$ *над* каким-то произвольно фиксированным *множеством* F как „черный ящик“, некое устройство, на вход которого подаются всевозможные *наборы значений переменных*, а на выходе появляются соответствующие этим наборам значения функции f, представляемой формулой Φ (рис. 6.22).

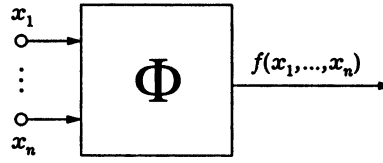


Рис. 6.22

Чтобы понять, как устроен „черный ящик“, мы должны разобрать процесс построения формулы из *подформул*. Добираясь до „базисных“ подформул, т.е. элементов множества F, мы приходим к „кирпичикам“, структурным элементам, из которых собран „черный ящик“, вычисляющий функцию f. Каждая функция „базиса“ F вычисляется соответствующим „узлом“, который рассматривается как мельчайшая структурная единица нашего „черного ящика“, и его внутренняя структура уже не анализируется.

Пример 6.22. Выберем в качестве множества F *стандартный базис*. Тогда формула над стандартным базисом, представляющая функцию $\sim$ (*эквивалентность*), строится следующим образом:

$$x_1 \sim x_2 = \overline{x_1\overline{x}_2 \vee \overline{x}_1 x_2}. \qquad (6.21)$$

Вычисление по этой формуле (и процесс ее построения из элементов стандартного базиса) можно схематически изобразить так, как показано на рис. 6.23.

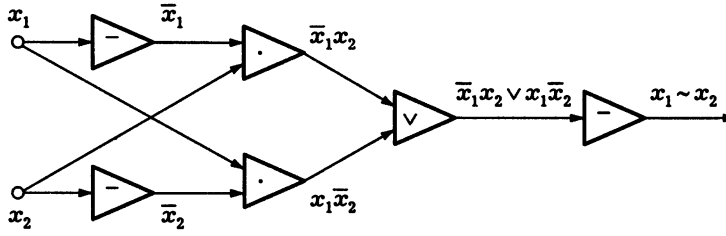


Рис. 6.23

Переменное x_1 (точнее, значение этого переменного) подается на вход структурного элемента, называемого ***инвертором*** (рис. 6.24, *а*) и вычисляющего *отрицание*. Снимаемое с выхода инвертора отрицание x_1, т.е. функция $\overline{x}_1$, подается на один из входов ***конъюнктора*** (рис. 6.24, *б*), на второй вход которого подается переменное x_2. На выходе конъюнктора появляется функция $\overline{x}_1x_2$. Аналогично прослеживается вычисление функции $x_1\overline{x}_2$. Обе эти функции подаются на входы ***дизъюнктора*** (рис. 6.24, *в*), с выхода которого снимается функция $x_1\overline{x}_2 \vee \overline{x}_1x_2$ (это не что иное, как *сумма по модулю 2*: $x_1 \oplus x_2$). И наконец, эта функция подается на вход инвертора, на выходе которого уже получается функция $\sim$ (эквивалентность). #

а *б* *в*

Рис. 6.24

Таким образом, мы приходим к идее „схемы“ — математической модели вычислителя булевой функции, представленной некоторой формулой, собранного из структурных элементов,

каждый из которых вычисляет одну из „базисных" булевых функций. В общем случае „схема" вычисляет *булев оператор*, причем каждая *координатная функция* этого оператора снимается с одного из выходов схемы.

Математически „схема" определяется как *ориентированный граф* специального вида, в котором и *вершины*, и *дуги* снабжены некоторыми *метками*.

Введем обозначение: если F — какое-то множество булевых функций, то через $F^{(n)}$ обозначаем подмножество F, состоящее из всех функций от n переменных ($n \geqslant 0$).

Определение 6.14. Пусть фиксированы *множества*: F (булевых функций) и X (булевых переменных).

Схемой из функциональных элементов над базисом $F \cup X$ (***СФЭ***), или просто ***схемой над базисом*** $F \cup X$, также (F,X)-***схемой***, называют *бесконтурный ориентированный граф* (т.е. *сеть*), каждая *вершина* которого помечена одним из элементов множества $F \cup X$ так, что выполняются следующие требования:

1) каждый *вход сети* помечен либо некоторым переменным из X, либо некоторой константой из $F^{(0)}$;

2) если вершина v сети помечена функцией f от n переменных (т.е. $f \in F^{(n)}$), то ее *полустепень захода* равна n, причем на множестве *дуг, заходящих в вершину* v, задана (взаимно однозначная) *нумерация*, при которой каждая дуга получает номер от 1 до n.

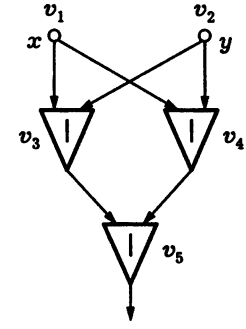


Рис. 6.25

При изображении схем входы обозначаются кружочками, а вершины, не являющиеся входами, — треугольниками, внутри которых записано обозначение функции, помечающей данную вершину. Выходы отмечаются „выходными" стрелками. На рис. 6.25 приведена СФЭ над базисом $\{|, x, y\}$.

Если базис подразумевается, то мы будем говорить просто „схема". Кроме того, если множество переменных фиксировано „раз и навсегда" и при рассмотрении различных схем мы меняем только множество функций F, то, как это мы делали, вводя понятия *формулы* и *суперпозиции над* заданным *базисом*, будем говорить о СФЭ над базисом F, полагая каждый раз, что подразумевается однажды фиксированное множество переменных X, которое (если это не вредит точности) не упоминается.

Определим теперь по индукции понятие ***булевой функции, вычисляемой вершиной схемы.***

Определение 6.15. Пусть задана СФЭ S над базисом $F \cup X$, множество вершин которой есть V.

1. Принимается, что каждый вход СФЭ вычисляет булеву функцию, которой он помечен (т.е. некоторое переменное или константу).

2. Если вершина $v \in V$ помечена функцией $f \in F^{(n)}$, заходящая в нее дуга с номером i $(1 \leqslant i \leqslant n)$ исходит из вершины $u_i \in V$, которая вычисляет функцию g_i, то вершина v вычисляет *суперпозицию* $f(g_1, \ldots, g_n)$.

Таким образом, если каждая вершина СФЭ над F вычисляет некоторую функцию, то порядок, в котором перечисляются функции $g_1, \ldots, g_n$, подставляемые на места переменных функции f, в общем случае существен. Естественно назвать булеву функцию f от n переменных ***коммутативной***, если она сохраняет значение при произвольной перестановке ее переменных. В этом случае мы можем не заботиться о нумерации дуг, заходящих в вершину схемы, помеченную такой функцией.

Пример 6.23. Рассмотрим СФЭ на рис. 6.25. Вершины v_1 и v_2 — входы СФЭ. Эти вершины вычисляют соответственно функции x и y. Тогда вершина v_3, равно как и вершина v_4, согласно определению 6.15, вычисляет функцию $x|y$ (*штрих Шеффера*), а вершина v_5 (*выход сети*) — функцию $(x|y)|(x|y)$, которая, как известно, равна конъюнкции $x \cdot y$.

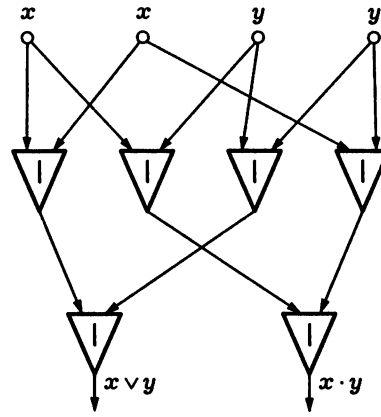


Рис. 6.26

СФЭ, изображенная на рис. 6.26, имеет два выхода, вычисляющие функции $(x|x)|(y|y) = x \vee y$ и $(x|y)|(x|y) = x \cdot y$.

Определение 6.16. ***Булева функция, вычисляемая СФЭ*** над базисом $F \cup X$, — это функция, вычисляемая каким-либо из ее выходов.

Таким образом, СФЭ вычисляет ровно столько булевых функций, сколько имеет выходов. СФЭ на рис. 6.25 вычисляет одну функцию, а СФЭ на рис. 6.26 — две.

В общем случае, если $\{x_1, \ldots, x_n\}$ — множество всех переменных, которые служат метками входов схемы $\mathcal{S}$ над базисом $F \cup X$, имеющей m выходов, СФЭ $\mathcal{S}$ определяет отображение *булева куба* $\mathbb{B}^n$ в булев куб $\mathbb{B}^m$, т.е. *булев оператор*.

Замечание 6.10. В некоторых случаях функцию, вычисляемую данной СФЭ, определяют несколько иначе, полагая, что это функция, вычисляемая любой вершиной из подмножества выделенных вершин СФЭ. В частности, это могут быть и выходы. В любом случае договоримся из выделенных (в только что указанном смысле) вершин схемы проводить „выходную“ стрелку. #

Таким образом, каждая схема из функциональных элементов вычисляет некоторый булев оператор, в частности, если число выходов схемы равно 1, то она вычисляет некоторую булеву функцию.

Можно доказать и обратное: по любому булеву оператору может быть построена СФЭ над базисом F, где F — *полное множество*, вычисляющая данный оператор.

Таблица 6.9

x_1	x_2	x_3	y_1	y_2
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

Пример 6.24. Зададим таблицей булев оператор, отображающий $\mathbb{B}^3$ в $\mathbb{B}^2$ (табл. 6.9).

Из таблицы легко увидеть, что

$$y_1 = x_1x_2 \vee x_1x_3 \vee x_2x_3,$$
$$y_2 = x_1 \oplus x_2 \oplus x_3$$

(функция y_1 есть не что иное, как *мажоритарная функция* от переменных x_1, x_2, x_3, и выше написана *минимальная ДНФ* для нее, см. пример 6.12).

Представим функцию y_1 в *базисе Жегалкина*. Используя *законы де Моргана*, получим

$$x_1x_2 \vee x_1x_3 \vee x_2x_3 = \overline{\overline{x_1x_2} \cdot \overline{x_1x_3} \cdot \overline{x_2x_3}}.$$

Учитывая, что $\overline{x} = x \oplus 1$, будем иметь

$$\overline{\overline{x_1x_2} \cdot \overline{x_1x_3} \cdot \overline{x_2x_3}} = (x_1x_2 \oplus 1)(x_1x_3 \oplus 1)(x_2x_3 \oplus 1) \oplus 1 =$$
$$= x_1x_2x_3 \oplus x_1x_2x_3 \oplus x_1x_2x_3 \oplus x_1x_2 \oplus x_1x_2x_3 \oplus$$
$$\oplus x_1x_3 \oplus x_2x_3 \oplus 1 \oplus 1 = x_1x_2 \oplus x_1x_3 \oplus x_2x_3$$

(напомним, что *сумма по модулю 2* любого четного числа равных слагаемых равна 0).

Итак,

$$y_1 = x_1x_2 \oplus x_1x_3 \oplus x_2x_3 = x_1x_2 \oplus x_3(x_1 \oplus x_2).$$

СФЭ для булева оператора, заданного в табл. 6.9, над базисом Жегалкина приведена на рис. 6.27.

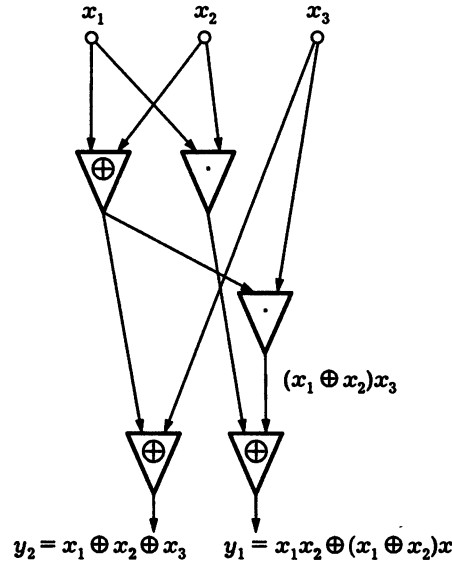


Рис. 6.27

При проектировании СФЭ полезно иметь в виду числовой параметр, называемый ее сложностью.

Сложность СФЭ — это число ее вершин, не являющихся входами.

Приведенная на рис. 6.27 СФЭ над базисом Жегалкина имеет сложность 5.

Рассмотрим теперь СФЭ для того же оператора над стандартным базисом.

По таблице (см. табл. 6.9) строим *СДНФ* для функции y_2:

$$y_2 = \overline{x}_1\overline{x}_2x_3 \vee \overline{x}_1x_2\overline{x}_3 \vee x_1\overline{x}_2\overline{x}_3 \vee x_1x_2x_3.$$

Карта Карно для этой функции, изображенная на рис. 6.28, показывает, что ее нельзя минимизировать (точнее, записанная выше СДНФ и есть минимальная ДНФ для этой функции).

Но можно пойти по другому пути. Мы можем рассматривать табл. 6.9 как таблицу, определяющую *частичную булеву функцию* $y_2 = y_2(x_1, x_2, x_3, y_1)$. Минимизируя эту функцию по

x_1 \ x_2x_3	00	01	11	10
0		1		1
1	1		1	

Рис. 6.28

x_1x_2 \ x_3y_1	00	01	11	10	
00	0	–	–	1	××01
01	1	–	0	–	×1×0
11	–	0	1	–	1××0
10	1	–	0	–	111×

Рис. 6.29

карте Карно*, изображенной на рис. 6.29, получаем

$$y_2 = x_1x_2x_3 \vee \overline{y}_1(x_1 \vee x_2 \vee x_3).$$

СФЭ над стандартным базисом для рассматриваемого булева оператора приведена на рис. 6.30. Сложность этой СФЭ составляет 11. Заметим, что вершина, вычисляющая функцию y_1, не является выходом.

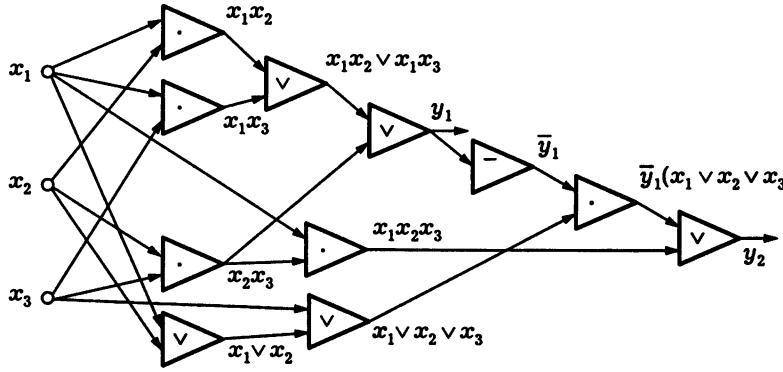


Рис. 6.30

*На этой карте мы явно обозначили наборы, на которых функция принимает значение 0, проставив нули в соответствующих клетках. Тем самым мы хотим еще раз зафиксировать внимание на том, что не следует путать нули с прочерками: прочерк в клетке карты, задающей частичную функцию, означает, что на данном наборе значение функции не определено, т.е. не равно ни 0, ни 1.

Булев оператор, рассмотренный в этом примере, вычисляет двухразрядную сумму (по модулю 2) трех одноразрядных слагаемых. Его можно считать также одноразрядным двоичным сумматором — функциональным блоком многоразрядного двоичного сумматора — для двух слагаемых. Тогда функция y_1 интерпретируется как „сигнал переноса“ в старший разряд. На рис. 6.31 изображено „соединение“ трех СФЭ (таких, как показано на рис. 6.30), с помощью которого вычисляется сумма двух трехразрядных двоичных чисел. На третий вход сумматора для младшего разряда подается константа 0, а „сигнал переноса“ старшего разряда есть старший разряд суммы, которая в общем случае будет четырехразрядным числом.

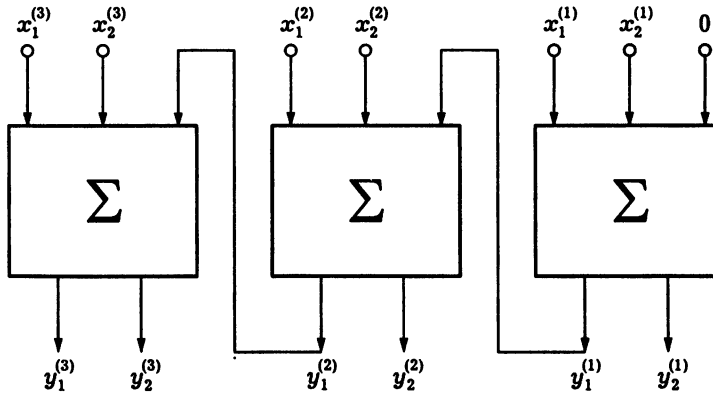


Рис. 6.31

Замечание 6.11. Если мы проектируем СФЭ над стандартным базисом и хотим минимизировать ее сложность, то нам необходимо прежде всего построить соответствующую минимальную ДНФ. В этом случае мы можем принять во внимание еще один критерий, по которому минимизируется сама ДНФ, — число отрицаний. Среди всех минимальных (в смысле определения 6.6) ДНФ следует отобрать те, в которых число вхождений переменных под знаком отрицания является наименьшим. С точки зрения сложности СФЭ, которая будет

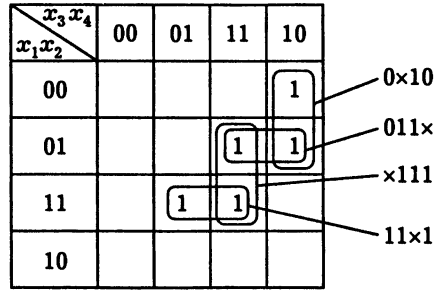


Рис. 6.32

построена по минимальной ДНФ, это означает, что в ней минимизируется число „инверторов“ — вершин СФЭ, помеченных функцией отрицания.

Например, для функции, заданной картой Карно (рис. 6.32), к *ядру*, состоящему из *простых импликант* $x_1x_2x_4$ и $\overline{x}_1x_3\overline{x}_4$, следует добавить простую импликанту $x_2x_3x_4$, а не $\overline{x}_1x_2x_3$, поскольку она не содержит отрицаний.

Вопросы и задачи

6.1. Найти число дуг в булевой n-сети.

6.2. Доказать, что объединение двух соседних граней размерности $n-k$ булева куба $\mathbb{B}^n$ является гранью размерности $n-k+1$. Как найти направление этой грани?

6.3. Найти число вершин k-слоя булева куба $\mathbb{B}^n$.

6.4. Построить таблицы для булевых функций, заданных формулами:

а) $(x \to y) \vee (x \to (x \cdot y))$;

б) $(\overline{x} \to \overline{(x \cdot y)}) \to (x \vee z)$;

в) $(x \cdot (y \to x)) \to \overline{x}$;

г) $(x \to (y \to z)) \to ((x \to y) \to (x \to z))$;

д) $(x \cdot (y \vee \overline{x})) \cdot ((\overline{y} \to x) \vee y)$.

6.5. Доказать, что если высказывания U и $(U \Rightarrow B)$ тождественно истинны, то высказывание B тождественно истинно.

6.6. Доказать, что если формулы $(U \vee B)$ и $(\overline{U} \vee W)$ тождественно истинны, то формула $(B \vee W)$ тождественно истинна.

6.7. Найти фиктивные переменные функции f, заданной вектором значений:

а) $f = (0,0,1,1,0,0,1,1)$; б) $f = (0,0,1,1,1,1,0,0)$.

6.8. Показать, что x является фиктивным переменным функции f, представленной формулами:

а) $f = ((z \to y) \vee x)(y \to x)z\overline{x}$;

б) $f = ((x \vee y)(x \vee \overline{z}) \to (\overline{x} \to y\overline{z}))y$.

6.9. Доказать, что для любой булевой функции f от n переменных имеет место следующее равенство:

$$f(x_1,\ldots,x_i,\ldots,x_n) = x_i f(x_1,\ldots,1,\ldots,x_n) \vee \overline{x}_i f(x_1,\ldots,0,\ldots,x_n).$$

(Это равенство называют разложением функции f по переменному x_i.)

6.10. Пусть булевы функции f и g имеют одно и то же число существенных переменных, равное r. Тогда каждый набор $\widetilde{\alpha} \in \in \{0,1\}^r$ однозначно определяет набор значений существенных переменных каждой из функций. Пусть для любого такого $\widetilde{\alpha}$ значения функций f и g на соответствующих наборах своих существенных переменных равны. Следует ли отсюда, что эти функции равны?

6.11. Используя алгоритм Квайна — Мак-Клоски, найти минимальные ДНФ для функций:

а) $f = (0,1,1,0,1,0,0,1)$;

б) $f = (1,0,0,1,0,0,1,1)$;

в) $f = \{1,3,4,7,8,11,14,15\}$;

г) $f = \{2,4,6,9,10,11,13\}$.

6.12. Каждый из четырех членов комитета голосует „за“, нажимая на кнопку. Решение считается принятым, когда не менее трех членов комитета голосуют „за“. Найти минимальную ДНФ для функции голосования.

6.13. Определить число булевых функций от n переменных: а) сохраняющих константу (0 или 1); б) самодвойственных; в) несамодвойственных; г) линейных.

6.14. Доказать, что замыкание множества функций, состоящего из дизъюнкции и суммы по модулю 2, совпадает с классом T_0.

6.15. Доказать, что множество $T_0 \cap T_1$ является замыканием одноэлементного множества $\{f = xy \oplus z \oplus t\}$.

6.16. Доказать, что число всех монотонных функций от n переменных равно числу всех антицепей булева куба размерности n. Одноэлементное множество считать антицепью.

6.17. Доказать, что сокращенная ДНФ, представляющая монотонную функцию, является минимальной.

6.18. Доказать, что любая монотонная функция, отличная от константы, может быть представлена ДНФ без отрицаний.

6.19. Доказать, что в булевом кубе размерности n существует антицепь, мощность которой составляет $C_n^{[n/2]}$. Используя этот результат, доказать, что число монотонных функций от n переменных не меньше $2^{C_n^{[n/2]}}$.

6.20. Доказать, что число монотонных функций от n переменных не меньше $\left(\sum\limits_{k=0}^{n} 2^{C_n^k}\right) - n$.

Указание: используйте результаты задач 6.3 и 6.16.

6.21. Методом неопределенных коэффициентов найти полином Жегалкина для функций:

а) $f = (0, 1, 1, 0, 1, 0, 0, 1)$;

б) $f = (x_1|x_2) \downarrow x_3$;

в) $f = (1, 0, 1, 0, 0, 0, 1, 1)$;
г) $f = (1001110000011000)$.

6.22. Выяснить, являются ли самодвойственными следующие функции:
а) $\overline{xy}(\overline{z}(y \to x)) \to (\overline{x} \sim y)$;
б) $(((\overline{x} \vee y) \sim z) \sim (y \sim z)) \to (\overline{x} \vee z)$;
в) $f = (01101001)$;
г) $f = (10101011)$;
д) $f = (1100100101101100)$.
Для несамодвойственных функций найти двойственные функции.

6.23. Выяснить, полно ли множество булевых функций F:
а) $F = \{f_1 = \overline{x}, f_2 = x(y \sim z) \sim yz, f_3 = x \oplus y \oplus z\}$;
б) $F = \{f_1 = xy \oplus yz \oplus zt, f_2 = 0, f_3 = 1, f_4 = x \vee y\}$;
в) $F = \{f_1 = 0110, f_2 = (11000011), f_3 = (10010110)\}$;
г) $F = \{f_1 = x \vee y, f_2 = (1001101111110110)\}$.
Для полного множества F построить формулы над F, представляющие элементы стандартного базиса и базиса Жегалкина. Реализовать эти формулы схемами из функциональных элементов.

6.24. Полное множество булевых функций называют базисом, если оно не содержит полных собственных подмножеств (т.е. является минимальным по включению полным множеством). Найти любой базис, содержащий импликацию ($\to$).

6.25. Проверить полноту множества F, состоящего из функций $f_1 = xy \vee \overline{x}z$, $f_2 = x \to y$, 0 и $x \oplus zy$. Выделить в нем всевозможные базисы.

7. КОНЕЧНЫЕ АВТОМАТЫ И РЕГУЛЯРНЫЕ ЯЗЫКИ

В этой главе мы начинаем изложение элементов теории формальных языков.

Говоря „формальный язык“, мы имеем в виду то, что приведенные здесь результаты используются прежде всего при описании искусственных языков, придуманных людьми для специальных целей, например языков программирования. Но непреодолимой преграды между специально придуманными искусственными (формальными) языками и стихийно возникающими и развивающимися естественными языками не существует. Оказывается, что естественные языки характеризуются сложными грамматическими правилами, т.е. довольно жестко формализованы, а даже самый „научно разработанный“ язык программирования содержит „темные места“, однозначное понимание которых является проблемой.

Изучая языки, следует иметь в виду три основных аспекта.

Первый из них — ***синтаксис языка***. Язык — это какое-то множество „слов“, где „слово“ есть определенная конечная последовательность „букв“ — символов какого-то заранее фиксированного алфавита. Термины „буква“ и „слово“ могут пониматься по-разному (математическое определение этих терминов будет дано ниже). Так, „буквами“ могут быть действительно буквы алфавита какого-нибудь естественного или формального языка, например русского языка или языка программирования „Паскаль“. Тогда „словами“ будут конечные последовательности „букв“: „крокодил“, „**integer**“. Такие слова называют „лексемами“. Но „буквой“ может быть „слово“ („лексема“) в целом. Тогда „слова“ — это предложения естественного языка или программы языка программирования. Ес-

ли фиксировано какое-то множество „букв“, то не каждая их последовательность будет „словом“, т.е. „лексемой“ данного языка, а только такая последовательность, которая подчиняется определенным правилам. Слово „крыкадил“ не является лексемой русского языка, а слово „iff“ не является лексемой в „Паскале“. Предложение „Я люблю ты“ не является правильным предложением русского языка, точно так же, как и запись „x:==t“ не есть правильно написанный оператор присваивания „Паскаля“. Синтаксис* языка и представляет собой систему правил, в соответствии с которыми можно строить „правильные“ последовательности „букв“. Каждое слово языка характеризуется определенной структурой, специфичной именно для данного языка. Тогда необходимо, с одной стороны, разработать механизмы перечисления, или порождения, слов с заданной структурой, а с другой — механизмы проверки того, что данное слово принадлежит данному языку. Прежде всего именно эти механизмы и изучает классическая теория формальных языков.

Второй аспект — ***семантика языка***. Семантика** предполагает сопоставление словам языка некоего „смысла“, „значения“. Например, записывая математическую формулу, мы должны соблюдать определенные синтаксические правила (расстановка скобок, правописание символов, порядок символов и т.п.), но, кроме этого, формула имеет вполне определенный смысл, что-то обозначает.

Язык — это средство общения, передачи информации. Если мы хотим, чтобы нас поняли, мы должны не только синтаксически правильно, соблюдая должный порядок букв в слове и слов в предложении, строить свою речь, но и заботиться об ее смысле, о тех идеях, которые мы выражаем в речи. Математические

*Слово „синтаксис“ происходит от древнегреческих „syn“ — „вместе“ и „taxis“ — „порядок, строй“. Таким образом, синтаксис можно понимать как „составление“.

**От древнегреческих слов „sema“ — „знак, знамение“ и „semanticos“ — „обозначающий“.

теории „смысла“ появились сравнительно недавно, и в дополнении **Д.8.2** к следующей главе мы очень коротко рассмотрим некоторые подходы к математическому описанию семантики языков программирования.

Наконец, третий аспект — ***прагматика языка***. Прагматика* связана с теми целями, которые ставит перед собой носитель языка: например, человек произносит речь, имея перед собой цели, связанные не с синтаксисом, не с семантикой языка, на котором он говорит или пишет, а, скажем, с получением за речь определенной суммы денег. Прагматика является уже скорее дисциплиной социально-философской, затрагивающей целеполагающую деятельность личности. Мы ни в малейшей степени не будем ее касаться.

В этой главе вначале будут рассмотрены основные понятия математической теории формальных языков, важнейшим среди которых является понятие порождающей грамматики, а затем — так называемые регулярные языки. Теория регулярных языков вместе с теорией конечных автоматов образует фундамент всей теории формальных языков.

7.1. Алфавит, слово, язык

Рассмотрим самое простое понятие теории языков — понятие алфавита.

Алфавит — это произвольное *непустое конечное множество* $V = \{a_1, \ldots, a_n\}$, *элементы* которого называют ***буквами*** или ***символами***.

Обычно задают определенную *нумерацию* алфавита (как, скажем, для русского алфавита: „а“ — первая буква, „б“ — вторая и т.д. до 33-й — „я“). Впредь договоримся, фиксируя алфавит, записывать его буквы в порядке их номеров.

*От древнегреческих „pragma“ — „дело, действие“ и „pragmateia“ — „деятельность“.

Определение 7.1. ***Словом*** или ***цепочкой*** в алфавите V называют произвольный кортеж из множества V^k (k-й *декартовой степени* алфавита V) для различных $k = 0, 1, 2, \ldots$

Например, если $V = \{a, b, c\}$, то (a), (b), (c), (a, b), (a, b, c), (c, b, a, a, c) и т.д. есть слова в V.

При $k = 0$ получаем *пустой кортеж*, называемый в данном контексте ***пустым словом*** или ***пустой цепочкой*** и обозначаемый λ. Множество всех слов в алфавите V обозначают V^*, а множество всех непустых слов в V — как V^+. Слова, ради удобства чтения и простоты записи, будем записывать без скобок и запятых (ср. с записями кортежей в **1.2**). Так, для записанных выше слов получим: a, b, c, ab, abc, $cbaac$.

Такая запись слова согласуется с его интуитивным пониманием как цепочки следующих друг за другом символов. Тогда пустое слово — это слово, не имеющее символов, „пустой лист бумаги“, на котором еще ничего не написано.

По определению, ***длина слова*** w — число компонент кортежа, т.е. если $w \in V^r$, то длина слова w равна r. Длину слова w договоримся обозначать $|w|$. Ясно, что для пустого слова $|\lambda| = 0$. Длину слова тем самым можно понимать как число составляющих это слово букв.

Докажем, что множество V^* *счетно*. Для этого достаточно построить какую-либо *нумерацию* этого множества. Рассмотрим здесь нумерацию, называемую ***лексикографической***.

В данной нумерации пустому слову присваивается номер 0, а буквам $a_1, \ldots, a_n$ алфавита V — номера $1, \ldots, n$ соответственно. Если слово x имеет лексикографический номер l_x, то слову xa_i присваивается номер $nl_x + i$. Отсюда следует, что лексикографический номер слова $a_{i_1} a_{i_2} \ldots a_{i_k}$ будет равен

$$n^{k-1} i_1 + n^{k-2} i_2 + \ldots + i_k.$$

Заметим, что последняя сумма напоминает запись числа в системе счисления по модулю n (*мощности* алфавита) с тем лишь различием, что используется цифра n, но не допускается

цифра 0. Итак, по любому слову в алфавите V однозначно вычисляется его лексикографический номер. Обратно, любое натуральное число однозначно раскладывается по степеням n указанным выше образом.

Действительно, если дано число N, то при $0 \leqslant N \leqslant n$ оно служит номером пустого слова ($N = 0$) или некоторой буквы алфавита. Иначе представим N в виде

$$N = k_1 n + r_0,$$

где $1 \leqslant r_0 \leqslant n$.

Если $k_1 \leqslant n$, то N есть номер слова $a_{k_1} a_{r_0}$. Иначе раскладываем k_1 в виде

$$k_1 = k_2 n + r_1,$$

где $1 \leqslant r_1 \leqslant n$. Тогда

$$N = k_2 n^2 + r_1 n + r_0.$$

С числом k_2 поступаем точно так же, как и с k_1. После конечного числа шагов получим разложение числа N в виде

$$N = n^m r_m + n^{m-1} r_{m-1} + \ldots + n r_1 + r_0,$$

где каждое число r_i $(0 \leqslant i \leqslant m)$ находится в диапазоне от 1 до n. По полученному разложению N однозначно восстанавливается слово в V, имеющее номер N:

$$a_{r_m} a_{r_{m-1}} \ldots a_{r_1} a_{r_0}.$$

Пример 7.1. Вычислим номер слова *cbaac* в алфавите $\{a, b, c\}$. Имеем

$$3^4 \cdot 3 + 3^3 \cdot 2 + 3^2 \cdot 1 + 3^1 \cdot 1 + 3 = 279.$$

Решим обратную задачу, найдя слово в данном трехбуквенном алфавите, имеющее номер 321.

Согласно приведенному выше алгоритму, получим

$$\begin{aligned} 321 = 106\cdot 3+3 = (35\cdot 3+1)3+3 = \\ = (11\cdot 3+2)3^2+1\cdot 3+3\cdot 3^0 = \\ = (3\cdot 3+2)3^3+2\cdot 3^2+1\cdot 3+3\cdot 3^0 = \\ = 3\cdot 3^4+2\cdot 3^3+2\cdot 3^2+1\cdot 3+3. \end{aligned}$$

Следовательно, искомое слово есть *cbbac*. #

Лексикографическая нумерация напоминает способ упорядочения слов в словарях: однобуквенные слова следуют в порядке номеров букв в алфавите, среди двух двухбуквенных слов меньший номер имеет слово, начинающееся буквой с меньшим номером, и т.д. Но полного совпадения нет, так как в словаре слова группируются по начальной букве, а не по длине.

Нам будет удобно в дальнейшем использовать следующую запись непустого слова x в алфавите V по буквам:

$$x = x(1)x(2)\dots x(k),$$

где $x(i)$, $1 \leqslant i \leqslant k$, — i-я буква слова x.

Определение 7.2. ***Языком в алфавите*** V называется произвольное подмножество множества V^*.

Множество всех языков в алфавите V, т.е. множество 2^{V^*}, есть *булеан* счетного множества, и, следовательно, оно в силу теоремы 1.15 Кантора имеет *мощность континуума*.

Наша следующая задача — определить на множестве 2^{V^*} всех языков в произвольном (но фиксированном!) алфавите V *алгебраическую структуру*. На множестве 2^{V^*} можно определять различные *операции*. Прежде всего языки — это множества, следовательно, над ними можно производить все те же операции, что и над множествами: *объединение, пересечение, разность, дополнение* и т.п. *Универсальное множество* в данном случае есть множество слов V^*, которое называют ***универсальным языком***.

Кроме перечисленных теоретико-множественных операций можно рассматривать и специальные операции над языками.

Прежде чем обратиться к этим операциям, определим операцию ***соединения*** (или ***конкатенации***) ***слов***. Соединением слов $x = x(1)x(2)\ldots x(k)$ и $y = y(1)y(2)\ldots y(m)$ называют слово

$$xy = x(1)x(2)\ldots x(k)y(1)y(2)\ldots y(m).$$

По определению, считаем $x\lambda = \lambda x = x$ для любого x. Соединение иногда обозначают точкой $(\cdot)$.

Неформально соединение xy получается приписыванием слова y справа к слову x. Таким образом, для любых двух слов $x \in V^k$ и $y \in V^m$ конкатенация $xy \in V^{k+m}$ $(k, m \geqslant 0)$. Следовательно, $|xy| = |x| + |y|$.

Из определения также следует, что соединение слов *ассоциативно*, т.е. для произвольных трех слов x, y, z имеет место $x(yz) = (xy)z$, и поэтому — с учетом написанного выше свойства пустого слова — множество V^* всех слов в алфавите V с операцией соединения образует *моноид* $(V^*, \cdot, \lambda)$. *Единица моноида* — пустое слово. Этот моноид есть не что иное, как *свободный моноид*, порожденный алфавитом V (см. пример 2.7). Для него используют то же обозначение, что и для самого множества всех слов в алфавите V, т.е. V^*.

На основе понятия соединения слов определим понятие вхождения одного слова в другое.

Определение 7.3. ***Вхождение слова*** $x \in V^*$ ***в слово*** $y \in V^*$ — это *упорядоченная тройка* слов (u, x, v), такая, что $y = uxv$.

При этом слово u называют ***левым***, а слово v — ***правым крылом*** указанного ***вхождения***. Слово x называют ***основой вхождения***.

Говорят, что слово x входит в слово y, если существует вхождение x в y. При этом также слово (цепочку) x называют ***подсловом*** (или ***подцепочкой***) слова (цепочки) y. Подцепочку x цепочки y называют ***началом*** (или ***префиксом***) ***цепочки*** y, если $y = xz$ для некоторой непустой цепочки z; если же для

некоторой непустой цепочки z имеет место $y = zx$, то цепочку x называют ***концом*** (или ***постфиксом***) ***цепочки*** y.

Заметим, что каждое слово входит в себя само и пустое слово входит в любое слово.

Например, слова „цикл“ и „циклоп“ входят в слово „энциклопедия“. Соответствующие вхождения записывают следующим образом:

(эн, цикл, опедия), (эн, циклоп, едия).

Может существовать несколько разных вхождений одного и того же слова x в некоторое слово y. Так, слово „абра“ дважды входит в слово „абракадабра“. Число вхождений пустого слова в данное слово p на единицу больше длины слова p. Среди всех вхождений слова x в слово y вхождение с наименьшей длиной левого крыла называют ***первым*** или ***главным вхождением*** x в y.

Так, вхождение (λ, абра, кадабра) есть первое вхождение слова „абра“ в слово „абракадабра“.

Определение 7.4. Говорят, что ***вхождения*** (u, x, v) и (s, z, t) слов x и z в одно и то же слово y ***не пересекаются***, если существуют такие (может быть, и пустые) слова p и q, что $y = uxpzt$ (и тогда $v = pzt$, а $s = uxp$) или $y = szqxv$ (и тогда $u = szq$, а $t = qxv$) (рис. 7.1). В противном случае говорят, что указанные ***вхождения пересекаются***.

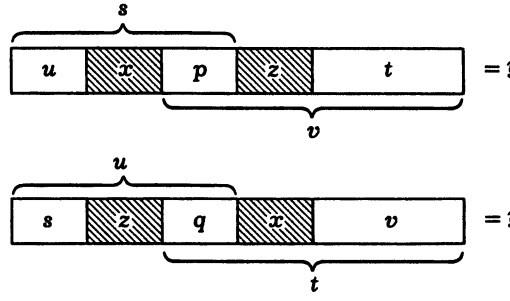


Рис. 7.1

Так, вхождения слов „цикл“ и „циклоп“ в слово „энциклопедия“ пересекаются, а два разных вхождения слова „абра“ в слово „абракадабра“ не пересекаются. Мы иногда будем использовать обозначение $x \sqsubseteq y$ для утверждения „слово x входит в слово y“. Можно доказать, что $\sqsubseteq$ — *отношение порядка.*

Определив таким образом операцию соединения слов, введем теперь операцию с таким же названием, но уже для языков. Перед этим обратим внимание на то, что всякий раз, говоря о языках и операциях над ними, мы полагаем фиксированным какой-то алфавит V. Он не всегда явно упоминается, но нужно четко усвоить, что нельзя говорить просто о слове, просто о языке, но всегда — о слове или языке в том или ином алфавите.

Определение 7.5. ***Соединением (конкатенацией) языков*** L_1 и L_2 называют язык L_1L_2, состоящий из всех возможных соединений слов xy, в которых слово x принадлежит первому, а слово y — второму языку, т.е.

$$L_1L_2 = \{xy\colon x \in L_1 \text{ и } y \in L_2\}.$$

Соединение конечных языков легко вычислить.

Пример 7.2. Если $V = \{a, b, c\}$, $L_1 = \{ab, bcc, cab\}$, $L_2 = \{ca, bcc\}$, то

$$L_1L_2 = \{abca, abbcc, bccca, bccbcc, cabca, cabbcc\},$$

а

$$L_2L_1 = \{caab, cabcc, cacab, bccab, bccbcc, bcccab\}.$$

Вычисление конкатенации языков в конечном случае очень похоже на умножение (раскрытие скобок) в обычной школьной алгебре. Можно было бы формально написать так:

$$(ab + bcc + cab)(ca + bcc) = \\ = abca + abbcc + bccca + bccbcc + cabca + cabbcc.$$

В данном случае плюс (+) — это только соединительный знак, используемый вместо запятой. Позже мы увидим, что подобным чисто формальным записям может быть придан строгий алгебраический смысл. #

Соединение языков не коммутативно, и, как показывает только что разобранный пример, пересечение $L_1L_2 \cap L_2L_1$ в общем случае не пусто. В нашем примере оно содержит одну цепочку *bccbcc*.

Операция соединения языков позволяет определить операцию возведения языка в произвольную натуральную степень. А именно, по определению, $L^0 = \{\lambda\}$ для любого $L \subseteq V^*$, а $L^n = L^{n-1}L$ при $n > 0$.

Итерацией языка L называют объединение всех его степеней:

$$L^* = \bigcup_{n=0}^{\infty} L^n.$$

Рассматривая объединение всех степеней языка L, начиная с первой, получим ***позитивную итерацию***

$$L^+ = \bigcup_{n=1}^{\infty} L^n.$$

Сформулируем основное алгебраическое свойство множества всех языков в алфавите V.

Теорема 7.1. Алгебра $\mathcal{L}(V) = \left(2^{V^*}, \cup, \cdot, \varnothing, \{\lambda\}\right)$ есть *замкнутое полукольцо.*

◀ Проверка *аксиом полукольца* (см. **3**) сводится к доказательству:

1) того, что по операции объединения множество всех языков образует *коммутативный* и *идемпотентный моноид* (с пустым множеством в качестве нейтрального элемента (*нуль полукольца*)); это тривиально ввиду известных свойств операции объединения множеств;

2) того, что по операции конкатенации множество языков образует моноид (с языком $\{\lambda\}$, состоящим из одного пустого слова, в качестве нейтрального элемента (*единицы полукольца*)); для этого достаточно доказать, что операция соединения языков ассоциативна, а также доказать для любого языка L тождество

$$\{\lambda\}L = L\{\lambda\} = L,$$

что вытекает из ассоциативности операции соединения слов и из тождества $\lambda x = x\lambda = x$ для любого слова x;

3) следующих тождеств:

$$L_1(L_2 \cup L_3) = L_1L_2 \cup L_1L_3,$$
$$(L_1 \cup L_2)L_3 = L_1L_3 \cup L_2L_3$$

(эти тождества определяют свойство *дистрибутивности* операции соединения относительно объединения).

Докажем первое из этих тождеств. Пусть слово x принадлежит его левой части, т.е. языку $L_1(L_2 \cup L_3)$. Тогда, согласно определению соединения языков, это слово может быть представлено в виде $x = yz$, где $y \in L_1$, а $z \in L_2 \cup L_3$, т.е. $z \in L_2$ или $z \in L_3$. Если $z \in L_2$, то $yz \in L_1L_2$, а если $z \in L_3$, то $yz \in L_1L_3$, т.е. $x = yz \in L_1L_2 \cup L_1L_3$. Пусть теперь $x \in L_1L_2 \cup L_1L_3$. Тогда $x = yz$, где $y \in L_1$, а $z \in L_2$ или $z \in L_3$, т.е. $x \in L_1(L_2 \cup L_3)$, что и завершает доказательство первого тождества. Второе доказывается аналогично.

Напомним, что в полукольце $\mathcal{S} = (S, +, \cdot, 0, 1)$ *отношение порядка* вводится следующим образом: для любых $x, y \in S$ по определению полагают $x \leqslant y$ тогда и только тогда, когда $x + y = y$. Так как в полукольце всех языков в алфавите V операция *сложения* — это операция объединения множеств, то в данном случае отношение порядка $\leqslant$ есть не что иное, как *теоретико-множественное включение* $\subseteq$ (действительно, включение $L \subseteq K$ равносильно тому, что $L \cup K = K$). Тогда замкнутость полукольца $\mathcal{L}(V)$ следует из существования *объединения* любого

семейства множеств (в частности, бесконечной последовательности множеств — см. **1.5**), служащего *точной верхней гранью* этого семейства (относительно теоретико-множественного включения), а также из следующих тождеств (для любого языка L и любого семейства языков P_i, $i \in I$):

$$L\Big(\bigcup_{i\in I} P_i\Big) = \bigcup_{i\in I}(LP_i), \quad \Big(\bigcup_{i\in I} P_i\Big)L = \bigcup_{i\in I} P_iL, \tag{7.1}$$

что гарантирует выполнение *непрерывности* операции *умножения* данного *полукольца*, т.е. непрерывности операции соединения. Эти тождества доказываются точно так же, как тождества обычной дистрибутивности.

Рассмотрим, например, доказательство второго тождества из (7.1), используя, как и выше, *метод двух включений*. Если $x \in \Big(\bigcup\limits_{i\in I} P_i\Big)L$, то $x = yz$, где $y \in \bigcup\limits_{i\in I} P_i$, а $z \in L$. Согласно определению объединения семейства множеств, найдется такое $i \in I$, что $y \in P_i$, и тогда $yz = x \in P_iL$, т.е. $x \in \bigcup\limits_{i\in I} P_iL$. Обратное включение доказываем так: из $x \in \bigcup\limits_{i\in I} P_iL$ следует, что для некоторого $i \in I$ $x \in P_iL$, т.е. $x = yz$, где $y \in P_i$, а $z \in L$, откуда $y \in \bigcup\limits_{i\in I} P_i$, и, следовательно, $yz = x \in \Big(\bigcup\limits_{i\in I} P_i\Big)L$. ▶

Следствие 7.1. Для любого языка L верно тождество $L^+ = L^*L = LL^*$.

◀ Вычислим соединение LL^*: $LL^* = L\Big(\bigcup\limits_{n=0}^{\infty} L^n\Big)$. Применяя первое из тождеств (7.1), получим $L\bigcup\limits_{n=0}^{\infty} L^n = \bigcup\limits_{n=0}^{\infty} LL^n = \bigcup\limits_{n=1}^{\infty} L^n$, т.е. $L^+ = L^*L$. Тождество $L^+ = LL^*$ доказывается аналогично. ▶

Заметим, что в общем случае нельзя утверждать, что позитивная итерация языка L получается выбрасыванием из обычной итерации пустого слова. Это верно в том и только в том случае, когда язык L не содержит пустого слова. Если же $\lambda \in L$, то $L^+ = L^*$, так как тогда $L^0 = \{\lambda\} \subseteq L$.

7.2. Порождающие грамматики

Как уже отмечалось, классическая теория формальных языков изучает прежде всего *синтаксис языка*. Она вводит математическую модель синтаксиса, которая описывает механизмы порождения и распознавания „правильно построенных" цепочек. В этом разделе мы рассмотрим первый из этих механизмов. Задать такой механизм — значит описать некую процедуру, позволяющую вывести (или, как обычно говорят, породить) произвольную *цепочку* языка согласно определенному конечному множеству правил. Последнее очень важно: язык может быть бесконечным, но множество правил, с помощью которых выводятся его цепочки, обязано быть конечным. Заметим, что не каждый язык может быть представлен таким образом, но мы в рамках этого учебника будем рассматривать только такие языки, т.е. языки, синтаксис которых задается конечным множеством правил. Эти языки, вообще говоря бесконечные, но допускающие конечное описание, называют ***перечислимыми***. Таким образом, мы рассматриваем только перечислимые языки.

Классическим способом определения языков через порождающую процедуру является определение их с помощью ***порождающих грамматик***, или ***грамматик Хомского***.

Порождающая грамматика позволяет выводить (порождать) цепочки языка из некоторой начальной цепочки с помощью определенных правил замены (или правил подстановки, правил переписывания). Порождение есть пошаговый процесс, в котором на каждом шаге из цепочки, уже полученной на предыдущем шаге (в частности, из начальной), можно путем применения к ней правил замены получить новую цепочку. Именно так мы действуем, решая какую-нибудь математическую задачу, например выполняя дифференцирование и переходя от одной формулы к другой с помощью таблицы производных.

Порождающая грамматика (далее просто грамматика) задается упорядоченной четверкой $G = (V, N, S, P)$, в которой

V — алфавит, называемый ***терминальным алфавитом***; N — алфавит, называемый ***нетерминальным алфавитом***, причем пересечение V и N пусто; S — выделенный символ нетерминального алфавита, называемый ***аксиомой***; P — конечное множество ***правил вывода***, или ***продукций***. Каждое правило вывода является *упорядоченной парой* (α,β) цепочек в алфавите $V \cup N$, причем цепочка α должна содержать *вхождение* хотя бы одного символа нетерминального алфавита; цепочку α называют ***левой***, цепочку β — ***правой частью правила вывода***.

Правило вывода принято записывать в таком виде:

$$\alpha \to \beta,$$

разделяя левую и правую части „стрелкой“, которая рассматривается как „метасимвол“ и не принадлежит ни одному из алфавитов грамматики. *Буквы* терминального алфавита грамматики называют ***терминальными символами*** (или просто ***терминалами***); буквы нетерминального алфавита называют ***нетерминальными символами*** (или ***нетерминалами***). Любую цепочку в терминальном (нетерминальном) алфавите называют ***терминальной*** (***нетерминальной***) ***цепочкой***. Алфавит $V \cup N$ называют ***объединенным алфавитом*** грамматики G.

Пример 7.3. Четверка

$$G_0 = \big(\{a,b\},\ \{S,A,B\},\ S,\ \{S \to aABb,\ aA \to aB,\ aB \to aBa,\ aA \to aa,\ aBb \to aabb,\ aBa \to abBba,\ Bb \to Ab\}\big)$$

задает грамматику с терминальным алфавитом $V = \{a,b\}$, с нетерминальным алфавитом $N = \{S,A,B\}$, с аксиомой S и множеством правил вывода P, элементы которого перечислены в фигурных скобках после аксиомы. Обычно ради наглядности правила вывода грамматики выписывают отдельно:

$$S \to aABb,\quad aA \to aB,\quad aB \to aBa,\quad aA \to aa,$$
$$aBb \to aabb,\quad aBa \to abBba,\quad Bb \to Ab.$$

Замечание 7.1. Нам будет удобно условиться о некоторых обозначениях, которыми мы все время будем пользоваться, имея дело с грамматиками. Терминалы будем обозначать малыми латинскими буквами из начала латинского алфавита: a, b, c и т.д.; нетерминалы — большими латинскими буквами из начала и середины латинского алфавита: A, B, C, S, T и т.д.; терминальные цепочки (т.е. слова в терминальном алфавите) — малыми латинскими буквами из середины и конца латинского алфавита: s, t, p, q, x, y, z и т.д.; цепочки в объединенном алфавите — малыми греческими буквами. Наконец, большими латинскими буквами из конца алфавита будем обозначать символы объединенного алфавита или пустую цепочку. #

Определение грамматики указывает пока только на то, что следует фиксировать, чтобы задать какую-либо грамматику. А именно следует фиксировать два непересекающихся алфавита, выделив в одном из них символ, названный аксиомой, а также конечное множество правил вывода. Но мы еще не знаем, каким образом „работает“ грамматика как инструмент порождения (и преобразования) слов. Чтобы это понять, нужно определить важнейшее понятие выводимости в данной грамматике.

Неформально каждое правило вывода грамматики трактуется как „правило замены“, разрешающее произвольное вхождение левой части правила в некоторую цепочку в объединенном алфавите заменить его правой частью, получив (или выведя) тем самым новую цепочку. Так, для примера 7.3 мы можем от цепочки S, используя правило $S \to aABb$, перейти к цепочке $aABb$. В этой цепочке есть вхождение левой части правил $aA \to aB$ и $aA \to aa$, а также левой части правила $Bb \to Ab$. Можно использовать любое из них (но какое-нибудь одно!): если мы заменим (в данном случае единственное) вхождение цепочки aA правой частью правила $aA \to aB$, то получим цепочку $aBBb$ и т.д. Таким образом мы и строим так называемые выводы — последовательности цепочек, в которых каждый последующий член получается из предыдущего заменой, подобной только что рассмотренной.

Определение 7.6. ***Цепочка δ непосредственно выводима*** в грамматике G из цепочки γ (или из γ непосредственно выводится δ), если существуют такие цепочки σ и ρ и такое правило вывода $\alpha \to \beta$ из P, что $\gamma = \sigma\alpha\rho$ (т.е. существует *вхождение* левой части правила вывода в цепочку γ), а $\delta = \sigma\beta\rho$.

Бинарное отношение на множестве цепочек в объединенном алфавите, которое состоит из всех упорядоченных пар (γ, δ), таких, что вторая цепочка непосредственно выводится из первой, называют ***отношением непосредственной выводимости***. Его будем обозначать символом $\vdash_G$, опуская зачастую имя грамматики: $\gamma \vdash \delta$. В этом случае говорят также, что правило $\alpha \to \beta$ применяется к цепочке γ (и что цепочка δ получена применением правила $\alpha \to \beta$ к цепочке γ или, что равносильно, в результате замены данного *вхождения* левой части правила $\alpha \to \beta$ его правой частью).

В общем случае, если левая часть правила вывода входит в данную цепочку, говорят, что правило применимо к этой цепочке (или имеет место ***применимость правила вывода*** к цепочке). Если же левая часть правила вывода не входит в цепочку, то говорят, что правило не применимо к данной цепочке.

Рефлексивно-транзитивное замыкание отношения непосредственной выводимости обозначаем $\vdash_G^*$ (опять-таки опуская имя грамматики G, если это не приводит к недоразумению: $\vdash^*$) и называем ***отношением выводимости***. Иначе говоря, ***выводимость цепочки δ из цепочки γ***, $\gamma \vdash_G^* \delta$, в грамматике G имеет место, по определению, тогда и только тогда, когда найдутся такие слова $\alpha_0 = \gamma, \alpha_1, \ldots, \alpha_n = \delta$, где $n \geqslant 0$, что

$$\alpha_0 \vdash_G \alpha_1 \vdash_G \ldots \vdash_G \alpha_{n-1} \vdash_G \alpha_n.$$

Для данной цепочки γ в объединенном алфавите, если к ней применимо хотя бы одно правило вывода рассматриваемой грамматики, существует в общем случае не одна, а много цепочек, непосредственно выводимых из нее. Эта неоднозначность

связана с двумя факторами. Во-первых, может существовать несколько разных правил вывода, применимых к цепочке γ. Так, для грамматики примера 7.3 к цепочке $aAaBb$ применимы все правила грамматики, кроме первого. Тогда любая цепочка, полученная применением к указанной цепочке любого из этих правил, будет непосредственно выводима из нее. Во-вторых, если уже фиксировано правило вывода, применимое к цепочке γ, то существуют, вообще говоря, несколько различных вхождений левой части этого правила в цепочку γ. Тогда любая цепочка, полученная заменой любого из этих вхождений правой частью выбранного правила вывода, будет непосредственно выводима из γ. Для грамматики примера 7.3, правила $aB \to Ba$ и цепочки $\gamma = baBaBa$ получим: $baBaBa \vdash bBaaBa$ и $baBaBa \vdash baBBaa$.

Разные вхождения левой части некоторого правила в заданную цепочку могут и пересекаться. Так, правило $aBa \to bBb$ грамматики примера 7.3 можно к написанному выше слову γ применить двояко: $baBaBa \vdash bbBbBa$ и $baBaBa \vdash baBbBb$.

Заметим, наконец, что множество цепочек, непосредственно выводимых из данной, пусто тогда и только тогда, когда среди правил вывода грамматики нет ни одного, применимого к данной цепочке. Для примера 7.3 в качестве такой цепочки можно взять хотя бы цепочку aBA.

Определение 7.7. ***Выводом в грамматике*** G называют произвольную, конечную или бесконечную, последовательность слов α_0, α_1, ..., α_n, ..., в которой для любого $i \geqslant 0$ $\alpha_i \vdash \alpha_{i+1}$, если цепочка α_{i+1} существует. Число $n \geqslant 0$ называют ***длиной вывода***, если указанная последовательность конечна и α_n — ее последний элемент. В этом случае говорят также, что α_n выводится из α_0 за n шагов. Для положительного n пишем $\alpha_0 \vdash^+ \alpha_n$ или, если нужно специально оговорить длину вывода $\alpha_0 \vdash^n \alpha_n$.

Из определений следует, что выводимость $\gamma \vdash^* \delta$ равносильна тому, что существует вывод $\gamma = \alpha_0, \alpha_1, \ldots, \alpha_n = \delta$, где $n \geqslant 0$.

Мы будем использовать также термин „***фрагмент вывода***“, понимая под этим такую подпоследовательность вывода, которая сама является выводом. В частности, фрагмент вывода, состоящий из двух соседних его членов, называется ***шагом вывода***.

Построение вывода в порождающей грамматике можно трактовать как „игру“, правила которой заданы множеством P. Подобно тому как в игре (например, в шахматах) можно из заданной позиции перейти в одну из множества следующих позиций, используя правила игры, так и при проведении вывода в грамматике можно из данной цепочки получить одну из множества непосредственно выводимых из нее, заменяя в ней произвольное вхождение левой части правила его правой частью. При этом мы можем выбрать любое правило, применимое к очередной цепочке, и среди различных вхождений его левой части выбрать любое.

На основании сказанного можно предположить, что существуют несколько различных выводов фиксированной цепочки β из фиксированной цепочки α. Далее на примерах мы убедимся в том, что это действительно так.

Введенные выше понятия непосредственной выводимости, выводимости и вывода можно уподобить известным из теории графов понятиям *непосредственной достижимости*, *достижимости* и *пути* (в *ориентированных графах*) соответственно. Мы специально выбирали обозначения выводимости близкими к тем, которые используют в теории графов. Нужно только заметить, что множество вершин графа конечно (по крайней мере, мы в этой книге изучаем только конечные графы), а множество слов бесконечно. Но графовая аналогия позволяет придать введеным выше понятиям большую наглядность: так, построение вывода в грамматике можно уподобить „путешествию“ по некоторому пути в ориентированном графе.

Центральным понятием в теории порождающих грамматик является понятие языка, порождаемого данной грамматикой.

Определение 7.8. ***Язык, порождаемый грамматикой*** G, — это множество $L(G)$ всех выводимых из аксиомы грамматики *терминальных цепочек*:

$$L(G) = \{x\colon S \vdash_G^* x, x \in V^*\}.$$

Продолжая „игровую аналогию“, мы можем теперь сказать, что у „игры“, во-первых, появилась выделенная „начальная позиция“, аксиома; во-вторых, возникла „цель“ — строя вывод в согласии с правилами грамматики, получить цепочку, не содержащую нетерминалов. Так, скажем, и в шахматах, мы начинаем игру не с произвольной позиции, а со вполне определенной начальной позиции, предусматривающей строго фиксированную расстановку фигур, а закончить игру стремимся в любой позиции, в которой вражескому королю поставлен мат. Но при этом нужно понимать, что правила вывода грамматики априори никак не связаны с ограничением строить выводы терминальных цепочек из аксиомы, а определяют возможность построения любого вывода, даже бесконечного.

Заметим также, что, согласно определению 7.8, нетерминалы не содержатся в цепочках языка, порождаемого грамматикой. Это совсем не значит, что нетерминалы „не нужны“, напротив, с их помощью мы организуем вывод и можем получить требуемые терминальные цепочки. Когда мы решаем математическую расчетную задачу и должны в результате получить число, то это не значит, что мы не должны пользоваться формулами. Но все буквенные обозначения в окончательном результате должны исчезнуть, хотя без них этот результат получить невозможно.

Определение 7.9. Две грамматики называются ***эквивалентными***, если они порождают один и тот же язык.

Примем соглашение о следующей сокращенной записи правил с одинаковой левой частью: вместо записи

$$A \to \alpha_1, \quad A \to \alpha_2, \quad \ldots, \quad A \to \alpha_n$$

будем использовать запись

$$A \to \alpha_1 | \alpha_2 | \ldots | \alpha_n,$$

разделяя разные правые части вертикальной чертой.

Кроме того, рассматривая примеры грамматик, мы зачастую будем записывать только их множества правил вывода, избегая скрупулезной записи всего кортежа, определяющего грамматику. С учетом введенных выше соглашений об обозначениях это не должно приводить к недоразумениям.

Пример 7.4. Рассматриваемая ниже грамматика моделирует простейший фрагмент естественного (русского) языка: ее терминалы — это некоторые слова русского языка*, нетерминалами являются „грамматические категории", а именно понятия „предложение", „подлежащее" и „сказуемое" (они даны как слова, заключенные в угловые скобки):

$$G_1 = (\{\text{кот, пес, крокодил, мяукает, лает, плачет}\},$$
$$\{\langle\text{предложение}\rangle, \langle\text{подлежащее}\rangle, \langle\text{сказуемое}\rangle\},$$
$$\langle\text{предложение}\rangle, P_1).$$

Множество правил P_1 имеет вид

$$\langle\text{предложение}\rangle \to \langle\text{подлежащее}\rangle\ \langle\text{сказуемое}\rangle,$$
$$\langle\text{подлежащее}\rangle \to \text{кот} | \text{пес} | \text{крокодил},$$
$$\langle\text{сказуемое}\rangle \to \text{мяукает} | \text{лает} | \text{плачет}.$$

Каждое правило вывода грамматики G_1 можно трактовать как определение той или иной грамматической категории: например, первое правило есть запись такого определения: „предложение — это подлежащее, за которым идет сказуемое". Так как нас интересует только *синтаксис языка*, то это определение,

*Эти слова рассматриваются как неделимые символы, а именно буквы данного терминального алфавита. Нас никак не должно это смущать, ибо алфавит — это любое непустое конечное множество.

касающееся исключительно записи предложения, есть требование, согласно которому, записывая предложение, сначала нужно поставить подлежащее, а после него — сказуемое. Другие правила грамматики определяют подобным образом „подлежащее“ и „сказуемое“. Но тут новых грамматических категорий не возникает — просто перечисляются те слова, которые могут быть подлежащим и сказуемым.

Построим какой-нибудь вывод в грамматике G_1:

$$\langle\text{предложение}\rangle \vdash \langle\text{подлежащее}\rangle\langle\text{сказуемое}\rangle \vdash$$
$$\vdash \text{кот}\ \langle\text{сказуемое}\rangle \vdash \text{кот лает}.$$

Заметим, что „смысл“ (*семантика*) выводимой цепочки нас никак не интересует. Мы вообще пока не знаем, что такое „смысл“. Так что кот может лаять, крокодил мяукать и т.д. #

Приведенный пример, несмотря на его простоту, позволяет нам дать еще одну содержательную интерпретацию понятия грамматики. Грамматику можно рассматривать как „систему определений“ некоторых „терминов“, „понятий“. Выделяется самое общее понятие (в данном примере понятие „предложение“), оно сводится к менее общим „понятиям“ до тех пор, пока мы не придем к „конкретным объектам“ (в данном случае к цепочкам в каком-то алфавите), подпадающим под определяемые „понятия“. Самое общее „понятие“ — это аксиома, другие „понятия“ — нетерминалы, „конкретные объекты“ — терминальные цепочки. В подобном духе строится определение синтаксиса языков программирования с помощью так называемых форм Бэкуса — Наура. Пример такого описания приведен ниже (см. **Д.8.2**). За другими примерами следует также обратиться к литературе по языкам программирования.

Пример 7.5. а. Грамматика

$$G_2 = (\{a, b, 0, 1\},\ \{I, D\},\ I,\ P_2)$$

имеет множество правил P_2:

$$I \to aD|bD|a|b,$$
$$D \to aD|bD|0D|1D|a|b|0|1.$$

Нетрудно видеть, что данная грамматика порождает простейшие „идентификаторы“ в алфавите, состоящем из двух „букв“ и двух „цифр“, т.е. цепочки, начинающиеся обязательно с „буквы“.

Примеры выводов в грамматике G_2:

$$I \vdash aD \vdash abD \vdash ab0D \vdash ab0bD \vdash ab0b1,$$
$$I \vdash a,\ I \vdash b.$$

б. Грамматика

$$G_3 = \langle\{(,)\}, \{S\}, S, S \to ()|(S)|SS\rangle$$

порождает множество так называемых „правильных скобочных структур“*, например $S \vdash SS \vdash (S)S \vdash (())S \vdash (())()$.

Полезно сопоставить с этой грамматикой индуктивное определение множества правильных скобочных структур: цепочка () есть правильная скобочная структура; если известно, что цепочки x и y — правильные скобочные структуры, то цепочку xy также считаем правильной скобочной структурой; если известно, что x — правильная скобочная структура, то и цепочка (x) есть правильная скобочная структура; никаких правильных скобочных структур, кроме определенных выше, не существует. Видно, что грамматика G_3 есть не что иное, как форма записи сформулированного только что индуктивного определения: „правильная скобочная структура“ (понятие, обозначенное аксиомой S) есть либо цепочка (), либо „правильная

*Мы заключили четверку, задающую грамматику G_3, в угловые, а не в круглые скобки, чтобы не смешивать их со скобками, являющимися терминалами данной грамматики.

скобочная структура" в скобках — (S), либо две „правильные скобочные структуры", записанные одна после другой, — SS.

в. Рассмотрим грамматику

$$G_4 = (\{a, b, c\},\ \{S, A, B, C\},\ S,\ P_4)$$

с множеством правил вывода P_4:

$$\begin{array}{ll} S \to aSBC \,|\, abC, & (1)\,|\,(2) \\ CB \to BC, & (3) \\ bB \to bb, & (4) \\ bC \to bc, & (5) \\ cC \to cc. & (6) \end{array}$$

Разберем некоторые примеры выводов: под символом $\vdash$ будем указывать номер применяемого на данном шаге правила, а над символом — количество повторений этого правила:

$S \vdash_2 abC \vdash_5 abc$;

$$S \vdash_1 aSBC \vdash_2 aabCBC \vdash_3 aabBCC \vdash_4 \vdash_4 aabbCC \vdash_5 aabbcC \vdash_6 aabbcc;$$

$$S \vdash_1 aSBC \vdash_1 aaSBCBC \vdash_2 aaabCBCBC \vdash_3 \vdash_3 aaabBCCBC \vdash_3 aaabBCBCC \vdash_3 aaabBBCCC \vdash_4 \vdash_4 aaabbBCCC \vdash_4 aaabbbCCC \vdash_5 aaabbbcCC \vdash_6^2 aaabbbccc.$$

Представим теперь вывод произвольной цепочки $a^n b^n c^n$:

$$S \vdash_1 aSBC \vdash_1 aaSBCBC \vdash_1 \ldots \vdash_1 a^{n-1}S(BC)^{n-1} \vdash_2 \vdash_2 a^n bC \underbrace{BCBC\ldots BC}_{n-1} \vdash_3 a^n bBCC \underbrace{BC\ldots BC}_{n-2} \vdash_3 \ldots \ldots \vdash_3 a^n bB^{n-1}C^n \vdash_4 a^n bbB^{n-2}C^n \vdash_4 \ldots \ldots \vdash_4 a^n b^n C^n \vdash_5 a^n b^n cC^{n-1} \vdash_6^{n-1} a^n b^n c^n. \quad (7.2)$$

Можно заметить, что посредством многократного применения правила (3) в выводе (7.2) происходит „перегонка“ всех букв B влево от всех букв C, т.е. из цепочки $a^nbC\underbrace{BCBC\ldots BC}_{n-1}$ выводится цепочка $a^nbB^{n-1}C^n$. Если считать, что правило (3) на каждом шаге соответствующего вывода применяется так, что происходит замена первого вхождения цепочки CB цепочкой BC, то первый (самый левый) символ B в цепочке $a^nbC\underbrace{BCBC\ldots BC}_{n-1}$ „проходит“ через один символ C, следующему символу B (в цепочке $a^nbBCC\underbrace{BC\ldots BC}_{n-2}$) нужно пройти уже два символа C и т.д. Отсюда следует, что правило (3) в указанном фрагменте вывода (7.2) применяется

$$1+2+\ldots+(n-1)=\frac{n(n-1)}{2}\text{ раз.}$$

Тогда последовательность номеров применяемых правил (протокол вывода) может быть записана так:

$$(1)^{n-1}(2)(3)^{\frac{n(n-1)}{2}}(4)^{n-1}(5)(6)^{n-1},\ n\geqslant 1.$$

Гораздо труднее доказать, что данная грамматика порождает только цепочки указанного вида. Проанализируем другие варианты проведения вывода после применения правила (2) в выводе (7.2).

1. После применения правила (2) применяем правило (5) и после многократного применения правила (3) получаем

$$a^nbC(BC)^{n-1}\vdash_5 a^nbc(BC)^{n-1}=$$
$$=a^nbcBCBC\ldots BC\vdash_3^+ a^nbcB^{n-1}C^{n-1}. \quad (7.3)$$

Цепочка, записанная последней, такова, что, во-первых, она не является терминальной, а во-вторых, к ней не применимо ни

одно правило грамматики. Любая такая цепочка называется ***тупиковой***.

2. Вывод (7.2) можно модифицировать, прервав на определенном шаге применение правила (3) и применив правило (4):

$$a^n bB^m C^{m+1}(BC)^{n-1-m} =$$

$$= a^n bB^m C^{m+1} BC \ldots BC \vdash_4 a^n bbB^{m-1} C^{m+1}(BC)^{n-1-m},$$

где $m < n-1$.

Далее, если посредством применения правила (3) мы „перегоним“ все символы B влево, то получим цепочку $a^n bbB^{n-2}C^n$, из которой легко получается цепочка $a^n b^n c^n$. Таким образом, мы получили другой вариант вывода этой цепочки.

Но если мы будем применять правило (4), начиная с цепочки

$$a^n bB^m C^{m+1} \underbrace{BC \ldots BC}_{n-1-m},$$

до тех пор, пока это возможно, то получим цепочку

$$a^n b^{m+1} C^{m+1}(BC)^{n-1-m}.$$

Из нее, если не применять сразу правило (5), снова получится цепочка $a^n b^n c^n$. Применение же правила (5) приведет к тупиковой цепочке.

По аналогии можно рассмотреть произвольное чередование применения правил (3) и (4).

Все варианты вывода терминальной цепочки тем самым разобраны.

В общем же случае строгое доказательство того, что какая-либо предъявленная нам грамматика не порождает никаких других цепочек, кроме цепочек определенного вида, может быть весьма нетривиальным.

г. Зададим грамматику

$$G_5 = (\{a\},\ \{S, A, C, D, E, F\},\ S,\ P_5),$$

множеством P_5 правил

$$\begin{aligned}
&S \to aCA,\\
&A \to a^2EA \mid F,\\
&EF \to DF,\\
&ED \to Da^2E,\\
&Ea \to aE,\\
&Ca \to aC,\\
&CD \to Ca,\\
&CF \to \lambda.
\end{aligned}$$

Можно доказать, что данная грамматика порождает язык $\{a^{n^2}: n \geqslant 1\}$, т.е. все квадраты натуральных чисел, закодированных в однобуквенном алфавите. #

При доказательстве утверждений о грамматиках нам будет удобно использовать одну процедуру, которую мы назовем ***переименованием нетерминалов грамматики.***

Пусть дана какая-то грамматика $G = (V, N, S, P)$. Введем новый алфавит N', не пересекающийся с $V \cup N$, так, что между алфавитами N и N' установлено взаимно однозначное соответствие, при котором каждый символ $A \in N$ переходит в символ $A' \in N'$. Построим новую грамматику $G' = (V, N', S', P')$, заменив каждое правило в исходной грамматике новым, в котором на месте каждого вхождения каждого нетерминала $A \in N$ поставлен нетерминал $A' \in N'$. Нетрудно доказать, что построенная таким образом грамматика G' эквивалентна исходной. Говоря неформально, описанное преобразование грамматики состоит в переобозначении ее нетерминалов таким образом, что вместо каждого нетерминала ставится его „двойник“, причем „двойники“, соответствующие разным нетерминалам, различны.

7.3. Классификация грамматик и языков

Напомним, что единственное ограничение, накладываемое на правило вывода любой грамматики, состоит в том, что в *левую часть* правила должен входить хотя бы один *нетерминал*. В зависимости от дополнительных ограничений, накладываемых на *правила вывода грамматики*, различают следующие основные классы грамматик.

1. ***Грамматики типа 0***, или ***грамматики общего вида***. Здесь на правила вывода не накладывается никаких дополнительных ограничений.

2. ***Неукорачивающие грамматики***. Каждое правило такой грамматики имеет вид $\alpha \to \beta$, где $|\alpha| \leqslant |\beta|$. Таким образом, длина правой части правила не меньше длины левой. Грамматика G_4 из примера 7.5 есть неукорачивающая грамматика, но грамматика G_5 из того же примера таковой не является.

3. ***Контекстно-зависимые грамматики*** (***КЗ-грамматики***). Грамматику называют контекстно-зависимой грамматикой (КЗ-грамматикой), если любое ее правило вывода имеет вид $\varphi A \psi \to \varphi \xi \psi$, где A — *нетерминал*, ξ — некоторая цепочка, $\xi \neq \lambda$. Каждое такое правило, называемое ***КЗ-правилом***, позволяет заменить нетерминал A в „контексте“, образуемом цепочками φ и ψ в *объединенном алфавите*, непустой цепочкой ξ (см. замечание 7.1). Иногда цепочку φ называют ***левым контекстом***, а цепочку ψ — ***правым контекстом*** данного ***КЗ-правила***. Из определения видно, что каждая КЗ-грамматика является неукорачивающей.

Грамматика G_4 из примера 7.5 не является КЗ-грамматикой, так как правило $CB \to BC$ не является КЗ-правилом. Остальные же правила вывода этой грамматики — КЗ-правила. Грамматика G_5 из примера 7.5 не является КЗ-грамматикой хотя бы из-за наличия правила вывода с пустой правой частью. КЗ-грамматикой является грамматика из примера 7.3.

Если в КЗ-правиле снять требование непустоты цепочки ξ, то получим грамматику, которую называют ***обобщенной КЗ-грамматикой*** (или, коротко, ***ОКЗ-грамматикой***).

4. ***Контекстно-свободные грамматики*** (***КС-грамматики***). Каждое правило такой грамматики имеет вид $A \to \alpha$, т.е. левая часть каждого правила вывода есть нетерминал, а правая — произвольная (может быть и пустая) цепочка в объединенном алфавите.

С практической точки зрения это наиболее важный класс грамматик, поскольку именно в терминах КС-грамматик описывается синтаксис языков программирования, и этим грамматикам будет посвящена отдельная глава. Грамматики G_2, G_3, G_4 из примера 7.5 являются КС-грамматиками.

5. ***Линейные грамматики.*** Каждое правило такой грамматики имеет вид $A \to uBv$ или $A \to u$, т.е. в правой части правила может содержаться не более одного вхождения нетерминала. Если во всех правилах вида $A \to uBv$ имеет место $v = \lambda$, то грамматика называется ***праволинейной***, а если $u = \lambda$ — ***леволинейной***.

Пример 7.6. Линейной является грамматика

$$G_6 = (\{a_1, \ldots, a_n\},\ \{S\}, S, P_6),$$

где множество правил вывода P_6 есть

$$S \to a_1 S a_1 \,|\, a_2 S a_2 \,|\, \ldots a_n S a_n \,|\, a_1 \,|\, a_2 \,|\, \ldots a_n \,|\, \lambda.$$

Можно доказать, что эта грамматика порождает все палиндромы в алфавите V, т.е. все цепочки, читаемые слева направо так же, как и справа налево. Например, для $V = \{a, b, c\}$ цепочка $acbbca$ — палиндром. Вывод его в грамматике G_6 (для данного терминального алфавита) будет иметь вид

$$S \vdash aSa \vdash acSca \vdash acbSbca \vdash acb\lambda bca = acbbca.$$

Замечание 7.2. Формальное определение палиндрома таково. Назовем ***инверсией*** непустой ***цепочки***

$$x = x(1)x(2)\dots x(k-1)x(k) \in V$$

цепочку

$$x^R = x(k)x(k-1)\dots x(2)x(1).$$

Для пустой цепочки λ по определению считаем $\lambda^R = \lambda$. ***Палиндром*** в алфавите V — это любая цепочка x, для которой $x^R = x$.

6. ***Регулярные грамматики.*** У такой грамматики каждое правило имеет вид $A \to aB$ или $A \to a$, где a есть либо терминал, либо пустая цепочка. Регулярные грамматики — частный случай праволинейных грамматик.

Эти грамматики подробно будут рассмотрены в **7.4**.

Приведем без доказательства некоторые утверждения о классах грамматик.

Теорема 7.2. 1. Для любой грамматики типа 0 может быть построена *эквивалентная* ей ОКЗ-грамматика.

2. Для любой неукорачивающей грамматики может быть построена эквивалентная ей КЗ-грамматика.

3. Для любой леволинейной грамматики может быть построена эквивалентная ей праволинейная грамматика, и, наоборот, для любой праволинейной грамматики может быть построена эквивалентная ей леволинейная грамматика.

4. Для любой праволинейной грамматики может быть построена эквивалентная ей регулярная грамматика. #

Отметим, что доказательства первых двух пунктов теоремы 7.2 весьма нетривиальны*.

Классификация языков, порождаемых грамматиками, тесно связана с классификацией самих грамматик, хотя и не тождественна ей. Язык относят к ***классу*** C, если существу-

*См.: *Глушков В.М., Цейтлин Г.Е., Ющенко Е.Л.*

ет грамматика класса C, порождающая данный язык. Таким образом определяются ***языки типа 0***, ***неукорачивающие языки***, ***контекстно-зависимые языки*** (***КЗ-языки***), ***обобщенные контекстно-зависимые языки*** (***ОКЗ-языки***), ***контекстно-свободные языки*** (***КС-языки***), ***линейные языки***, ***право-*** и ***леволинейные языки***, ***регулярные языки***.

Так как всякая ОКЗ-грамматика является грамматикой типа 0, то в соответствии с п. 1 теоремы 7.2 классы языков типа 0 и ОКЗ-языков совпадают. В силу п. 2 того же утверждения, а также ввиду того, что любая КЗ-грамматика является неукорачивающей грамматикой, совпадают классы неукорачивающих и КЗ-языков. В силу пп. 3 и 4 теоремы 7.2 совпадают классы право-, леволинейных и регулярных языков.

Но можно доказать следующие факты.

Теорема 7.3. 1. Существует ОКЗ-язык, не являющийся КЗ-языком.

2. Существует КЗ-язык, не являющийся КС-языком.

3. Существует КС-язык, не являющийся линейным языком.

4. Существует линейный язык, не являющийся регулярным языком. #

Некоторые из утверждений теоремы 7.3 мы докажем позже. Сейчас заметим только, что языки, порождаемые грамматиками G_4 и G_5 из примера 7.5, не являются КС-языками, хотя для языка, порождаемого грамматикой G_5, можно построить порождающую его КЗ-грамматику. Язык правильных скобочных структур, порождаемый грамматикой G_3 из примера 7.5, не является линейным языком, а язык палиндромов из примера 7.6 не есть регулярный язык.

Итак, можно утверждать, что имеют место следующие строгие включения классов языков:

$$\text{РЕГ} \subset \text{ЛИН} \subset \text{КС} \subset \text{ОКЗ} = \text{ТИП } 0; \quad \text{КЗ} \subset \text{ОКЗ},$$

где РЕГ, ЛИН, КС, КЗ, ОКЗ, ТИП 0 — классы регулярных, линейных, КС-языков, КЗ-языков, ОКЗ-языков и языков типа 0 соответственно.

Замечание 7.3. В силу п. 1 теоремы 7.3 требование, состоящее в том, чтобы в КЗ-правиле $\varphi A\psi \to \varphi\xi\psi$ цепочка ξ была непустой, является принципиальным. В **8.2** мы докажем, что с определенными оговорками класс КС-языков можно включить в класс КЗ-языков, поскольку любую КС-грамматику можно преобразовать к эквивалентной КС-грамматике, не содержащей правила вывода вида $A \to \lambda$. #

Принципиальное различие между классификацией грамматик и языков состоит в следующем. Чтобы определить класс грамматики, достаточно посмотреть на множество ее правил вывода. Чтобы доказать „положительное" утверждение о том, что заданный язык есть язык такого-то класса, достаточно придумать любую грамматику из соответствующего класса, которая его порождает. Но чтобы доказать „отрицательное" утверждение о классе языка, т.е. доказать, что язык не принадлежит такому-то классу языков, нужно доказать, что не существует грамматики соответствующего класса, которая его порождает. Эта задача гораздо труднее. Некоторые методы построения подобных доказательств будут рассморены далее.

7.4. Регулярные языки и регулярные выражения

В *замкнутом полукольце* $\mathcal{L}(V)$ всех языков в алфавите V рассмотрим *подалгебру, порожденную множеством*, которое состоит из пустого языка, языка $\{\lambda\}$, всех языков $\{a\}$, $a \in V$ (каждый из которых содержит единственную однобуквенную цепочку a), и *замкнутую относительно итерации*. Эта подалгебра, обозначаемая $\mathcal{R}(V)$, есть *полукольцо с итерацией*. Оно играет важнейшую роль в теории формальных языков. Его называют ***полукольцом регулярных языков***. Далее будет

доказано, что элементы полукольца $\mathcal{R}(V)$ являются в точности *регулярными языками*, т.е. языками, порождаемыми *регулярными грамматиками*.

Теорема 7.4. Язык в алфавите V регулярен тогда и только тогда, когда он является элементом полукольца $\mathcal{R}(V)$. #

Здесь мы не будем доказывать эту теорему, а выведем ее позже, опираясь на другие теоремы о регулярных языках (см. **7.5**).

Таким образом, множество регулярных языков в алфавите $V = \{a_1, \ldots, a_n\}$ есть не что иное, как *замыкание* конечного множества $\{\varnothing, \{\lambda\}, \{a_1\}, \ldots, \{a_n\}\}$ *относительно операций* объединения, соединения и итерации. Следовательно, как и всякое Ω-*замыкание*, оно может быть описано индуктивно (см. **4.1**), а именно:

1) пустое множество $\varnothing$, множество $\{\lambda\}$ (состоящее из одной пустой цепочки) и множество $\{a\}$ для каждого $a \in V$ считаем регулярным языком в алфавите V;

2) если известно, что P и Q — регулярные языки в алфавите V, то к множеству регулярных языков в алфавите V следует добавить *объединение* $P \cup Q$ и *соединение* PQ;

3) если известно, что P — регулярный язык в алфавите V, то к множеству регулярных языков в алфавите V следует добавить его итерацию P^*;

4) никаких других регулярных языков, кроме определенных в пп. 1–3, не существует.

Из определения регулярного языка, теоремы 7.4 и следствия 7.1 немедленно вытекает, что и *позитивная итерация* регулярного языка регулярна.

Далее мы зачастую будем говорить просто о регулярных языках (или регулярных множествах), подразумевая, что фиксирован некоторый алфавит V.

Алгебраические операции над регулярными языками удобно представлять с помощью так называемых ***регулярных выражений***. Каждое регулярное выражение представляет (или

обозначает) некоторый (однозначно определяемый) регулярный язык, причем языки $\varnothing$, $\{\lambda\}$ и $\{a\}$, где $a \in V$, обозначаются выражениями $\varnothing$, λ и a соответственно, и если регулярное выражение p обозначает регулярный язык P, а регулярное выражение q обозначает регулярный язык Q, то регулярные выражения $(p+q)$, (pq) и (p^*) обозначают регулярные множества $P \cup Q$, PQ и P^* соответственно. Таким образом, в регулярных выражениях для обозначения операции объединения языков используют знак „+“ (плюс).

Условимся также для регулярного выражения $\alpha\alpha^*$ или $\alpha^*\alpha$ использовать обозначение α^+ и называть это выражение позитивной итерацией выражения α.

Замечание 7.4. Введенные выше обозначения регулярных выражений создают при использовании символа λ некий „конфликт“ обозначений. А именно мы можем, в зависимости от контекста, понимать этот символ и как обозначение самой пустой цепочки, и как обозначение языка, состоящего из одной пустой цепочки (это и будет собственно регулярное выражение λ). Эти интерпретации символа λ следует тщательно разграничивать. В то же время (и такова традиция изложения теории формальных языков) вряд ли целесообразно вводить для регулярного выражения, обозначающего язык $\{\lambda\}$, какое-то новое обозначение. #

Можно сократить количество скобок в регулярных выражениях*, приняв следующее соглашение о приоритетах: самый высокий приоритет имеет операция итерации, затем — соединения и, наконец, — объединения. Так, регулярное выражение $a^* + (bc)^*$ обозначает множество цепочек, состоящее из цепочек вида a^n, $n \geqslant 0$, и цепочек вида $(bc)^n$, $n \geqslant 0$, где a, b, $c \in V$. Использование регулярных выражений позволяет получать более наглядную и простую нотацию для регулярных языков.

*По аналогии с тем, как мы делали это в *формулах*, представляющих *булевы функции* (см. 6.4).

Так, вместо рассмотренного выше регулярного выражения мы должны были бы использовать гораздо менее выразительную и более громоздкую формулу: $\{a\}^* \cup (\{b\} \cdot \{c\})^*$.

Соответствие между регулярными множествами и регулярными выражениями не является взаимно однозначным: одно и то же регулярное множество может представляться многими регулярными выражениями. Продемонстрируем это на таком примере.

Рассмотрим регулярное выражение

$$x = (b^+a)^*(b^+a + b^*), \quad a, b, c \in V.$$

Используя *аксиомы полукольца* (см. **3**), преобразуем x следующим образом:

$$x = (b^+a)^+ + (b^+a)^*b^* = (b^+a)^+ + (b^+a)^* + (b^+a)^*b^+ = \\ = (b^+a)^* + (b^+a)^*b^+ = (b^+a)^*b^*.$$

Все ***регулярные выражения***, фигурирующие в этих преобразованиях, ***эквивалентны***, т.е. все они обозначают один и тот же регулярный язык. Проблемы распознавания эквивалентности двух произвольных регулярных выражений, автоматизации тождественных преобразований регулярных выражений и поиск самого короткого („оптимального“) регулярного выражения, обозначающего данный регулярный язык, весьма трудны и здесь не обсуждаются. В целом соотношение между регулярными выражениями и регулярными языками вполне аналогично соотношению между *формулами* и *булевыми функциями* (см. **6.4**). В частности, если переход от формулы к эквивалентной формуле в теории булевых функций совершается согласно *аксиомам булевой алгебры* и другим тождествам, выводимым из этих аксиом, то переход от заданного регулярного выражения к эквивалентному регулярному выражению производится согласно аксиомам полукольца и тождествам, выводимым из них.

Зачастую в дальнейшем, если это не повлечет непонимания, мы будем отождествлять регулярный язык с обозначающим его регулярным выражением (любым!), что позволит не вводить новых обозначений и пояснений. Так, для рассмотренного выше примера мы можем написать $bbababbb \in (b^+a)^*b^*$, что, строго говоря, обозначает факт принадлежности цепочки $bbababbb$ языку, обозначенному регулярным выражением $(b^+a)^*b^*$. Разумеется, следует воздерживаться, например, от употребления такой записи: $\lambda \in \lambda$, хотя, если подумать, и здесь все понятно: пустая цепочка λ принадлежит языку $\{\lambda\}$, обозначаемому регулярным выражением λ.

7.5. Конечные автоматы. Теорема Клини

Одной из наиболее важных задач, решаемых в теории формальных языков, является следующая.

Пусть задана некоторая *порождающая грамматика* G с *терминальным алфавитом* V и *цепочка* x в алфавите V. Спрашивается: принадлежит ли цепочка x *языку, порождаемому грамматикой* G? Эту задачу называют ***проблемой принадлежности*** для грамматики G. В теории формальных языков доказывается, что проблема принадлежности разрешима для КЗ-грамматик и для КС-грамматик, но в общем случае не разрешима для грамматик типа 0. Решение проблемы принадлежности состоит в разработке распознающего алгоритма, который для произвольных грамматики G (из заданного класса грамматик) и цепочки x за конечное число шагов выдает ответ на поставленный выше вопрос. В основе подобного рода алгоритмов лежит математическая модель языка, называемая ***распознающей моделью*** или ***анализирующей моделью*** и являющаяся как бы зеркальной к ***порождающей модели***, примером которой служит порождающая грамматика. Традиционно анализирующие модели языков называют ***автоматами***. В каждом классе языков может быть определена пара

„порождающая модель — анализирующая модель“ или, другими словами, пара „грамматика — автомат“, где автомат в определенном смысле анализирует (распознает) цепочки, порождаемые грамматикой.

Неформально автомат можно описать как устройство, состоящее из ***блока управления***, ***входной ленты***, ***головки автомата*** и блока ***внутренней памяти автомата*** (рис. 7.2). На ленте, которая предполагается полубесконечной (не ограниченной справа) и разделена на ячейки, записываются цепочки во ***входном алфавите*** (обозначаемом далее V) так, что буквы цепочки занимают последовательно, начиная с первой и без пропусков, ячейки ленты — по одной букве в каждой ячейке. Цепочку, записанную таким образом на входной ленте автомата, будем называть ***входной цепочкой***. Блок управления может в каждый момент времени находиться в одном из конечного множества ***состояний*** (обозначим его через Q), а головка может быть установлена в точности на одну ячейку входной ленты. В таком случае говорят, что автомат обозревает данную ячейку.

Автомат, читая входную цепочку, работает по шагам (или по тактам). Пусть в некоторый момент времени автомат обозревает какую-то ячейку ленты, а блок управления находится в некотором состоянии $q \in Q$. ***Такт работы автомата***

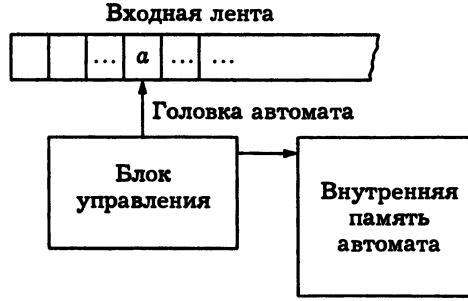


Рис. 7.2

состоит в том, что в зависимости от содержимого обозреваемой ячейки, состояния q, а также содержимого внутренней памяти автомат сдвигает головку вправо или влево на одну ячейку либо оставляет ее на прежнем месте, его блок управления переходит в некоторое новое состояние r (возможно, что это состояние совпадет с исходным состоянием q), а содержимое внутренней памяти определенным образом модифицируется. В общем случае автомат может и поменять содержимое той ячейки ленты, которую он обозревает (т.е. автомат может работать с лентой или только в режиме чтения, не меняя ее содержимого, а лишь определенным образом продвигая головку, или в режиме чтения/записи).

Вводят понятие ***конфигурации автомата***: она определяется состоянием блока управления, содержимым обозреваемой ячейки и содержимым внутренней памяти*. Автомат в общем случае не является детерминированным устройством, т.е. для него из заданной конфигурации возможны переходы в несколько различных конфигураций. Правила, согласно которым автомат переходит из одной конфигурации в другую, составляют в совокупности его ***систему команд автомата***. Каждая команда разрешает переход из некоторой конфигурации в какую-то другую конфигурацию. Это напоминает игру, например шахматную, когда из текущей позиции на доске можно, сделав ход, получить новую позицию — одну из множества всех позиций, в которые можно попасть из текущей позиции, сделав ход согласно правилам игры. В данном случае правила игры аналогичны системе команд, а позиция на доске — конфигурации автомата.

Автоматы классифицируются в соответствии со структурой своей внутренней памяти, с режимом работы с лентой (только чтение или чтение/запись), а также с типом движения

*Для автоматов конкретных классов понятие конфигурации несколько модифицируется: так, например, в конфигурацию входит не только символ обозреваемой ячейки, но и вся подцепочка входной цепочки, включающая символ обозреваемой ячейки и все символы справа от него.

головки по ленте (одностороннее, двустороннее, число ячеек, на которые за один такт можно сдвинуть головку). Множество команд предполагается конечным, т.е. автомат, как и грамматика, имеет конечное описание.

Представим теперь следующую ситуацию. Пусть на входной ленте автомата записана некоторая цепочка $x \in V^*$. Допустим также, что среди состояний блока управления выделено некоторое специальное состояние, называемое ***начальным***, а также некоторое подмножество состояний, которые называются ***заключительными***.

В начальный момент времени блок управления находится в начальном состоянии, головка обозревает первую (крайнюю левую) ячейку ленты, в которой записан первый символ входной цепочки x. Читая цепочку x и делая один такт за другим, автомат, после того как он прочитает последнюю букву цепочки, окажется в каком-то состоянии q' (точнее говоря, в этом состоянии окажется блок управления). Если это состояние является заключительным, то тогда говорят, что автомат допустил цепочку x. Спрашивается: каким образом посредством грамматики можно описать множество всех цепочек в алфавите V, которые автомат допускает?

Оказывается, что каждому классу грамматик соответствует свой класс автоматов, причем для каждой грамматики соответствующего класса может быть построен автомат, который допускает цепочки, порождаемые данной грамматикой, и только их. Образно говоря, в каждом классе языков возникает пара „писатель — читатель“: грамматика, как „писатель“, порождает определенное множество „текстов“ (цепочек в заданном алфавите), а „читатель“ (автомат) проверяет „правильность“ этих текстов, допуская те и только те цепочки, которая порождает „его“ грамматика. В качестве „писателя“ может выступать программист, пишущий тексты компьютерных программ, а в качестве „читателя“ — системные программы, обеспечивающие проверку правильности написанного текста

(соответствия его грамматике того или иного языка программирования). Тем самым допускающий автомат становится прообразом некоторого распознающего алгоритма, решающего проблему принадлежности в том или ином классе грамматик.

Заметим, однако, что автомат сам по себе еще не является таким алгоритмом и что оказывается, например, в классе грамматик типа 0 в общем случае построить алгоритм для решения проблемы принадлежности невозможно, хотя автоматы, соответствующие грамматикам типа 0, существуют (так называемые машины Тьюринга, см. **Д.7.4**). Некоторые вопросы, связанные с переходом от анализирующей модели языка к алгоритму, который решает проблему принадлежности для грамматики, порождающей этот язык, рассмотрены в **Д.8.1**.

Мы начинаем с простейших анализирующих моделей — ***конечных автоматов***. Конечные автоматы — это анализирующие модели для *регулярных языков*. Конечный автомат не имеет внутренней памяти, головка движется по ленте только вправо — на одну ячейку за такт. С ленты можно только читать. Кроме того, автомат может переходить „спонтанно“ из одного состояния в другое, не читая ленту и не продвигая головку. Такой такт можно рассматривать как переход из состояния в состояние по *пустой цепочке*. Его называют λ-тактом.

Итак, из каждого состояния автомат может переходить в другое состояние, читая тот или иной символ входной цепочки, или делать переход по пустой цепочке, причем принимается по определению, что эти два типа переходов исключают друг друга. Поведение конечного автомата определяется его системой команд, в которой каждая команда задается записью

$$qa \to r, \tag{7.4}$$

что означает: „из состояния q по символу $a \in V$ или по пустой цепочке (тогда $a = \lambda$) можно перейти в состояние r“ (возможно, что $q = r$).

При этом по определению принимается, что переход по пустой цепочке и переход по входному символу исключают друг друга. То есть для любой пары (q, r) состояний конечного автомата имеет место следующее: если существует команда (7.4) при $a = \lambda$, то нет ни одной команды (для такой же пары состояний) при $a \in V$ и наоборот.

Конфигурация конечного автомата определяется как *упорядоченная пара* (q, ay), где q — состояние блока управления, a — символ в обозреваемой ячейке, y — цепочка во входном алфавите, расположенная в ячейках справа от обозреваемой (цепочку ay принято называть ***непрочитанной частью входной цепочки***). При этом если обозреваемая ячейка пуста, т.е. не содержит какого-либо символа входного алфавита, то непрочитанная часть входной цепочки считается пустой цепочкой.

Чтобы дать математическое определение конечного автомата, нужно заметить, что он, в свете только что изложенного неформального описания, допускает естественную интерпретацию в терминах *размеченных ориентированных графов*. Будем рассматривать состояния блока управления конечного автомата как *вершины* ориентированного графа, множество *дуг* которого определяется системой команд следующим образом: дуга ведет из состояния q в состояние r (для данных состояний q и r) тогда и только тогда, когда в системе команд автомата есть команда (7.4), т.е. возможен переход из состояния q в состояние r. Каждая дуга имеет метку, которая является либо множеством букв входного алфавита, либо пустой цепочкой.

Метка дуги (q, r) есть пустая цепочка λ, если из q в r можно перейти по пустой цепочке; в противном случае метка дуги (q, r) есть множество всех входных символов, по которым возможет переход из состояния q в состояние r.

Пример 7.7. Зададим конечный автомат с входным алфавитом $\{a, b, c\}$ и множеством состояний $\{q_0, q_1, q_2, q_3\}$ такой

системой команд:

$$q_0\lambda \to q_1,$$
$$q_0\lambda \to q_3,$$
$$q_1a \to q_2,$$
$$q_1b \to q_2,$$
$$q_1a \to q_3,$$
$$q_2b \to q_1,$$
$$q_3b \to q_2,$$
$$q_3c \to q_2,$$
$$q_3c \to q_3.$$

По этой системе команд построим размеченный ориентированный граф, изображенный на рис. 7.3. Среди состояний автомата выделены начальное состояние q_0 и два заключительных состояния q_2 и q_3. На рис. 7.4 показана последовательность конфигураций, которую проходит конечный автомат, читая цепочку *abac*. Эту цепочку автомат допускает, так как, читая ее, он переходит из начального состояния в одно из заключительных. В формальной записи последовательность конфигураций на рис. 7.4 выглядит так: $(q_0, abac)$, $(q_1, abac)$, (q_2, bac), (q_1, ac), (q_3, c), (q_3, λ).

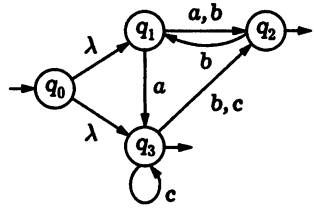


Рис. 7.3

Этой последовательности отвечает *путь* в ориентированном графе, ведущий из вершины q_0 в вершину q_3:

$$q_0 \to_\lambda q_1 \to_a q_2 \to_b q_1 \to_a q_3 \to_c q_3$$

(под каждой стрелкой подписана буква, принадлежащая метке соответствующей дуги и являющаяся очередной буквой читаемой входной цепочки). Заметим, что, например, находясь в состоянии q_1 и обозревая второй от конца цепочки символ, т.е. символ a, автомат мог, согласно системе команд, сделать переход в состояние q_2, а не в состояние q_3, но тогда он бы „завис“

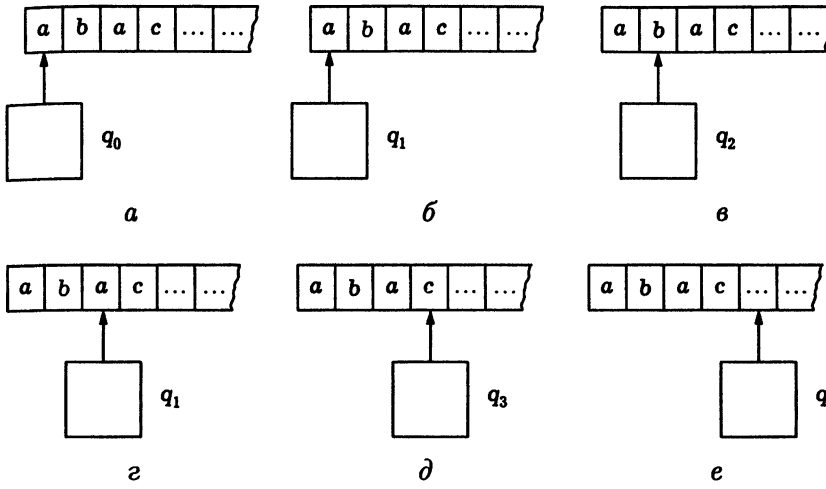


Рис. 7.4

в состоянии q_2 и не смог бы прочитать последний символ записанной на ленте цепочки, т.е. символ c, так как среди команд нет такой, которая разрешала бы переход из состояния q_2 куда-либо по символу c.

Эта ситуация демонстрирует как раз недетерминированность конечного автомата как распознающего устройства: система команд („правила игры“) позволяет автомату допустить цепочку $abac$ („игроку“ найти последовательность ходов, ведущую к „выигрышу“), но из этого вовсе не следует, что последовательность конфигураций, которую проходит автомат, читая записанную на ленте цепочку, является единственной. Автомат может „ошибиться“, сделав „неправильный“ ход. Но и последовательность „правильных“ ходов может быть не единственной. Например, читая последний символ цепочки, т.е. символ c, автомат мог „выбрать“ переход в состояние q_2, которое также является заключительным. Рассматриваемый автомат допускает не всякую цепочку в алфавите $\{a, b, c\}$. Видно, что ни одна цепочка, которая начинается с *префикса* ca, не будет допущена автоматом. #

Обозначение пустой цепочки λ, фигурирующей в виде метки дуги ориентированного графа, который представляет конечный автомат, можно интерпретировать как *регулярное выражение*, т.е. как регулярный язык, состоящий из одной пустой цепочки (см. замечание 7.4). Поскольку метка дуги, являющаяся множеством букв $\{b_1, \ldots, b_m\} \subseteq V$, может быть также записана в виде регулярного выражения, а именно как сумма $b_1 + \ldots + b_m$, то метку каждой дуги можно считать регулярным выражением определенного вида или, так как мы отождествляем регулярные языки и регулярные выражения, регулярным языком. Это позволяет нам формально определить конечный автомат как *ориентированный граф, размеченный над полукольцом* $\mathcal{R}(V)$ регулярных языков. Такое математическое определение конечного автомата весьма удобно: оно дает нам возможность применить при изучении конечных автоматов уже изученные нами алгебраические методы анализа размеченных ориентированных графов.

Итак, математическое определение конечного автомата формулируется следующим образом. ***Конечный автомат*** — это ориентированный граф, размеченный над полукольцом $\mathcal{R}(V)$ регулярных языков в алфавите V с выделенной вершиной q_0, которая называется ***начальной***, и с выделенным подмножеством вершин F, каждый элемент которого называется ***заключительной вершиной***.

На *функцию разметки* при этом накладываются следующие ограничения: *метка* каждой *дуги* есть либо язык $\{\lambda\}$, либо непустое подмножество *алфавита* V.

Вершины графа называют обычно в этом случае ***состояниями конечного автомата***, начальную вершину — ***начальным состоянием***, а заключительную вершину — ***заключительным состоянием конечного автомата***.

Замечание 7.5. Если для какой-то дуги $e \in E$ ее метка $\varphi(e)$ есть язык $\{\lambda\}$, то, поскольку этот язык не является подмножеством алфавита V, в этом случае $\varphi(e) \not\subseteq V$, и, наоборот,

если $\varphi(e) \subseteq V$, то исключается, что $\varphi(e) = \{\lambda\}$. Таким образом, два рассмотренных случая для значений функции разметки исключают друг друга, на что и было указано в рассмотренном выше неформальном описании конечного автомата. #

На рис. 7.5 изображен конечный автомат, для которого алфавит $V = \{0, 1\}$. Начальное состояние показано входной стрелкой, заключительное — выходной. Метки дуг обычно пишут без фигурных скобок. Разрешена запись меток дуг и в виде *регулярных выражений* (см. рис. 7.5).

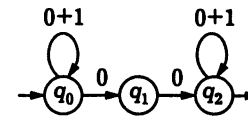


Рис. 7.5

Конечный автомат, таким образом, может быть задан как пятерка:

$$M = (Q, E, \varphi, q_0, F),$$

где Q — множество состояний автомата; E — множество дуг; φ — функция разметки (весовая функция), причем для каждой дуги $e \in E$ ее метка $\varphi(e) = \{\lambda\}$ или $\varphi(e) \subseteq V$, при этом подмножество $\varphi(e)$ не пусто; $q_0 \in Q$ — начальное состояние; $F \subseteq Q$ — подмножество заключительных состояний.

Алфавит V называется ***входным алфавитом автомата*** M, а его буквы — ***входными символами*** (или ***входными буквами***) данного ***автомата***.

Замечание 7.6. Конечный автомат определен как ориентированный граф, размеченный над полукольцом регулярных языков, но метка дуги задается не как произвольный регулярный язык, а как язык, состоящий из одной пустой цепочки, либо язык, являющийся подмножеством букв входного алфавита. Это ни в коей мере не противоречит основному определению размеченного ориентированного графа, ибо совершенно не обязательно, чтобы *область значения* функции разметки совпадала с *носителем* полукольца. Чисто формально, конечно, можно обобщить определение конечного автомата и допустить в качестве меток дуг произвольные регулярные языки, но, как

можно показать, это обобщение не является принципиальным, и такое определение конечного автомата сводится к данному выше определению (см. задачи в конце этой главы). Немаловажно и то, что приведенное формальное определение конечного автомата и задание меток дуг как регулярных языков специального вида согласуются с интуитивным представлением об автомате как об устройстве, которое „побуквенно" читает входные цепочки, переходя из одного состояния в другое. Требование, чтобы такое устройство за один такт анализировало („обозревало") любое, сколь угодно сложное регулярное выражение, не отвечает нашей „вычислительной" интуиции, в соответствии с которой за один такт может быть произведена только простая операция, каковой и является „реакция" автомата на обозреваемый одиночный символ или на пустую цепочку. #

Если $e = (q, r)$ — *дуга* автомата M и ее метка $\varphi(e)$ есть регулярное выражение λ, то в этом случае будем говорить, что в автомате M возможен ***переход из состояния*** q ***в состояние*** r по пустой цепочке, и писать $q \to_\lambda r$. Дугу с меткой λ будем называть ***λ-переходом*** (или ***пустой дугой***).

Если же метка дуги e есть множество, содержащее входной символ a, то будем говорить, что в автомате M возможен переход из состояния q в состояние r по символу a, и писать $q \to_a r$.

Для конечного автомата удобно ввести в рассмотрение ***функцию переходов***, определив ее как отображение

$$\delta\colon Q \times (V \cup \{\lambda\}) \to 2^Q,$$

такое, что

$$\delta(q,a) = \{r\colon q \to_a r\},$$

т.е. значение функции переходов на упорядоченной паре (состояние, входной символ или пустая цепочка) есть множество всех состояний, в которые из данного состояния возможен переход по данному входному символу или пустой цепочке. В частности, это может быть пустое множество.

На рис. 7.5 $\delta(q_0,0)=\{q_0,q_1\}$, $\delta(q_0,1)=\{q_0\}$, $\delta(q_1,0)=\{q_2\}$, $\delta(q_1,1)=\varnothing$, $\delta(q_2,0)=\{q_2\}$, $\delta(q_2,1)=\{q_2\}$.

Понятие функции переходов конечного автомата позволяет дать и математическую интерпретацию системы команд. С этой точки зрения система команд есть просто способ представления *конечной функции*, а именно функции переходов.

Система команд есть конечное множество ***команд*** вида

$$qa \to r,$$

где q и r — состояния автомата; a — входной символ или пустая цепочка, причем указанная команда тогда и только тогда содержится в системе команд, когда $q \to_a r$.

Содержательная интерпретация команды была приведена выше. Стрелка ($\to$), как и в записи правила грамматики, есть „метасимвол“. Он не содержится ни в одном из алфавитов, фигурирующих в определении конечного автомата.

Система команд автомата, изображенного на рис. 7.5, приведена ниже:

$$q_00 \to q_0,$$
$$q_00 \to q_1,$$
$$q_01 \to q_0,$$
$$q_10 \to q_2,$$
$$q_20 \to q_2,$$
$$q_21 \to q_2.$$

Используя функцию переходов, конечный автомат можно задавать как упорядоченную пятерку:

$$M=(V,\ Q,\ q_0,\ F,\ \delta),$$

где V — входной алфавит; Q — множество состояний; q_0 — начальное состояние; F — множество заключительных состояний; δ — функция переходов, заданная в виде системы команд.

Как правило, в дальнейшем мы именно так и будем определять конечный автомат.

Конечный автомат называют ***полностью определенным***, если из каждого его состояния по каждому входному символу возможен переход в некоторое состояние, т.е.

$$(\forall q \in Q)(\forall a \in V)(\delta(q,a) \neq \varnothing).$$

Заметим, что в полностью определенном конечном автомате, вообще говоря, могут существовать и переходы по пустой цепочке.

Конечный автомат называется ***детерминированным***, если в нем нет дуг с меткой λ и из любого состояния по любому входному символу возможен переход в точности в одно состояние, т.е.

$$(\forall q \in Q)(\forall a \in V)(|\delta(q,a)| = 1).$$

Конечный автомат называется ***квазидетерминированным***, если в нем нет дуг с меткой λ и из любого состояния по любому символу возможен переход не более чем в одно состояние, т.е.

$$(\forall q \in Q)(\forall a \in V)(|\delta(q,a)| \leqslant 1).$$

Замечание 7.7. Для детерминированного конечного автомата значением функции переходов для любой пары (состояние, входной символ) является одноэлементное подмножество множества состояний. Поэтому естественно понимать функцию переходов детерминированного конечного автомата не как отображение множества $Q \times V$ в множество 2^Q, а как отображение множества $Q \times V$ в само множество состояний Q. Впредь так и будем рассматривать функцию переходов детерминированного конечного автомата, не оговаривая этого особо. #

Согласно общему определению *метки пути в размеченном ориентированном графе*, *метка пути* в конечном автомате есть *соединение** меток входящих в этот путь дуг (в порядке их прохождения). Таким образом, метка любого *пути конечной длины* в конечном автомате есть регулярный язык. Отметим, что

* *Умножением полукольца* $\mathcal{R}(V)$ *является соединение языков.*

мы, изучая размеченные ориентированные графы, предполагаем, что рассматриваются только пути конечной длины. Так, для автомата, изображенного на рис. 7.5, метка пути q_0, q_0, q_1, q_2 равна соединению языков $\{0,1\}\cdot\{0\}\cdot\{0\} = \{000, 100\}$, что можно записать в виде регулярного выражения $(0+1)00$, а метка пути q_0, q_1, q_2, q_2, q_2 может быть задана таким регулярным выражением:

$$00(0+1)(0+1) = 00(00+01+10+11) = \\ = 0000+0001+0010+0011,$$

что означает, что эта метка есть множество цепочек

$$\{0000, 0001, 0010, 0011\}.$$

Метку пути W конечной длины обозначим через $\varphi^*(W)$. Здесь φ^* — это функция, отображающая множество всех путей конечной длины в размеченном ориентированном графе, в полукольцо $\mathcal{R}(V)$. Для пути W длины 1 значение $\varphi^*(W) = \varphi(W)$, т.е. функция разметки ориентированного графа есть *ограничение функции* φ^* на множество всех путей длины 1.

Если цепочка x принадлежит метке некоторого пути W — пути, ведущего из вершины q в вершину r конечного автомата M, то говорят, что цепочка x читается на пути W в M или что путь W несет цепочку x. Пишем $q \Rightarrow_x^* r$, если x читается на некотором пути из q в r. В том случае, когда явно надо указать *длину* n *пути*, на котором читается цепочка x, записываем $q \Rightarrow_x^n r$. Если нужно подчеркнуть, что цепочка x читается на некотором пути ненулевой длины из q в r, то используем запись $q \Rightarrow_x^+ r$. В терминах неформального описания конечного автомата любому пути, на котором читается входная цепочка x, отвечает последовательность конфигураций, которую проходит автомат, читая посимвольно входную цепочку, записанную на ленте, и продвигая головку на один символ вправо за один такт.

Стоимость прохождения из состояния q в состояние r есть (согласно общему определению этого понятия в размеченных ориентированных графах) *объединение* меток всех путей*, ведущих из q в r, т.е. множество всех таких x, что $q \Rightarrow_x^* r$. Это значит, что элемент c_{qr} *матрицы стоимостей* есть язык

$$c_{qr} = \{x\colon q \Rightarrow_x^* r\}.$$

Говоря об элементе c_{qr} матрицы стоимостей, мы отождествляем обозначение состояния автомата с его номером в некоторой нумерации состояний как вершин ориентированного графа.

В частности, ***язык*** $L(M)$ ***конечного автомата*** M есть множество всех цепочек во входном алфавите, читаемых в M на некотором пути из начального состояния в одно из заключительных состояний. Другими словами,

$$L(M) = \{x\colon q_0 \Rightarrow_x^* q_f,\ q_f \in F\} = \bigcup_{q_f \in F} c_{q_0 q_f}, \qquad (7.5)$$

где F — множество заключительных состояний.

Таким образом, язык конечного автомата есть объединение тех элементов матрицы стоимостей автомата, которые находятся в строке, соответствующей начальному состоянию q_0, и в столбцах, соответствующих всем заключительным состояниям $q_f \in F$.

Замечание 7.8. Необходимо обратить внимание на одну очень важную вещь. В конечном автомате метка произвольного пути конечной длины есть регулярный язык, поскольку он вычисляется как соединение конечного семейства регулярных языков. Но стоимость прохождения между заданной парой вершин априори не является регулярным языком, так как множество путей, ведущих из одной вершины в другую, может

*Здесь объединение понимается как *бесконечная сумма* замкнутого полукольца $\mathcal{L}(V)$ всех языков в алфавите V, т.е. как *точная верхняя грань* последовательности языков относительно теоретико-множественного включения.

быть бесконечным (каждый путь имеет конечную длину, но множество всех таких путей может оказаться бесконечным, содержать пути сколь угодно большой длины — простейший пример дает *петля*, по которой можно пройти сколько угодно раз). Поэтому объединение при определении стоимости прохождения между парой состояний конечного автомата мы можем сейчас рассматривать только как операцию замкнутого полукольца всех языков в данном алфавите, а именно операцию вычисления точной верхней грани („бесконечная сумма" в замкнутом полукольце). Но коль скоро элементы матрицы стоимостей уже вычислены, их объединение (в формуле (7.5)), дающее язык конечного автомата, разумеется, конечно. Позже будет доказано, что на самом деле все стоимости в конечном автомате также регулярны. #

О языке $L(M)$ говорят, что он допускается (или воспринимается) конечным автоматом M. О любой цепочке, принадлежашей языку $L(M)$, говорят, что она допускается (или воспринимается) конечным автоматом M. Такую цепочку называют также ***допустимой цепочкой*** данного ***конечного автомата.***

Определение 7.10. Два конечных автомата M_1 и M_2 называют ***эквивалентными***, если их языки совпадают:

$$L(M_1) = L(M_2).$$

Установим теперь связь между понятиями конечного автомата и *регулярной грамматики.*

Теорема 7.5. Язык допускается конечным автоматом тогда и только тогда, когда он порождается регулярной грамматикой.

◀ Чтобы доказать теорему, нужно:

1) указать способ построения регулярной грамматики G по заданому конечному автомату M, такой, чтобы ***язык, по-***

рождаемый грамматикой G, совпадал с языком, допускаемым автоматом M: $L(G) = L(M)$;

2) указать способ построения конечного автомата M по заданной регулярной грамматике G, такой, чтобы язык, допускаемый автоматом M, совпадал с языком, порождаемым грамматикой G: $L(M) = L(G)$.

Замечание 7.9. Регулярную грамматику G и конечный автомат M, такие, что $L(G) = L(M)$, иногда называют эквивалентными. #

Рассмотрим эти построения по очереди.

1. Построение регулярной грамматики по конечному автомату. Пусть дан конечный автомат

$$M = (V, Q, q_0, , F, \delta).$$

Определим регулярную грамматику $G = (V, N, S, P)$ следующим образом: *терминальный алфавит* V совпадает с входным алфавитом автомата M; *нетерминальный алфавит* N находится во *взаимно однозначном соответствии* с множеством состояний Q автомата M, причем состоянию $q \in Q$ сопоставлен *нетерминал* $S_q \in N$ и *аксиома* сопоставлена начальному состоянию q_0, т.е. $S = S_{q_0}$; множество *правил вывода* P строится по системе команд δ так: правило вывода $S_q \to aS_r$, где $a \in V \cup \{\lambda\}$, принадлежит P тогда и только тогда, когда в δ есть команда $qa \to r$; кроме того, если (и только если) состояние r заключительное ($r \in F$), то в P добавляется правило вывода $S_q \to a$. Если же $q_0 \in F$ (и только в этом случае), то в P помещаем правило вывода $S_{q_0} \to \lambda$.

Для конечного автомата на рис. 7.5 получим следующую регулярную грамматику:

$$S_{q_0} \to 0S_{q_0} \,|\, 1S_{q_0} \,|\, 0S_{q_1},$$
$$S_{q_1} \to 0S_{q_2} \,|\, 0,$$
$$S_{q_2} \to 0S_{q_2} \,|\, 1S_{q_2} \,|\, 0 \,|\, 1.$$

Заметим, что, читая цепочку в автомате, в грамматике мы ее выводим (порождаем). Например, $q_0 \to_0 q_1 \to_0 q_2 \to_1 q_2$ и $S_{q_0} \vdash 0S_{q_1} \vdash 00S_{q_2} \vdash 001$.

Докажем, что описанное построение корректно, т.е. $L(G) = L(M)$. Для этого сначала, используя индукцию по длине n пути в конечном автомате M, докажем, что из факта *достижимости* $q \Rightarrow_x^n r$ в автомате M (для любого $n \geqslant 0$) следует *выводимость* $S_q \vdash_G^* xS_r$ в грамматике G для любых $q, r \in Q$ и $x \in V^*$.

Пусть длина пути $n = 0$, т.е. $q \Rightarrow_x^0 r$ и $r = q$. Так как метка пути нулевой длины в ориентированном графе, размеченном над полукольцом $\mathcal{R}(V)$, равна единице полукольца — регулярному выражению λ, то $q \Rightarrow_\lambda^0 q$ при $x = \lambda$. Но $S_q \vdash_G^0 S_q$ выполняется тривиально.

Пусть для любого $k \leqslant m-1$ из достижимости $q \Rightarrow_x^k r$ следует выводимость $S_q \vdash^* xS_r$, и пусть в автомате M существует путь W длины m, ведущий из вершины q в вершину r, на котором читается цепочка x, т.е. $q \Rightarrow_x^m r$. Так как длина пути W не меньше 1, то в нем есть по крайней мере одна дуга и существуют такая вершина q' и такое $a \in V \cup \{\lambda\}$, что $q \to_a q' \Rightarrow_y^{m-1} r$, где $ay = x$ (мы выделили первую дугу пути W). Тогда, согласно построению множества P правил вывода грамматики G, в P содержится правило $S_q \to aS_{q'}$, а, по предположению индукции, из того, что в M $q' \Rightarrow_y^{m-1} r$, следует, что $S_{q'} \vdash_G^* yS_r$. В итоге получаем $S_q \vdash_G aS_{q'} \vdash_G^* ayS_r = xS_r$, т.е. $S_q \vdash_G^* xS_r$, что и требовалось.

Теперь если цепочка $x \in L(M)$, то в M существует путь, ведущий из начального состояния q_0 в одно из заключительных состояний q_f, на котором читается цепочка x, т.е. $q_0 \Rightarrow_x^* q_f$ для некоторого $q_f \in F$. Если $q_f = q_0$ и тогда $x = \lambda \in L(M)$, то в множестве P правил вывода есть правило $S_{q_0} \to \lambda$ и $\lambda \in L(G)$. Если же q_f отлично от q_0, то, какова бы ни была цепочка x, путь из q_0 в q_f, на котором она читается, имеет длину, не меньшую единицы. Выделяя в этом пути последнюю дугу, получим $q_0 \Rightarrow_y^* q' \to_a q_f$, где $ya = x$. Но тогда, во-первых,

как мы только что доказали, в грамматике G имеет место выводимость $S_{q_0} \vdash_G^* yS_{q'}$, а, во-вторых, из того, что $q' \to_a q_f$, согласно построению множества правил вывода грамматики G, следует, что в ней есть правило $S_{q'} \to a$, и окончательно получим

$$S_{q_0} \vdash_G^* yS_{q'} \vdash ya = x,$$

т.е. $x \in L(G)$.

Таким образом, мы полностью доказали, что для каждой цепочки $x \in V^*$ из того, что $x \in L(M)$, следует $x \in L(G)$, т.е. имеет место включение $L(M) \subseteq L(G)$.

Чтобы доказать обратное включение, используя индукцию по длине n вывода в грамматике G, покажем сначала, что из факта выводимости $S_q \vdash_G^n xS_r$ (для любого $n \geqslant 0$) следует достижимость $q \Rightarrow_x^* r$ в автомате M для любых $q, r \in Q$ и $x \in V^*$.

При $n = 0$ имеем $S_q \vdash_G^0 S_q$, $x = \lambda$ и тривиально выполняется $q \Rightarrow_\lambda^0 q$.

Пусть для любого $k \leqslant m-1$ из выводимости $S_q \vdash_G^k xS_r$ следует достижимость $q \Rightarrow_x^* r$ (в автомате M), и пусть в грамматике G существует *вывод D длины m* цепочки xS_r из цепочки S_q, т.е. $S_q \vdash_G^m xS_r$. Так как длина вывода D не меньше 1, то найдется такой нетерминал $S_{q'}$ и такое $a \in V \cup \{\lambda\}$, что $S_q \vdash_G aS_{q'} \vdash_G^{m-1} ayS_r$, где $ay = x$ (мы выделили первый *шаг вывода D*). *Непосредственная выводимость* $S_q \vdash_G aS_{q'}$ означает, что в множестве правил вывода грамматики G содержится правило $S_q \to aS_{q'}$ и, следовательно, в системе команд δ конечного автомата M есть команда $qa \to q'$ и тогда $q \to_a q'$. Согласно предположению индукции, из того, что $S_{q'} \vdash_G^{m-1} yS_r$, следует, что $q' \Rightarrow_y^* r$ в M. В итоге получаем $q \to_a q' \Rightarrow_y^* r$, т.е., так как $ay = x$, $q \Rightarrow_x^* r$, что и требовалось.

Если теперь $x \in L(G)$, то в грамматике G существует вывод цепочки x из аксиомы S_{q_0}, т.е. $S_{q_0} \vdash_G^* x$. На последнем шаге этого вывода применяется некоторое правило, правая часть которого не содержит *вхождений* нетерминалов, т.е. правило вида $S_{q'} \to a$. Поскольку любая цепочка в выводе в регулярной

грамматике может содержать нетерминал только как последний символ, то $S_{q_0} \vdash_G^* yS_{q'} \vdash_G ya = x$. Отсюда следует, что $q_0 \Rightarrow_y^* q'$. Кроме того, правило вывода $S_{q'} \to a$ может принадлежать множеству P тогда и только тогда (по построению P), когда в системе команд автомата M есть команда $q'a \to q_f$, $q_f \in F$. Следовательно, $q_0 \Rightarrow_y^* q' \to_a q_f$, что и означает $q_0 \Rightarrow_x^* q_f$, т.е. $x \in L(M)$.

Итак, доказано включение $L(G) \subseteq L(M)$, и $L(G) = L(M)$, т.е. регулярная грамматика G, построенная, как описано выше, по конечному автомату M, эквивалентна ему.

2. Построение конечного автомата по регулярной грамматике. По заданной регулярной грамматике

$$G = (V,\ N,\ S,\ P)$$

построим конечный автомат $M = (V,\ Q,\ q_0,\ F,\ \delta)$ так, что входной алфавит автомата M совпадает с терминальным алфавитом грамматики G, множество состояний Q совпадает с нетерминальным алфавитом грамматики G, пополненным новым состоянием f, не принадлежащим объединенному алфавиту грамматики G, т.е. $Q = N \cup \{f\}$; начальное состояние q_0 совпадает с аксиомой S, множество заключительных состояний $F = \{f\}$. Система команд δ определяется следующим образом: команда $Aa \to B$ принадлежит δ тогда и только тогда, когда в множестве P правил вывода есть правило $A \to aB$, а команда $Aa \to f$ принадлежит δ тогда и только тогда, когда правило вывода $A \to a$ принадлежит множеству P.

Например, по регулярной грамматике G_2 из примера 7.5 строится конечный автомат с входным алфавитом $\{a, b, 0, 1\}$, множеством состояний $\{I, D, f\}$, начальным состоянием I, единственным заключительным состоянием f и такой системой команд (рис. 7.6):

$$\begin{array}{llll} Ia \to D, & Ib \to D, & Da \to D, & Db \to D, \\ D0 \to D, & D1 \to D, & Da \to f, & Db \to f, \\ D0 \to f, & D1 \to f, & Ia \to f, & Ib \to f. \end{array}$$

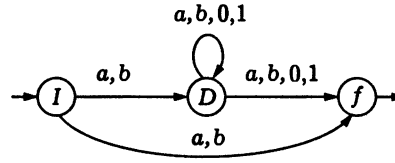


Рис. 7.6

Обоснование корректности этого построения может быть проведено так же, как и построения регулярной грамматики по конечному автомату („двойной" индукцией по длине вывода в грамматике и по длине пути в автомате), и мы его опускаем. ▶

Таким образом, конечные автоматы допускают в точности те же языки, которые порождают регулярные грамматики.

Выше (см. **7.4**) без доказательства сформулирована теорема 7.4, согласно которой регулярные языки (в алфавите V) и языки полукольца $\mathcal{R}(V)$ — одно и то же. Это позволило нам (так сказать, авансом) называть элементы полукольца $\mathcal{R}(V)$ регулярными языками. Теперь пришла пора доказать это подробно. Для этого, опять разделив понятия языка из полукольца $\mathcal{R}(V)$ и регулярного языка (т.е. языка, порождаемого регулярной грамматикой), докажем, что язык допускается конечным автоматом с входным алфавитом V тогда и только тогда, когда он есть элемент полукольца $\mathcal{R}(V)$. Тогда, с учетом доказанной теоремы 7.5, будет доказана и теорема 7.4.

Теорема 7.6 (теорема Клини). Пусть $V = \{a_1, \ldots, a_n\}$ — произвольный алфавит. Язык $L \subseteq V^*$ является элементом полукольца $\mathcal{R}(V)$ тогда и только тогда, когда он допускается некоторым конечным автоматом.

◀ Докажем, что всякий язык из $\mathcal{R}(V)$ допускается некоторым конечным автоматом.

Для доказательства этого утверждения мы воспользуемся методом индукции по построению языка из $\mathcal{R}(V)$ как элемента *замыкания* множества $\{\varnothing, \{\lambda\}, \{a_1\}, \ldots, \{a_n\}\}$. Этот метод

состоит в следующем: сначала утверждение доказывается для языков исходного множества (замыкание которого строится), а затем в предположении, что утверждение доказано для языков L и K из $\mathcal{R}(V)$, оно доказывается для $L \cup K$, LK и L^*.

Конечные автоматы для языков $\varnothing$, $\{\lambda\}$, $\{a\}$, где $a \in V$, приведены на рис. 7.7.

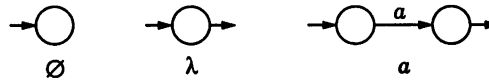


Рис. 7.7

Пусть конечные автоматы

$$M_1 = (V,\ Q_1,\ q_{01},\ F_1,\ \delta_1) \quad \text{и} \quad M_2 = (V,\ Q_2,\ q_{02},\ F_2,\ \delta_2)$$

для языков L и K полукольца $\mathcal{R}(V)$ соответственно уже построены. Обратим внимание на то, что входные алфавиты этих автоматов совпадают (мы работаем на множестве языков в произвольном, но фиксированном алфавите V) и автоматы не имеют ни общих вершин, ни общих дуг.

На рис. 7.8 изображен метод построения конечных автоматов для языков $L \cup K$, LK и L^*. На рисунке видно, что автомат для объединения (см. рис. 7.8, *а*) получается при сохранении всех дуг и вершин автоматов объединяемых языков путем добавления новой начальной вершины s_0, проведения из нее пустых дуг в каждую из начальных вершин объединяемых автоматов (q_{01} и q_{02}) и объявления множеством заключительных вершин объединения множеств заключительных вершин (F_1 и F_2) объединяемых автоматов. Получается „параллельное соединение“ автоматов для языков L и K. Любая цепочка x, читаемая на некотором пути из вершины s_0 в какую-то из вершин множества $F_1 \cup F_2$, может быть представлена так: $x = \lambda x_1 = x_1$ (переход по пустой цепочке из s_0 q_{01} и дальнейшее чтение произвольной цепочки x_1, допускаемой автоматом M_1) или $x = \lambda x_2 = x_2$ (переход

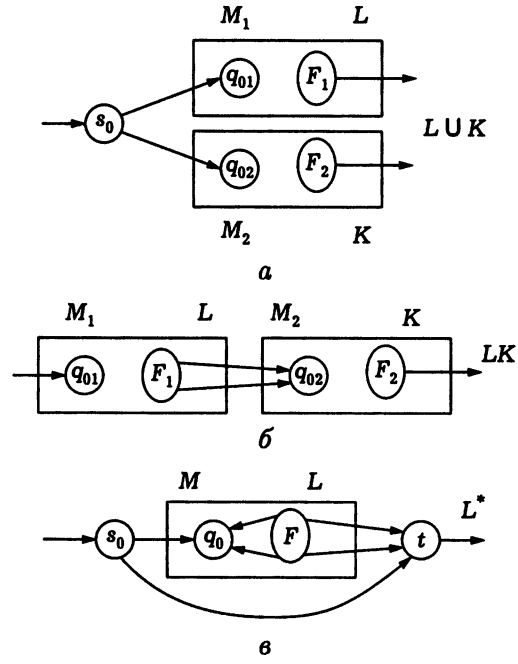


Рис. 7.8

по пустой цепочке из s_0 q_{02} и дальнейшее чтение произвольной цепочки x_2, допускаемой автоматом* M_2).

Аналогично при построении конечного автомата для соединения (см. рис. 7.8, *б*) достаточно объявить новой начальной вершиной начальную вершину первого автомата (q_{01}), а множеством заключительных вершин — таковое для второго автомата (F_2), после чего из каждой заключительной вершины первого автомата провести пустую дугу в начальную вершину второго автомата. Получится „последовательное соединение“ автоматов.

*Новый конечный автомат как некое распознающее устройство может из своей начальной вершины по пустой цепочке, т.е. не читая ни одного символа входной ленты, перейти в начальную вершину первого автомата и потом „работать за него“ или „сделать выбор в пользу“ второго автомата, перейдя по пустой цепочке из вершины s_0 в вершину q_{02}.

Конечный автомат для итерации (см. рис. 7.8, *в*) строится следующим образом. Нужно: а) ввести новые начальную (s_0) и заключительную (t) вершины, проведя пустую дугу из первой в последнюю; б) провести пустые дуги из новой начальной вершины в прежнюю начальную вершину автомата итерируемого языка (q_0), а также из каждой заключительной вершины множества F автомата языка L в новую заключительную вершину и прежнюю начальную вершину. Итак, каждый язык полукольца $\mathcal{R}(V)$ допускается некоторым конечным автоматом.

Докажем теперь, что язык произвольного конечного автомата есть элемент полукольца $\mathcal{R}(V)$. Язык конечного автомата, как следует из формулы (7.5), — это конечное объединение языков, являющихся определенными элементами матрицы стоимостей автомата. Матрица стоимостей есть *итерация матрицы* меток дуг, задающей автомат. Метка каждой дуги — регулярное выражение, обозначающее язык из полукольца $\mathcal{R}(V)$, и матрица стоимостей, следовательно, является итерацией матрицы, все элементы которой могут быть определены регулярными выражениями, т.е. принадлежат полукольцу $\mathcal{R}(V)$. Полукольцо $\mathcal{R}(V)$ есть полукольцо с итерацией. Поэтому в силу теоремы 3.9 и матрица стоимостей конечного автомата будет состоять из языков полукольца $\mathcal{R}(V)$. Отсюда следует, что язык конечного автомата есть элемент этого полукольца. ▶

Замечание 7.10. Обратим внимание на то, что в общем случае при построении итерации нельзя обойтись без добавления новых начальной и заключительной вершин. Рассмотрим в этой связи построение автомата для итерации языка, допускаемого автоматом на рис. 7.9, *а*.

Если мы построим автомат для итерации так, как показано на рис. 7.9, *б*, не вводя новую начальную вершину, то этот автомат будет допускать, в частности, все цепочки языка $(01)^*$. Однако эти цепочки не содержатся в итерации исходного языка. В частности, $01 \notin ((01)^*1)^*$. Действительно, любая цепочка языка исходного автомата задается регулярным выражением

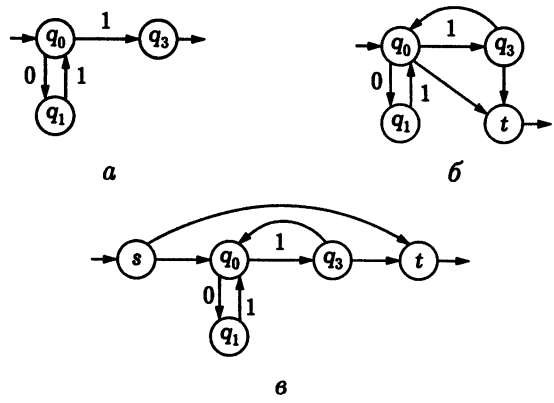


Рис. 7.9

(01)*1 и есть либо 1, либо цепочка, оканчивающаяся подцепочкой 11. Следовательно, любая цепочка из итерации исходного языка есть либо цепочка 1^n, где $n \geqslant 0$, либо цепочка, оканчивающаяся подцепочкой 11.

Можно построить аналогичный пример и для того, чтобы убедиться в необходимости добавления новой заключительной вершины.

На рис. 7.9, *в* приведен пример правильно построенной итерации.

Если при построении конечного автомата для итерации мы не будем проводить пустую дугу $s \to t$, то получим автомат для позитивной итерации исходного языка. #

Из теорем 7.5 и 7.6 получим следующий результат.

Следствие 7.2. Следующие утверждения о языке L в алфавите V эквивалентны:

1) L есть элемент полукольца $\mathcal{R}(V)$;

2) L порождается некоторой регулярной грамматикой;

3) L допускается некоторым конечным автоматом.

Задачу построения конечного автомата по данному регулярному выражению называют ***задачей синтеза конечного***

автомата. Вычисление же языка конечного автомата есть ***задача анализа конечного автомата.***

Рассмотрим более подробно решение задачи анализа конечного автомата.

Как же практически вычислить язык конечного автомата? Для этого, вообще говоря, достаточно вычислить, как следует из формулы (7.5), матрицу стоимостей автомата (как размеченного ориентированного графа) любым из способов, которыми мы анализировали размеченные ориентированные графы (см. 5). Если мы выберем для вычисления итерации способ, связанный с решением систем линейных уравнений, нам понадобится решить $n = |Q|$ систем вида

$$\xi_j = A\xi_j + \varepsilon_j,$$

где A — квадратная матрица n-го порядка*, элемент a_{ij} которой является регулярным выражением, служащим меткой дуги из вершины (состояния) q_i в вершину (состояние) q_j (при некоторой выбранной нумерации состояний!), если такая дуга существует, и равен регулярному выражению $\varnothing$, если нет дуги из q_i в q_j; ε_j — j-й столбец единичной матрицы, т.е. столбец, у которого все компоненты, кроме j-й, равны $\varnothing$ (нулю полукольца $\mathcal{R}(V)$), а j-я компонента равна λ (единице полукольца $\mathcal{R}(V)$).

Решив указанные n систем, найдем матрицу стоимостей $C = A^*$ заданного конечного автомата. Но нам, как правило, нужна не вся матрица стоимостей, а только элементы вида c_{st}, где s — номер начального, а t — один из номеров заключительного состояния.

Поэтому, вместо того чтобы решать несколько систем линейных уравнений, достаточно решить одну:

$$\xi = A\xi + \beta, \tag{7.6}$$

*Это не что иное, как матрица меток дуг, определяющая размеченный ориентированный граф (см. 5).

где β — столбец, все компоненты которого равны $\varnothing$ (нулю полукольца $\mathcal{R}(V)$), кроме компонент с номерами $t_1, \ldots, t_m$, которые являются номерами заключительных состояний. Эти компоненты равны λ (единице полукольца $\mathcal{R}(V)$). Другими словами, ко всем уравнениям системы, соответствующим заключительным состояниям, добавляется слагаемое λ.

Действительно, решение системы (7.6) будет иметь вид

$$\xi = A^*\beta = A^*(\varnothing, \ldots, \varnothing, \lambda, \varnothing, \ldots, \varnothing, \lambda, \varnothing, \ldots, \varnothing)^{\mathrm{T}} \qquad (7.7)$$

(элементы λ находятся в строках с номерами $t_1, \ldots, t_m$).

Умножая в (7.7) матрицу A^*, равную матрице C стоимостей, на столбец β, получим столбец, s-я компонента которого x_s будет равна произведению s-й строки матрицы C $(c_{s1}, \ldots, c_{st_1}, \ldots, c_{st_m}, \ldots, c_{sn})$ на столбец β в формуле (7.7), т.е.

$$x_s = c_{st_1} + \ldots + c_{st_m},$$

но это и есть регулярное выражение, обозначающее язык конечного автомата (ср. с (7.5)).

Таким образом, алгебраическая техника анализа ориентированных графов, размеченных над полукольцами (см. **5**), дает возможность чисто алгебраически решать задачи анализа конечных автоматов.

Пример 7.8. Найдем язык конечного автомата, изображенного на рис. 7.10. Запишем для этого автомата матрицу A меток дуг и систему уравнений:

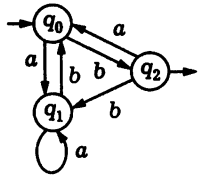


Рис. 7.10

$$A = \begin{pmatrix} 0 & a & b \\ b & a & 0 \\ a & b & 0 \end{pmatrix}, \qquad \begin{cases} x_0 = ax_1 + bx_2, \\ x_1 = bx_0 + ax_1, \\ x_2 = ax_0 + bx_1 + \lambda \end{cases}$$

(слагаемое λ добавлено в уравнение для x_2, так как вершина q_2 является заключительной).

Исключая x_0, получаем

$$\begin{cases} x_1 = b(ax_1 + bx_2) + ax_1, \\ x_2 = a(ax_1 + bx_2) + bx_1 + \lambda, \end{cases}$$

откуда

$$\begin{cases} x_1 = (ba + a)^* b^2 x_2, \\ x_2 = (a^2 + b)(ba + a)^* b^2 x_2 + abx_2 + \lambda. \end{cases}$$

Тогда

$$\begin{cases} x_2 = ((a^2 + b)(ba + a)^* b^2 + ab)^*, \\ x_1 = (ba + a)^* b^2 ((a^2 + b)(ba + a)^* b^2 + ab)^*. \end{cases}$$

Отсюда получаем регулярное выражение, обозначающее язык конечного автомата, как значение переменной x_0:

$$x_0 = a(ba + a)^* b^2 ((a^2 + b)(ba + a)^* b^2 + ab)^* + \\ + b((a^2 + b)(ba + a)^* b^2 + ab)^*.$$

Как видим, полученное регулярное выражение весьма сложно и найти его, не располагая заранее разработанным алгоритмом, было бы затруднительно.

7.6. Детерминизация конечных автоматов

Для дальнейшего изучения свойств *конечных автоматов* и, в частности, для решения *задачи синтеза* важное значение имеет следующая теорема.

Теорема 7.7 (*теорема о детерминизации*). Для любого конечного автомата может быть построен *эквивалентный* ему *детерминированный* конечный автомат.

◀ Для того чтобы доказать теорему, нужно, во-первых, описать алгоритм построения детерминированного конечного автомата по исходному; во-вторых, обосновать этот алгоритм, строго доказав, что он действительно дает конечный автомат,

который является детерминированным и эквивалентным исходному. Здесь мы приведем только сам алгоритм построения детерминированного автомата. Его обоснование дано в **Д.7.1**.

Преобразование произвольного конечного автомата к эквивалентному детерминированному осуществляется в два этапа: сначала удаляются *дуги* с *меткой* λ, затем проводится собственно детерминизация.

1. Удаление λ-*переходов* (дуг с меткой λ).

Чтобы перейти от исходного конечного автомата $M = (V, Q, q_0, F, \delta)$ к эквивалентному конечному автомату $M' = (V, Q', q_0, F', \delta')$ без λ-переходов, достаточно в исходном графе M проделать следующие преобразования.

а. Все состояния, кроме начального, в которые заходят только дуги с меткой λ, удаляются; тем самым определяется множество Q' конечного автомата M'. Понятно, что $Q' \subseteq Q$. При этом полагаем, что начальное состояние остается прежним.

б. Множество дуг конечного автомата M' и их меток (тем самым и функция переходов M') определяется так: для любых двух состояний $p, r \in Q'$ $p \to_a r$ имеет место тогда и только тогда, когда $a \in V$, а в графе M имеет место одно из двух: либо существует дуга из p в r, метка которой содержит символ a, либо существует такое состояние q, что $p \Rightarrow_\lambda^+ q$ и $q \to_a r$. При этом вершина q, вообще говоря, может и не принадлежать множеству Q', т.е. она может и исчезнуть при переходе к автомату M' (рис. 7.11). Если же $q \in Q'$, то, естественно, в M' сохранится дуга (q, r) и символ a будет одним из символов, принадлежащих метке этой дуги (рис. 7.12).

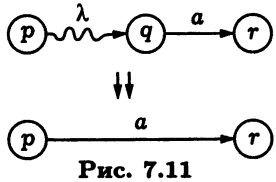


Рис. 7.11

Таким образом, в M' сохраняются все дуги M, метки которых отличны от λ и которые соединяют пару (вершин) состояний из множества Q'(не удаляемых согласно п. а). Кроме этого, для любой тройки состояний p, q, r (не обязательно различных!),

такой, что $p, r \in Q'$ и существует путь ненулевой длины из p в q, метка которого равна λ (т.е. путь по λ-переходам), а из q в r ведет дуга, метка которой содержит символ a *входного алфавита*, в M' строится дуга из p в r, метка которой содержит символ a (см. рис. 7.11).

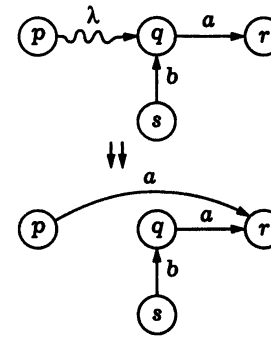


Рис. 7.12

в. Множество заключительных состояний F' конечного автомата M' содержит все состояния $q \in Q'$, т.е. состояния конечного автомата M, не удаляемые согласно п. а, для которых имеет место $q \Rightarrow_\lambda^* q_f$ для некоторого $q_f \in F$ (т.е. либо состояние q само является заключительным состоянием конечного автомата M, либо из него ведет путь ненулевой длины по дугам с меткой λ в одно из заключительных состояний конечного автомата M) (рис. 7.13).

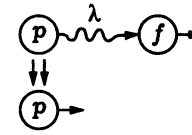


Рис. 7.13

2. Собственно детерминизация.

Пусть $M = (Q, V, q_0, F, \delta)$ — конечный автомат без λ-переходов. Построим эквивалентный M детерминированный конечный автомат M_1.

Этот конечный автомат определяется таким образом, что его множество состояний есть множество всех подмножеств множества состояний конечного автомата M. Это значит, что каждое отдельное состояние конечного автомата M_1 определено как некоторое подмножество множества состояний конечного автомата M. При этом начальным состоянием нового конечного автомата (т.е. M_1) является одноэлементное подмножество, содержащее начальное состояние старого конечного автомата (т.е. M), а заключительными состояниями нового конечного автомата являются все такие подмножества Q, ко-

торые содержат хотя бы одну заключительную вершину исходного конечного автомата M.

Впредь, допуская некоторую вольность речи, мы будем иногда называть состояния конечного автомата M_1 состояниями-множествами. Важно, однако, четко усвоить, что каждое такое состояние-множество есть отдельное состояние нового конечного автомата, но никак не множество его состояний. В то же время для исходного („старого“) конечного автомата M это именно множество его состояний. Образно говоря, каждое подмножество состояний старого конечного автомата „сворачивается“ в одно состояние нового конечного автомата*.

Функция переходов нового конечного автомата определена так, что из состояния-множества S по входному символу a конечный автомат M_1 переходит в состояние-множество, представляющее собой объединение всех множеств состояний старого конечного автомата, в которые этот старый конечный автомат переходит по символу a из каждого состояния множества S. Таким образом, конечный автомат M_1 является детерминированным по построению.

Приведенное выше словесное описание можно перевести в формулы следующим образом: строим конечный автомат M_1 так, что

$$M_1 = (Q_1,\ V,\ \{q_0\},\ F_1,\ \delta_1),$$

где

$$\begin{cases} Q_1 = 2^Q, \quad F_1 = \{T\colon T \cap F \neq \varnothing,\ T \in 2^Q\}, \\ (\forall S \subseteq Q)(\forall a \in V)(\delta_1(S,a) = \bigcup\limits_{q \in S} \delta(q,a)). \end{cases} \tag{7.8}$$

*Формально следовало бы определить множество Q_1 как множество, находящееся во *взаимно однозначном соответствии* с множеством 2^Q, но нам все-таки удобнее считать, что Q_1 совпадает с 2^Q, — ведь множеством состояний конечного автомата может быть любое непустое конечное множество.

Обратим внимание на то, что среди состояний нового конечного автомата есть состояние $\varnothing$, причем, согласно (7.8), $\delta_1(\varnothing, a) = \varnothing$ для любого входного символа a. Это значит, что, попав в такое состояние, конечный автомат M_1 уже его не покинет. Вообще же любое состояние q конечного автомата, такое, что для любого входного символа a $\delta(q, a) = q$, называют ***поглощающим состоянием конечного автомата***. Таким образом, состояние $\varnothing$ детерминированного конечного автомата M_1 является поглощающим. Полезно заметить также, что $\delta_1(S, a) = \varnothing$ тогда и только тогда, когда для каждого $q \in S$ (состояния старого конечного автомата из множества состояний S) $\delta(q, a) = \varnothing$, т.е. в графе M из каждого такого состояния q не выходит ни одна дуга, помеченная символом a.

Можно доказать (см. **Д.7.1**), что полученный по такому алгоритму конечный автомат эквивалентен исходному. ▶

Пример 7.9. Детерминизируем конечный автомат, изображенный на рис. 7.14.

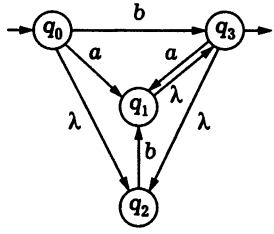


Рис. 7.14

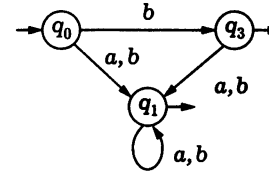


Рис. 7.15

Эквивалентный конечный автомат без λ-переходов изображен на рис. 7.15.

Заметим, что вершина q_2 исчезает, так как в нее заходят только „пустые“ дуги.

Чтобы детерминизировать полученный автомат, совершенно не обязательно выписывать все его $2^3 = 8$ состояний, среди которых многие могут оказаться не достижимыми из начального состояния $\{q_0\}$. Чтобы получить достижимые из $\{q_0\}$

состояния, и только их, воспользуемся так называемым методом вытягивания.

Этот метод в общем случае можно описать так.

В исходном конечном автомате (без пустых дуг) определяем все множества состояний, достижимых из начального, т.е. для каждого входного символа a находим множество $\delta(q_0,a)$. Каждое такое множество в новом автомате является состоянием, непосредственно достижимым из начального.

Для каждого из определенных состояний-множеств S и каждого входного символа a находим множество $\bigcup\limits_{q\in S}\delta(q,a)$. Все полученные на этом шаге состояния будут состояниями нового (детерминированного) автомата, достижимыми из начальной вершины по пути длины 2. Описанную процедуру повторяем до тех пор, пока не перестанут появляться новые состояния-множества (включая пустое!). Можно показать, что при этом получаются все такие состояния конечного автомата M_1, которые достижимы из начального состояния $\{q_0\}$.

Для конечного автомата на рис. 7.15 имеем:

$$\delta_1(\{q_0\},a)=\{q_1\};\qquad \delta_1(\{q_0\},b)=\{q_1,q_3\};$$
$$\delta_1(\{q_1\},a)=\{q_1\};\qquad \delta_1(\{q_1\},b)=\{q_1\};$$
$$\delta_1(\{q_1,q_3\},a)=\delta(q_1,a)\cup\delta(q_3,a)=\{q_1\}\cup\{q_1\}=\{q_1\};$$
$$\delta_1(\{q_1,q_3\},b)=\delta(q_1,b)\cup\delta(q_3,b)=\{q_1\}\cup\{q_1\}=\{q_1\}.$$

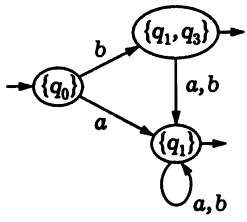


Рис. 7.16

Так как новых состояний-множеств больше не появилось, процедура „вытягивания“ на этом заканчивается, и мы получаем граф, изображенный на рис. 7.16. #

Одним из важных теоретических следствий теоремы о детерминизации является следующая теорема.

Теорема 7.8. *Дополнение регулярного языка* есть регулярный язык.

◀ Пусть L — регулярный язык в алфавите V. Тогда дополнение языка L (как множества слов) есть язык $\overline{L} = V^* \setminus L$.

Согласно теореме 7.7, для регулярного языка L может быть построен детерминированный конечный автомат M, допускающий L. Поскольку в детерминированном автомате из каждой вершины по каждому входному символу определен переход в точности в одну вершину, то, какова бы ни была цепочка x в алфавите V, для нее найдется единственный путь в M, начинающийся в начальном состоянии, на котором читается цепочка x. Ясно, что цепочка x допускается автоматом M, т.е. $x \in L(M)$, тогда и только тогда, когда последнее состояние указанного пути является заключительным. Отсюда следует, что цепочка $x \notin L(M)$ тогда и только тогда, когда последнее состояние указанного пути не заключительное. Но нам как раз и нужен конечный автомат M', который допускает цепочку x тогда и только тогда, когда ее не допускает исходный конечный автомат M. Следовательно, превращая каждое заключительное состояние M в не заключительное и наоборот, получим детерминированный автомат, допускающий дополнение языка L. ▶

Доказанная теорема позволяет строить конечный автомат, не допускающий определенное множество цепочек, следующим методом: строим сначала автомат, допускающий данное множество цепочек, затем детерминизируем его и переходим к автомату для дополнения так, как это указано в доказательстве теоремы 7.8.

Пример 7.10. а. Построим конечный автомат, допускающий все цепочки в алфавите $\{0,1\}$, кроме цепочки 101.

Сначала построим конечный автомат, допускающий единственную цепочку 101. Этот автомат приведен на рис. 7.17.

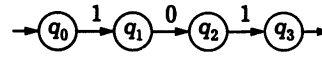


Рис. 7.17

Этот автомат *квазидетерминированный*, но не детерминированный, так как он не *полностью определен*. Проведем его детерминизацию и получим детерминированный эквивалентный конечный автомат, изображенный на рис. 7.18.

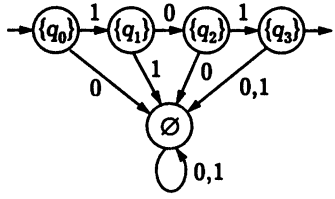


Рис. 7.18

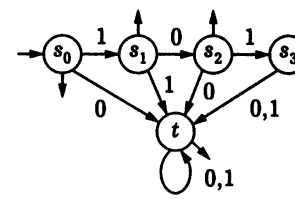


Рис. 7.19

И наконец, переходя к дополнению (и переименовывая состояния), получим автомат, изображенный на рис. 7.19.

Обратим внимание, что в полученном автомате все вершины, кроме вершины s_3, являются заключительными.

Заметим также, что переход к дополнению, о котором идет речь в доказательстве теоремы 7.8, может быть проведен только в детерминированном автомате. Если бы мы поменяли ролями заключительные и незаключительные вершины в автомате, изображенном на рис. 7.17, то получили бы автомат, допускающий язык $\{\lambda, 1, 10\}$, который не является — как нетрудно сообразить — множеством всех цепочек, отличных от цепочки 101.

Отметим также, что конечный автомат на рис. 7.19 допускает все цепочки, содержащие вхождение цепочки 101, но не совпадающие с самой этой цепочкой. Вот, например, путь, несущий цепочку 1011: s_0, s_1, s_2, s_3, t.

б. Построим конечный автомат, допускающий все цепочки в алфавите $\{0, 1\}$, кроме тех, которые содержат *вхождение* цепочки 101. Рассмотрим язык L, каждая цепочка которого содержит вхождение цепочки 101. Его можно задать так:

$$L = (0+1)^*101(0+1)^*.$$

Нам нужно построить автомат для дополнения языка L.

Непосредственно по *регулярному выражению* в этом случае легко построить конечный автомат, допускающий язык L (рис. 7.20).

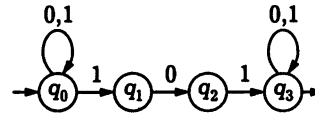


Рис. 7.20

Затем методом „вытягивания“ проведем детерминизацию. Результат детерминизации представлен на рис. 7.21.

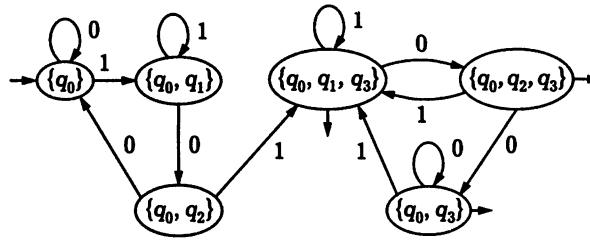


Рис. 7.21

Для полного решения задачи осталось только на рис. 7.21 поменять ролями заключительные и не заключительные вершины (рис. 7.22).

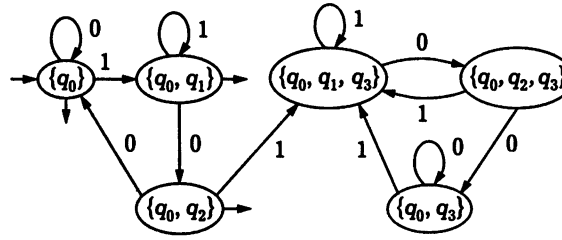


Рис. 7.22

в. Обсудим идею построения конечного автомата, допускающего те и только те цепочки в алфавите $\{0, 1\}$, которые не

начинаются цепочкой 01 и не заканчиваются цепочкой 11 (т.е. не разрешаются цепочки вида $01x$ и цепочки вида $y11$, каковы бы ни были цепочки x, $y \in \{0,1\}^*$).

В этом случае дополнением языка, для которого нужно построить конечный автомат, является множество всех таких цепочек нулей и единиц, которые начинаются цепочкой 01 или заканчиваются цепочкой 11. Допускающий это множество цепочек автомат строится как автомат для объединения $01(0+1)^* + (0+1)^*11$ тем способом, который изложен при доказательстве теоремы Клини (см. теорему 7.6). #

Из свойства замкнутости класса регулярных языков относительно дополнения (см. теорему 7.8) немедленно вытекает замкнутость этого класса относительно пересечения, теоретико-множественной и симметрической разности.

Следствие 7.3. Для любых двух регулярных языков L_1 и L_2 справедливы следующие утверждения:

1) пересечение $L_1 \cap L_2$ регулярно;
2) разность $L_1 \setminus L_2$ регулярна;
3) симметрическая разность $L_1 \triangle L_2$ регулярна.

◀ Справедливость утверждений вытекает из тождеств:

1) $L_1 \cap L_2 = \overline{\overline{L_1} \cup \overline{L_2}}$;
2) $L_1 \setminus L_2 = L_1 \cap \overline{L_2}$;
3) $L_1 \triangle L_2 = (L_1 \cup L_2) \setminus (L_1 \cap L_2)$. ▶

Во-первых, полученные результаты позволяют утверждать, что класс регулярных языков относительно операций объединения, пересечения и дополнения является *булевой алгеброй*, в которой единицей служит универсальный язык, а нулем — пустой язык. Во-вторых, эти алгебраические свойства семейства регулярных языков позволяют решить важную проблему распознавания эквивалентности двух произвольных конечных автоматов.

Согласно определению 7.10, конечные автоматы эквивалентны, если допускаемые ими языки совпадают. Поэтому, чтобы

убедиться в эквивалентности автоматов M_1 и M_2, достаточно доказать, что симметрическая разность языков $L(M_1)$ и $L(M_2)$ пуста. Для этого, в свою очередь, достаточно построить автомат, допускающий эту разность, и убедиться в том, что допускаемый им язык пуст. В общем случае проблему распознавания того, что язык конечного автомата пуст, называют ***проблемой пустоты для конечного автомата.*** Чтобы решить эту проблему, достаточно найти множество заключительных состояний автомата, достижимых из начального состояния. Так как конечный автомат — это ориентированный граф, то решить такую проблему можно, например, с помощью, поиска в ширину (см. **5.5**). Язык, допускаемый конечным автоматом, пуст тогда и только тогда, когда множество заключительных состояний, достижимых из начального состояния, пусто. Практически эквивалентность конечных автоматов предпочтительнее распознавать, используя алгоритм минимизации (см. **7.7**), но сейчас нам важно подчеркнуть, что принципиальная возможность решить проблему эквивалентности вытекает из теоремы 7.7 и ее алгебраических следствий.

7.7. Минимизация конечных автоматов

Может быть поставлен такой вопрос: нельзя ли для произвольного *конечного автомата* построить *эквивалентный конечный автомат* с меньшим числом *состояний*? Оказывается, что ответ на этот вопрос положителен. Более того, можно построить конечный автомат, эквивалентный исходному и имеющий наименьшее число состояний (среди всех конечных автоматов, эквивалентных ему). Процедуру построения такого автомата называют ***минимизацией конечного автомата.***

В силу *теоремы о детерминизации* (см. теорему 7.7) можно считать, что исходный конечный автомат

$$M = (V, Q, q_0, F, \delta)$$

является *детерминированным.*

Будем также предполагать, что в исходном конечном автомате нет состояний, которые не *достижимы* из *начального состояния*. Это предположение мотивировано тем, что если в M существуют состояния, не достижимые из начального, то их можно найти (используя, скажем, *поиск в ширину* на *графе* M) и удалить со всеми *инцидентными* им *дугами*.

На множестве состояний автомата M зададим семейство *отношений эквивалентности* следующим образом:

1) **0-эквивалентность**: для произвольных состояний q_1 и q_2 полагаем $q_1 \equiv^0 q_2$ тогда и только тогда, когда они оба являются заключительными или оба не являются заключительными;

2) ***k*-эквивалентность**: при $k \geqslant 1$ полагаем $q_1 \equiv^k q_2$ тогда и только тогда, когда $q_1 \equiv^{k-1} q_2$, т.е. состояния q_1 и q_2 $(k-1)$-эквивалентны, и, кроме того, для любого входного символа a состояния $\delta(q_1, a)$ и $\delta(q_2, a)$ также $(k-1)$-эквивалентны*.

Чтобы понять смысл отношения k-эквивалентности, обратимся к рис. 7.23. На этом рисунке q_1 и q_2 — $(k-1)$-эквивалентные состояния, т.е. они принадлежат одному и тому же *классу* $(k-1)$-*эквивалентности* C_1. Эти состояния, согласно данному выше определению, станут k-эквивалентными, если для любого входного символа a состояния $\delta(q_1, a)$ и $\delta(q_2, a)$ также являются $(k-1)$-эквивалентными, содержащимися в некотором классе $(k-1)$-эквивалентности C_2 (возможно, что $C_1 = C_2$). Можно сказать, что $(k-1)$-эквивалентные состояния будут также и k-эквивалентными, если переход из них по любому входному символу „сохраняет“ $(k-1)$-эквивалентность состояний, т.е. состояния, в которые конечный автомат переходит из $(k-1)$-эквивалентных состояний, снова окажутся $(k-1)$-эквивалентными.

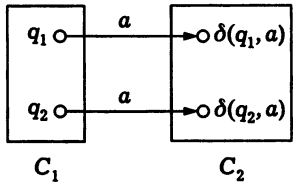


Рис. 7.23

*Напомним, что *функция переходов* детерминированного конечного автомата определена как отображение δ: $Q \times V \to Q$ (см. замечание 7.7).

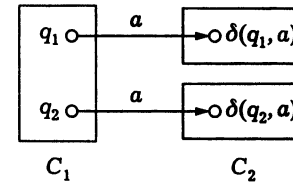


Рис. 7.24

Если же найдется хотя бы один входной символ a, такой, что состояния $\delta(q_1, a)$ и $\delta(q_2, a)$ окажутся в разных классах $(k-1)$-эквивалентности (рис. 7.24), то состояния q_1 и q_2 уже не будут k-эквивалентными (образно говоря, они разойдутся по разным классам k-эквивалентности, так как переход из них по некоторому символу „разрушает“ $(k-1)$-эквивалентность).

Таким образом, для любого $k \geqslant 0$ отношение эквивалентности $\equiv^{k+1}$ включается в отношение $\equiv^k$, и, следовательно, любой класс $(k+1)$-эквивалентности включается в некоторый класс k-эквивалентности (точнее, каждый класс k-эквивалентности или разбивается на несколько попарно непересекающихся классов $(k+1)$-эквивалентности, или, если все его состояния остаются $(k+1)$-эквивалентными, не изменяется). Это значит, что разбиение множества состояний на классы $(k+1)$-эквивалентности является, не более „крупным“, т.е. содержит не меньше классов эквивалентности, чем разбиение на классы k-эквивалентности.

Минимизация конечного автомата состоит в последовательном „измельчении“ разбиения множества Q на классы эквивалентности до тех пор, пока не получится разбиение, которое уже нельзя измельчить (очевидно, что такое разбиение для некоторого $k \leqslant n = |Q|$ всегда существует). Более строго: указанные выше отношения эквивалентности строятся до такого наименьшего k, что отношение $\equiv^k$ совпадет с отношением $\equiv^{k-1}$. Это отношение и определяет самое мелкое разбиение множества состояний. Обозначим его просто $\equiv$. Тогда ***минимальный конечный автомат*** $M = (V', Q', q_0', F', \delta')$, т.е. автомат с наименьшим числом состояний, эквивалентный исходному, определяется следующим образом:

$$V' = V, \quad Q' = Q/_{\equiv}, \quad q_0' = [q_0], \quad F' = \{[f]\colon f \in F\},$$
$$(\forall a \in V)(\forall q \in Q)\delta([q], a) = [\delta(q, a)],$$

где через $[q]$ обозначен класс эквивалентности состояния q по отношению $\equiv$.

Можно доказать вполне строго (аналогично доказательству корректности алгоритма детерминизации в **Д.7.1**), что конечные автоматы M и M' эквивалентны.

Резюмируем полученные результаты в виде теоремы.

Теорема 7.9. Для произвольного конечного автомата может быть построен эквивалентный ему конечный автомат с наименьшим числом состояний. #

Вытекающий из этой теоремы алгоритм минимизации может быть описан так:

1) строим эквивалентный исходному детерминированный конечный автомат;

2) если в полученном конечном автомате остались состояния, не достижимые из начальной вершины, удаляем их (для обнаружения таких вершин может быть использован алгоритм поиска в ширину, см. **5.5**);

3) к полученному конечному автомату применяем изложенный выше алгоритм построения разбиения множества состояний на классы эквивалентности по отношению $\equiv$ и строим минимальный конечный автомат M', как описано выше.

Замечание 7.11. Если детерминизация конечного автомата проводится методом „вытягивания“, то все вершины в детерминированном конечном автомате будут достижимы из начальной вершины.

Кроме того, подчеркнем еще раз, что алгоритм минимизации применяется только к детерминированным конечным автоматам.

Пример 7.11. Минимизируем конечный автомат, изображенный на рис. 7.21.

Введем новые обозначения для состояний автомата:

$$\{q_0\} = s_0, \quad \{q_0, q_1\} = s_1, \quad \{q_0, q_2\} = s_2,$$
$$\{q_0, q_1, q_3\} = s_3, \quad \{q_0, q_2, q_3\} = s_4, \quad \{q_0, q_3\} = s_5.$$

Полученный автомат изображен на рис. 7.25.

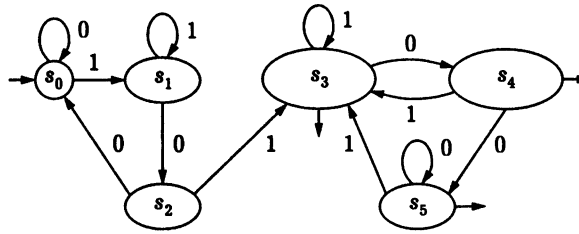


Рис. 7.25

Запишем разбиение множества состояний автомата для отношения $\equiv^0$:

$$\{s_0, s_1, s_2\}, \quad \{s_3, s_4, s_5\}.$$

Так как состояния $\delta(s_2, 0) = s_0$ и $\delta(s_2, 1) = s_3$ не являются 0-эквивалентными состояниями, то в разбиении для 1-эквивалентности они „разойдутся“ по разным классам; разбиение, определяемое отношением $\equiv^1$, будет иметь вид

$$\{s_0, s_1\}, \quad \{s_2\}, \quad \{s_3, s_4, s_5\}.$$

Далее, при переходе к 2-эквивалентности придется „развести“ состояния s_0 и s_1. Поскольку для всех состояний из множества $\{s_3, s_4, s_5\}$ конечный автомат переходит в одно из этих же состояний, то разбиение на классы 2-эквивалентности и есть искомое „мельчайшее“ разбиение:

$$\{s_0\}, \quad \{s_1\}, \quad \{s_2\}, \quad \{s_3, s_4, s_5\}.$$

На рис. 7.26 приведен граф минимального конечного автомата. #

Заметим, что применение рассмотренной процедуры минимизации может дать два крайних случая:

1) все состояния исходного конечного автомата окажутся эквивалентными и тогда в минимальном конечном автомате останется только одно состояние;

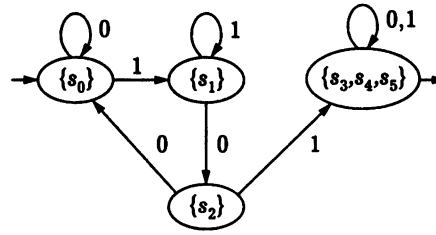


Рис. 7.26

2) итоговое разбиение будет состоять из одноэлементных классов эквивалентности — это означает, что исходный конечный автомат нельзя минимизировать, он уже минимален.

Первый случай проиллюстрирован на рис. 7.27. Здесь приведено построение и минимизация конечного автомата, допускающего язык, обозначенный регулярным выражением $(a^*b^*)^*$.

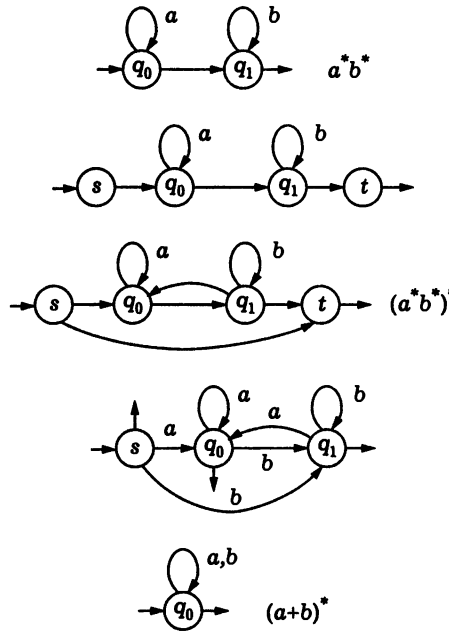


Рис. 7.27

После удаления λ-переходов получается детерминированный автомат, все состояния которого заключительные. Значит, минимальный автомат состоит из одной вершины и допускает язык $(a+b)^*$. Тем самым мы еще и доказали эквивалентность регулярных выражений $(a+b)^*$ и $(a^*b^*)^*$ (в алфавите $\{a, b\}$).

Алгоритм минимизации целесообразно использовать и при распознавании эквивалентности двух заданных конечных автоматов.

Если мы хотим выяснить, эквивалентны ли автоматы M_1 и M_2, то можно минимизировать каждый из них. Если минимальные автоматы M_1' и M_2' имеют множества состояний с разным числом вершин, то исходные автоматы заведомо не эквивалентны. Можно доказать, что исходные автоматы эквивалентны тогда и только тогда, когда соответствующие им ***минимальные автоматы*** M_1' и M_2' ***изоморфны***. Конечные автоматы M_1' и M_2' считаются изоморфными, если существует такая *биекция* h множества состояний первого автомата на множество состояний второго, которая является *изоморфизмом* данных конечных автоматов как *ориентированных графов* (см. **5.7**) и обладает дополнительно следующими свойствами:

1) если q_{01} — начальное состояние конечного автомата M_1', то $h(q_{01}) = q_{02}$ — начальное состояние конечного автомата M_2';

2) если F_1 — подмножество заключительных состояний конечного автомата M_1', то $h(F_1) = F_2$ — подмножество заключительных состояний конечного автомата M_2';

3) для любых двух состояний q и r конечного автомата M_1' и любого входного символа a $q \to_a r$ тогда и только тогда, когда $h(q) \to_a h(r)$.

В общем случае проблема распознавания эквивалентности M_1 и M_2 сводится к проблеме распознавания изоморфизма минимальных автоматов. Для малого числа вершин (до 10) этот изоморфизм, как правило, распознается „невооруженным глазом“, но в общем случае нужны специальные алгоритмы*.

*См., например: *Кристофидес Н.*

7.8. Лемма о разрастании для регулярных языков

В теории формальных языков большое значение имеют утверждения, в которых формулируется необходимое условие принадлежности *языка* к тому или иному классу языков. Эти утверждения известны в литературе под названием лемм о разрастании (или лемм о „накачке"). С помощью этих лемм удается доказать, что тот или иной язык не является языком данного класса, например, не является *регулярным*, *контекстно-свободным* и т.п. Доказывать подобного рода „отрицательные" утверждения гораздо труднее, чем „положительные" (что язык есть *язык* данного *класса* C), ибо в последнем случае требуется придумать любую *грамматику* соответствующего класса, *порождающую* данный *язык*, а в первом нужно каким-то образом доказать, что не существует грамматики этого класса, порождающей язык. Применение лемм о разрастании состоит в следующем: доказав, что предлагаемый язык не удовлетворяет условию леммы о разрастании, мы можем быть уверены в том, что он заведомо не принадлежит соответствующему классу языков, и нам нечего и пытаться искать для него грамматику того же класса.

Мы рассмотрим подобного рода утверждения для классов регулярных, контекстно-свободных и линейных языков.

Начнем с леммы о разрастании для регулярных языков. Эта лемма (см. теорему 7.10) утверждает, что любой регулярный язык допускает представление всех своих достаточно длинных цепочек в виде *соединения* трех цепочек, причем средняя цепочка из этих трех не пуста, ограничена по длине, и ее „накачка" — повторение любое число раз — или выбрасывание не выводит за пределы языка (т.е. дает цепочки, принадлежащие данному регулярному языку).

Теорема 7.10. Если L — регулярный язык, то существует натуральная константа k_L (зависящая от L), такая, что для

любой *цепочки* $x \in L$, *длина* которой не меньше k_L, x допускает представление в виде $x = uvw$, где $v \neq \lambda$ и $|v| \leqslant k_L$, причем для любого $n \geqslant 0$ цепочка $x_n = uv^n w \in L$.

◀ Поскольку язык L регулярен, то, согласно теореме Клини (см. теорему 7.6), существует конечный автомат $M = (V, Q, q_0, F, \delta)$, допускающий его, т.е. $L = L(M)$ (в силу теоремы 7.7 о детерминизации можно считать M *детерминированным* автоматом). Положив $k_L = |Q|$, т.е. введя константу k_L как число состояний конечного автомата M, фиксируем произвольно цепочку $x \in L$, длина l которой не меньше k_L. Так как $l > 0$, то цепочка x не является пустой, и мы можем положить $x = x(1) \ldots x(l)$, $l > 0$.

Поскольку конечный автомат M является детерминированным, то существует единственный *путь*, ведущий из *начального состояния* q_0 в одно из *заключительных состояний* q_f, на котором читается x (рис. 7.28).

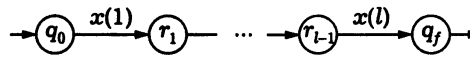


Рис. 7.28

Так как длина l цепочки x не меньше числа состояний M, т.е. числа всех вершин графа M, то, поскольку число вершин в любом пути ровно на единицу больше числа *дуг* в этом пути (т.е. *длины пути*), число вершин в рассмотренном выше пути будет больше, чем число всех вершин графа. Это значит, что хотя бы одна из вершин данного пути повторяется и она, таким образом, в силу следствия 5.1, содержится в некотором *контуре*. Обозначим эту вершину через p. Тогда *путь, несущий цепочку* x, разбивается на три части (рис. 7.29): 1) путь из q_0 в p, 2) контур, проходящий через p, 3) путь из p в q_f.

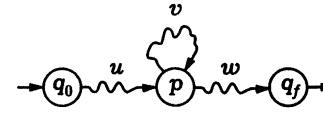


Рис. 7.29

Обозначая через u цепочку, читаемую на первой части пути, через v — цепочку, читаемую на контуре, а через w — цепочку, читаемую на третьей части, получим $x = uvw$, причем поскольку любой контур есть *простой путь*, то $|v| \leqslant k_L$ (длина простого пути не может быть больше, чем число вершин графа) и $v \neq \lambda$, так как контур имеет ненулевую длину. Но теперь совершенно очевидно, что контур (на котором лежит вершина p) можно пройти (по пути из q_0 в q_f) любое число n (при $n > 0$) раз или ни разу. В первом случае на этом пути будет прочитана цепочка uv^nw при $n > 0$, а во втором — и цепочка uw. Таким образом, любая цепочка $x_n = uv^nw$ $(n \geqslant 0)$ содержится в языке L. ▶

Рассмотрим теперь некоторые примеры доказательств нерегулярности языка с использованием леммы о разрастании. Стандартный ход рассуждений при решении таких задач состоит в предположении регулярности данного языка и последующем анализе возможности представить любую достаточно длинную цепочку языка в виде, указанном в условии леммы о разрастании. Анализируя все возможные случаи размещения „накачиваемой“ подцепочки v, мы должны получить противоречие.

Пример 7.12. а. Докажем нерегулярность языка

$$L_1 = \{a^n b^n : n \geqslant 0\}.$$

Выбирая n настолько большим, чтобы оно превосходило k_L (константу леммы), получаем следующие возможные случаи размещения средней подцепочки v в цепочке $a^n b^n$.

1. $v = a^s$, $s < n$, т.е. „накачиваемая“ подцепочка целиком располагается в „зоне символов a“ (рис. 7.30, *а*).

Ясно, что накачка в этом случае выведет за пределы языка, так как при повторении цепочки v число символов a будет неограниченно расти, а число символов b будет оставаться прежним.

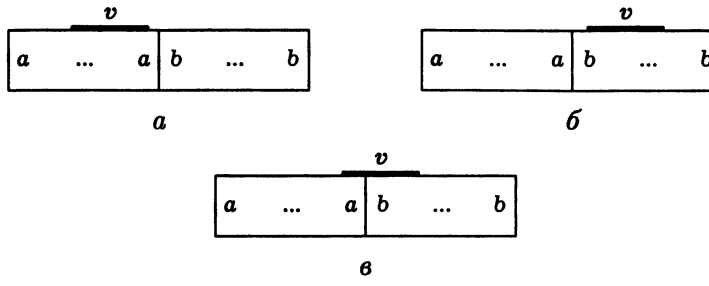


Рис. 7.30

2. $v = b^s$, $s < n$, т.е. „накачиваемая“ подцепочка целиком располагается в „зоне символов b“ (рис. 7.30, *б*).

Накачка невозможна по той же причине, что и в предыдущем случае.

3. $v = a^p b^q$, где $0 < p < n$, $0 < q < n$, т.е. „накачиваемая“ подцепочка находится „на стыке зон символов a и b“ (рис. 7.30, *в*).

В этом случае при накачке возникнет вхождение подцепочки ba в слово, которое, следовательно, уже не принадлежит языку L_1. Таким образом, рассматриваемый язык нерегулярен.

б. Докажем, что язык $L_2 = L_1^*$, где $L_1 = \{a^n b^n : n \geqslant 0\}$, нерегулярный.

Полагая, что язык L_2 регулярен, возьмем слово $(a^n b^n)^m$ для достаточно большого n ($m \geqslant 1$).

Применяя такую же схему рассуждений, как и выше, легко убеждаемся в том, что цепочка v не может состоять из одних символов a или из одних символов b.

Пусть $v = a^r b^s$, где $r + s < n$. Тогда, повторив цепочку v два раза, получим слово $\ldots a^{n-r} a^r b^s a^r b^s b^{n-s} \ldots$ Если $r \neq s$, то цепочка такого вида не может принадлежать языку L_2, так как в любом слове этого языка за подцепочкой символов a следует подцепочка символов b такой же длины. Но даже если $r = s$, то получим подцепочку $a^n b^s$, где $s < n$ (или $a^r b^n$, $r < n$), что также противоречит определению языка L_2. Аналогично рассматривается случай $v = b^r a^s$ $(r + s < n)$ (рис. 7.31). #

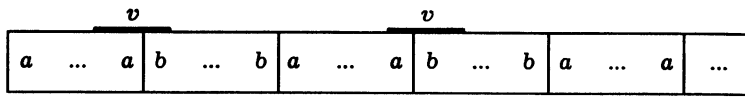


Рис. 7.31

Заметим, что в силу ограничения по длине на цепочку v никакие способы ее размещения в указанной цепочке языка L_2, кроме описанных выше, не возможны. #

Полезно иметь в виду следствие из леммы о разрастании.

Следствие 7.4. Если L — бесконечный регулярный язык, то в нем найдется последовательность слов, длины которых образуют возрастающую арифметическую прогрессию.

◀ В качестве такой последовательности можно взять последовательность слов x_n из формулировки леммы о разрастании. Их длины образуют арифметическую прогрессию со знаменателем $|v|$ (длина „накачиваемой“ средней подцепочки). ▶

Отсюда легко получается вывод о том, что не являются регулярными следующие языки:

1) $\{a^{n^2}: n \geqslant 0\}$ — язык в однобуквенном алфавите, длины слов которого являются полными квадратами;

2) $\{a^p: p$ — простое число$\}$.

Можно доказать, что любой конечный язык регулярен и, более того, допускается конечным автоматом без контуров. Значит, последовательность цепочек, указанная в формулировке леммы, в таком случае не существует, но утверждение леммы остается истинным. Нужно всегда помнить простое логическое правило: утверждение вида „если A, то $Б$“ равносильно утверждению вида „не A или $Б$“.

В заключение обсудим еще одну схему доказательства нерегулярности языка с использованием как леммы о разрастании, так и алгебраических свойств класса регулярных языков, которые были установлены в **7.6**.

Пример 7.13. Докажем нерегулярность языка правильных скобочных структур, порождаемых КС-грамматикой

$$\langle \{(,)\}, \{S\}, S, \{S \to ()\,|\,(S)\,|\,SS\} \rangle$$

(см. грамматику G_3 в примере 7.5). Для доказательства поступим так: рассмотрим пересечение данного языка с регулярным языком $(^*\,)^*$, который содержит все цепочки вида $(^m)^n$ для любых натуральных m, $n \geqslant 0$. Поскольку каждая правильная скобочная структура содержит одинаковое число символов „(“ и „)“, то указанное пересечение будет языком $\{(^n)^n\colon n \geqslant 0\}$. Если обозначить через a „открывающую скобку“ (символ „(“), а через b „закрывающую скобку“ (символ „)“), то получим язык L_1, который, как только что доказано, не является регулярным. Следовательно, предполагая регулярность языка правильных скобочных структур, мы вынуждены будем допустить и регулярность языка L_1, так как пересечение регулярных языков в силу следствия 7.3 регулярно. Полученное противоречие и доказывает нерегулярность исходного языка. Заметим, что доказательство этого факта с использованием одной лишь леммы о разрастании было бы весьма затруднительно.

Замечание 7.12. С использованием „техники пересечения“ можно доказать нерегулярность языка L_2 из примера 7.12 следующим образом. Так как язык

$$L_1 = L_2 \cap a^*b^* = \{a^n b^n\colon n \geqslant 0\}$$

нерегулярный, то и язык L_2 тоже не является регулярным. #

Итак, можно вывести такой „рецепт“: если возникают существенные затруднения в доказательстве нерегулярности какого-либо языка с помощью только леммы о разрастании, то можно попытаться пересечь этот язык с некоторым регулярным языком так, чтобы нерегулярность пересечения легко доказывалась с использованием леммы о разрастании.

Дополнение 7.1. Обоснование алгоритма детерминизации конечных автоматов

В этом дополнении мы подробно докажем корректность приведенного в доказательстве теоремы 7.7 о детерминизации алгоритма построения по заданному *конечному автомату эквивалентного* ему *детерминированного конечного автомата.*

Сначала докажем корректность алгоритма удаления λ-*переходов*, после чего подобное доказательство проведем уже для самого алгоритма детерминизации.

1. Корректность алгоритма удаления λ-переходов. Пусть $M = (V, Q, q_0, F, \delta)$ — исходный конечный автомат, а $M' = (V, Q', q_0, F', \delta')$ — конечный автомат без λ-переходов, построенный согласно алгоритму, описанному в доказательстве *теоремы 7.7 о детерминизации.* Мы должны доказать, что $L(M) = L(M')$.

а. Докажем, что для любых цепочки $x \in V^*$ и состояния $q \in Q$ из того, что в конечном автомате M *цепочка* x *читается на* некотором *пути из* q_0 в q, т.е. имеет место $q_0 \Rightarrow_x^* q$, следует, что в конечном автомате M' эта цепочка читается на некотором пути из q_0 в такое состояние $p \in Q'$, что в M имеет место $p \Rightarrow_\lambda^* q$, т.е. либо $p = q$, либо в M существует путь ненулевой длины по пустым дугам из p в q: $p \Rightarrow_\lambda^+ q$ (на рис. 7.32 и на аналогичных следующих рисунках мы соединяем тонкой штриховой линией кружки, изображающие одно и то же состояние, которое остается после удаления λ-переходов).

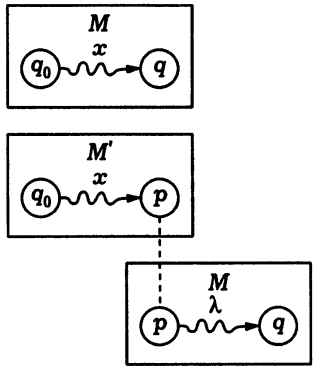


Рис. 7.32

Возможны два случая для состояния $q \in Q$: оно или остается

после удаления λ-переходов, т.е. $q \in Q'$, либо удаляется в результате удаления λ-переходов, т.е. $q \notin Q'$.

1°. Состояние q остается после удаления λ-переходов, т.е. $q \in Q'$.

Проведем индукцию по длине пути в конечном автомате M, на котором читается цепочка x. Для пути нулевой длины доказываемое тривиально. Пусть для всех $k \leqslant n-1$ доказываемое свойство имеет место, и пусть цепочка x читается в M на некотором пути длины n из q_0 в q, т.е. $q_0 \Rightarrow_x^n q$. Тогда существует такое состояние $r \in Q$, что в M имеет место

$$q_0 \Rightarrow_y^{n-1} r \to_a q, \tag{7.9}$$

причем $x = ya$ и $a \in V \cup \{\lambda\}$.

Если $a = \lambda$, то $x = y$, и тогда цепочка x читается в конечном автомате M на некотором пути длины $n-1$.

При $r \in Q'$ остается только использовать индукционное предположение, и тогда найдется такое состояние $r' \in Q'$, что в M' $q_0 \Rightarrow_x^* r'$ и в M $r' \Rightarrow_\lambda^* r$, но так как в M $r \to_\lambda q$, то в M выполняется $r' \Rightarrow_\lambda^+ q$ (рис. 7.33). Таким образом, мы, исходя из условия, что в M $q_0 \Rightarrow_x^n q$, нашли такое состояние $r' \in Q'$, что в M' имеет место $q_0 \Rightarrow_x^* r'$, а при этом в M выполняется $r' \Rightarrow_\lambda^+ q$, что и требовалось доказать.

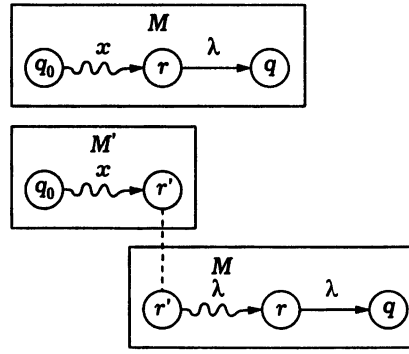


Рис. 7.33

Пусть теперь $r \notin Q'$, т.е. состояние r удаляется при удалении λ-переходов. Это значит, что в состояние r заходят только дуги с меткой λ. Но поскольку на некотором пути из q_0 в r в конечном автомате M читается цепочка y, то найдется такое состояние $p \in Q'$, что в M $q_0 \Rightarrow_y^m p \Rightarrow_\lambda^* r$, и $m < n-1$ (в частности, может быть $m = 0$ и тогда $p = q_0$). Согласно предположению индукции, отсюда следует, что в M' $q_0 \Rightarrow_{y=x}^* p'$, где $p' \Rightarrow_\lambda^* p$ в M, а так как в M имеет место $p \Rightarrow_\lambda^* r \to_\lambda q$, то в M $p' \Rightarrow_\lambda^+ q$ (рис. 7.34), и мы снова, исходя из условия, что в M $q_0 \Rightarrow_x^n q$, нашли в Q' такое состояние p', что цепочка x читается в M' на некотором пути из начального состояния в p', при том что в исходном конечном автомате M из этого состояния p' ведет путь по пустым дугам в состояние q.

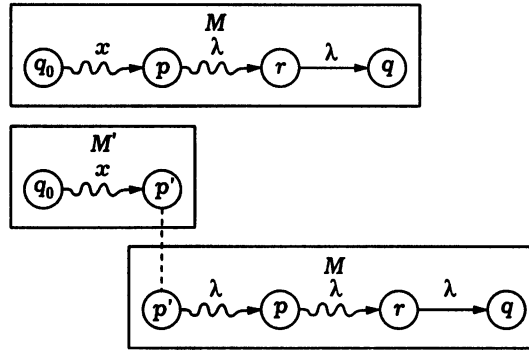


Рис. 7.34

Итак, случай $a = \lambda$ в (7.9) проанализирован полностью.

Пусть $a \neq \lambda$. Если при этом $r \in Q'$, то, согласно предположению индукции, существует такое состояние $r' \in Q'$, что в конечном автомате M' выполняется $q_0 \Rightarrow_y^* r'$, при том что в M $r' \Rightarrow_\lambda^* r$.

При $r' = r$, ввиду того что дуга (r, q) конечного автомата M, метка которой содержит символ a, останется и в конечном автомате M', получаем, что в M' $q_0 \Rightarrow_y^* r \to_a q$, т.е. в M' $q_0 \Rightarrow_x^* q$.

При $r' \neq r$, т.е. в случае, когда $r' \Rightarrow_\lambda^+ r$ в конечном автомате M, заключаем, что в M существует тройка состояний r', r, q, такая, что $r' \Rightarrow_\lambda^+ r$ и $r \to_a q$. По построению конечного автомата M' отсюда получаем, что $r' \to_a q$ в M'. Тогда в M' будет выполняться $q_0 \Rightarrow_y^* r' \to_a q$, т.е. $q_0 \Rightarrow_x^* q$ (рис. 7.35). Случай $r \in Q'$ тем самым полностью проанализирован.

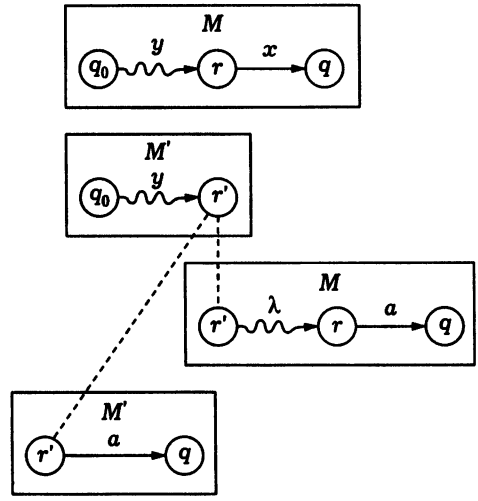


Рис. 7.35

Если же $r \notin Q'$, т.е. состояние r удаляется при переходе к конечному автомату M', то тогда существует такое состояние $p \in Q'$, что в M цепочка y читается на некотором пути длины $m < n - 1$ из q_0 в p, а из p в r ведет путь ненулевой длины по пустым дугам, т.е. в конечном автомате M $q_0 \Rightarrow_p^m \Rightarrow_\lambda^+ r$ (рис. 7.36). Тогда, согласно предположению индукции, в M' $q_0 \Rightarrow_y^* p'$, где $p' \Rightarrow_\lambda^* p$ в M. Тогда опять в M возникает тройка состояний p', r, q, такая, что $p' \Rightarrow_\lambda^+ r$ и $r \to_a q$ (см. рис. 7.36). По построению конечного автомата M' в нем будет выполнено $p' \to_a q$. Итак, в M' $q_0 \Rightarrow_y^* p' \to_a q$, т.е. в конечном автомате M' цепочка x читается на некотором пути из q_0 в q: $q_0 \Rightarrow_x^* q$.

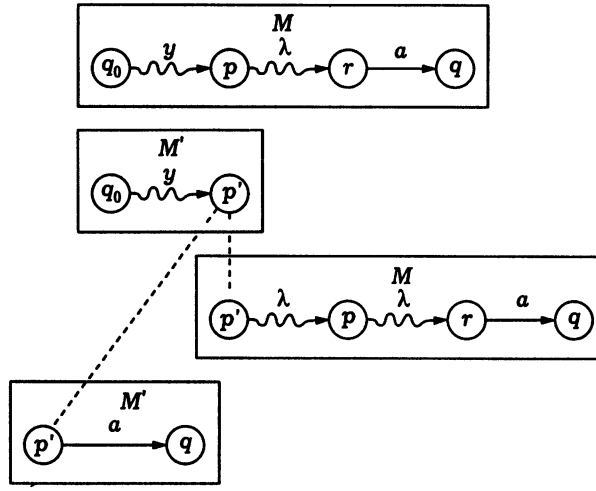


Рис. 7.36

2°. Состояние q удаляется при удалении λ-переходов, т.е. $q \notin Q'$.

В этом случае для некоторого $r \in Q'$ будет выполняться $q_0 \Rightarrow_x^* r \Rightarrow_\lambda^+ q$ в конечном автомате M, откуда, согласно результатам, доказанным для случая 1°, $q_0 \Rightarrow_x^* r'$ в конечном автомате M' для некоторого $r' \in Q'$, такого, что в M $r' \Rightarrow_\lambda^* r$. Следовательно, $r' \Rightarrow_\lambda^* r \Rightarrow_\lambda^+ q$ в M, т.е. $r' \Rightarrow_\lambda^* q$, что и требовалось доказать.

Итак, мы полностью доказали, что любая цепочка x, читаемая в исходном конечном автомате M на некотором пути из начального состояния q_0 в какое-то состояние q, читается также и в автомате M' на некотором пути из начального состояния q_0 в такое состояние p, что в M имеет место $p \Rightarrow_\lambda^* q$.

б. Докажем, что для любых состояния $q \in Q'$ и цепочки $x \in V^*$ из того, что в конечном автомате M' цепочка x читается на некотором пути из начального состояния q_0 в какое-то состояние q, т.е. имеет место $q_0 \Rightarrow_x^* q$, следует, что в исходном конечном автомате M цепочка x читается также на некотором пути из q_0 в q: $q_0 \Rightarrow_x^* q$ в M.

Проведем опять индукцию по длине пути в конечном автомате M', на котором читается цепочка x. Для пути нулевой длины доказываемое тривиально. Предполагая, что доказываемое верно для любой длины пути, не превосходящей $n-1$, допустим, что в M' $q_0 \Rightarrow_x^n q$. Тогда для некоторого $r \in Q'$ в M' имеет место $q_0 \Rightarrow_y^{n-1} r \to_a q$, причем $ya = x$, а так как в M' нет λ-переходов, то $a \in V$, т.е. a не может быть пустой цепочкой. Согласно предположению индукции, отсюда следует, что в M $q_0 \Rightarrow_y^* r$. Далее, из того, что в M' есть дуга из r в q, на которой читается символ a, т.е. $r \to_a q$ в M', следует, что либо эта дуга есть и в исходном конечном автомате M, и тогда $r \to_a q$ в M, либо в M существует такое состояние p, что $r \Rightarrow_\lambda^+ p \to_a q$. Как в том, так и в другом случае имеем $q_0 \Rightarrow_y^* r \Rightarrow_a^+ q$ в конечном автомате M, т.е. в M цепочка x читается на некотором пути из начального состояния в состояние q: $q_0 \Rightarrow_x^* q$.

в. Пусть цепочка $x \in L(M)$, т.е. для некоторого заключительного состояния $f \in F$ цепочка x читается на некотором пути из начального состояния в состояние f: $q_0 \Rightarrow_x^* f$. Тогда из п. а следует, что в M' цепочка x читается на некотором пути из q_0 в такое состояние f', что $f' \Rightarrow_\lambda^* f$ в M. Если $f' = f$, то $f' \in F'$; если же $f' \neq f$, т.е. в M существует путь ненулевой длины по пустым дугам из f' в f, то, согласно определению множества F', $f' \in F'$. Итак, $x \in L(M')$.

Обратно, если $x \in L(M')$, т.е. в M' имеет место $q_0 \Rightarrow_x^* f'$, где $f' \in F'$, то, согласно п. б, $q_0 \Rightarrow_x^* f'$ и в M. Но так как в множество F' попадают либо заключительные вершины конечного автомата M, либо те его вершины, из которых заключительная вершина достижима по пустым дугам, то найдется такое $f \in F$, что в M $f' \Rightarrow_\lambda^+ f$ и $q_0 \Rightarrow_x^* f' \Rightarrow_\lambda^+ f$, т.е. $q_0 \Rightarrow_x^* f$ в конечном автомате M, откуда $x \in L(M)$.

Итак, $L(M) = L(M')$, что и обосновывает корректность алгоритма удаления λ-переходов.

2. Корректность алгоритма детерминизации. Пусть теперь $M = (V, Q, q_0, F, \delta)$ — исходный конечный автомат без

λ-переходов, а $M' = (V, Q', q_0', F', \delta')$ — детерминированный конечный автомат, построенный согласно алгоритму, описанному в доказательстве теоремы о детерминизации, т.е. $Q' = 2^Q$, $q_0' = \{q_0\}$, $F' = \{S: S \cap F \neq \varnothing\}$, и для любого $S \subseteq Q$ и любого $a \in V$ $\delta'(S,a) = \bigcup\limits_{q \in S} \delta(q,a)$.

Мы должны доказать, что $L(M) = L(M')$.

а. Докажем, что для любых цепочки $x \in V^*$ и состояния $q \in Q$ из того, что в M цепочка x читается на некотором пути из начального состояния q_0 в какое-то состояние q, т.е. $q_0 \Rightarrow_x^* q$, следует, что эта цепочка читается и в M' на некотором пути из состояния $\{q_0\}$ в *состояние-множество* S, которое содержит q, т.е. в M' $\{q_0\} \Rightarrow_x^* S$, где $q \in S$.

Доказательство проводим индукцией по длине пути в конечном автомате M, на котором читается цепочка x (так как в автоматах уже нет пустых дуг, т.е. λ-переходов, то длина пути всегда совпадает с длиной цепочки, читаемой на этом пути; поэтому индукцию по длине пути в данном случае можно рассматривать как индукцию по длине цепочки).

Случай пути длины нуль тривиален. Полагая доказываемое справедливым для всех путей, длина которых не больше $n-1$, допустим, что цепочка x читается в M на некотором пути длины n из q_0 в q, т.е. $q_0 \Rightarrow_x^n q$. Тогда найдется такое $r \in Q$, что

$$q_0 \Rightarrow_y^{n-1} r \to_a q, \tag{7.10}$$

где $x = ya$ и $a \in V$.

Тогда, согласно предположению индукции, в конечном автомате M' цепочка y читается на некотором пути $\{q_0\}$ в такое R, что $r \in R$: $\{q_0\} \Rightarrow_y^* R$. Так как конечный автомат M' является детерминированным по построению, то из его состояния-множества R ведет дуга в некоторое состояние-множество S и метка этой дуги содержит символ a, т.е. $R \to_a S$ в M'. Докажем, что $S \ni q$. Состояние $S = \delta'(R,a)$ есть объединение всех множеств $\delta(p,a)$ при $p \in R$. В частности, при $p = r$ множество

состояний $\delta(r,a)$ в силу (7.10) содержит состояние q. Следовательно, это состояние принадлежит и множеству S, и тогда в M' имеет место $\{q_0\} \Rightarrow_y^* R \to_a S$, т.е. $\{q_0\} \Rightarrow_x^* S \ni q$.

б. Докажем теперь, что для любой цепочки $x \in V^*$ и любого состояния $S \in Q'$ из $\{q_0\} \Rightarrow_x^* S$ в M' следует $(\forall q \in S)(q_0 \Rightarrow_x^* r)$ в M.

Проведем индукцию по длине цепочки x. При $|x| = 0$, т.е. для пустой цепочки x, утверждение выполняется тривиально. Пусть оно верно при всех $k \leqslant n-1$, и пусть $\{q_0\} \Rightarrow_x^n S$ в M'. Отсюда для некоторого $R \in Q'$, т.е. $R \subseteq Q$, в M' выполняется $\{q_0\} \Rightarrow_y^{n-1} R \to_a S$, причем $x = ya$ и $a \in V$. Тогда, согласно предположению индукции, в M $q_0 \Rightarrow_y^* r$ для каждого $r \in R$. Поскольку $S = \delta'(R,a)$, то любой элемент q в S есть элемент некоторого множества $\delta(r,a)$ при $r \in R$, т.е. в M есть дуга $r \to_a q$. Но, как мы только что доказали, для любого состояния $r \in R$ имеет место $q_0 \Rightarrow_y^* r$, т.е. для любого $q \in S$ имеем $q_0 \Rightarrow_y^* r \to_a q$ в конечном автомате M, откуда $(\forall q \in S)(M{:}\, q_0 \Rightarrow_x^* q)$, что и требовалось доказать.

в. Теперь, если $x \in L(M)$, т.е. цепочка x читается в M на некотором пути из начального состояния в одно из заключительных, а именно $q_0 \Rightarrow_x^* f$ для некоторого $f \in F$, то, согласно результатам, доказанным в п. а, в M' цепочка x читается на некотором пути из начального состояния в заключительное состояние S_f, содежащее вершину f: $\{q_0\} \Rightarrow_x^* S_f \ni f$, т.е. $S_f \in F'$ и $x \in L(M')$.

Обратно, если $x \in L(M')$, т.е. $\{q_0\} \Rightarrow_x^* S_f \in F'$ в M', то для любого состояния $q \in S_f$ будем иметь в конечном автомате M $q_0 \Rightarrow_x^* q$. Но состояние-множество S_f обязательно содержит некоторое заключительное состояние исходного конечного автомата M (состояние из подмножества F). Тогда для любого такого состояния $f \in S_f \cap F$ получим $q_0 \Rightarrow_x^* f$ в M, что и означает $x \in L(M)$.

Итак, $L(M) = L(M')$, и тем самым вся процедура детерминизации конечных автоматов полностью обоснована.

Дополнение 7.2. Конечные автоматы с выходом. Структурный синтез

Пусть $M = (V, Q, q_0, F, \delta)$ — *детерминированный конечный автомат.* Модифицируем *метки* его *дуг*, а именно фиксируем произвольно *алфавит* W, который назовем ***выходным*** (хотя он может и совпадать со *входным алфавитом* V конечного автомата M), его *буквы* назовем ***выходными символами*** и для каждой *дуги* e конечного автомата M проделаем следующее: каждому *входному символу* a, принадлежащему *метке дуги* e, сопоставим однозначно *упорядоченную пару* $(a, b) \in V \times W$.

Полученный таким образом *размеченный ориентированный граф* называют ***конечным автоматом с выходом.***

Конечный автомат с выходом может быть определен и иначе, независимо от понятия детерминированного конечного автомата.

Конечный автомат с выходом есть упорядоченная семерка

$$M = (V, W, Q, q_0, F, \delta, \mu),$$

где V — входной алфавит (его элементы — входные символы); W — выходной алфавит (его элементы — выходные символы); Q — конечное множество ***состояний конечного автомата с выходом***; q_0 — выделенное состояние, называемое ***начальным состоянием конечного автомата с выходом***; F — выделенное непустое подмножество состояний, каждый элемент которого называют ***заключительным состоянием конечного автомата с выходом***; δ: $Q \times V \to Q$ — *отображение*, называемое ***функцией переходов конечного автомата с выходом***; μ: $Q \times V \to W$ — отображение, называемое ***функцией выходов конечного автомата с выходом.***

Графом (или ***диаграммой***) ***конечного автомата с выходом*** M называют *ориентированный граф*, множество *вершин* которого совпадает с множеством состояний Q, а множество дуг определяется так: *из вершины* (состояния) q *ведет дуга в вершину* (состояние) r тогда и только тогда, когда

$r = \delta(q, a)$ для некоторого входного символа a (причем, поскольку δ есть отображение, для каждой пары (q, a) указанное состояние r единственное).

Каждой дуге диаграммы конечного автомата с выходом M сопоставляется метка, являющаяся конечным множеством упорядоченных пар из $V \times W$, так, что пара (a, b) принадлежит метке дуги $q \to r$ тогда и только тогда, когда $\mu(q, a) = b$.

Как видно, метка каждой дуги конечного автомата с выходом есть конечное *соответствие, функциональное по второй компоненте* (т.е. *частичное отображение* из V в W). *Область определения* этого *соответствия* называется ***входной меткой дуги***, а область значений — ***выходной меткой дуги*** (диаграммы конечного автомата с выходом).

На рис. 7.37 показана диаграмма конечного автомата с выходом, у которого входной алфавит $V = \{a, b\}$, выходной алфавит $W = \{0, 1\}$, множество состояний $Q = \{q_0, q_1, q_2\}$ при начальном состоянии q_0 и заключительном — q_2. Функции переходов и выходов определяются метками дуг. Обычно при изображении диаграмм упорядоченная пара $(i,\ 0) \in V \times W$ записывается через косую черту: $i/0$.

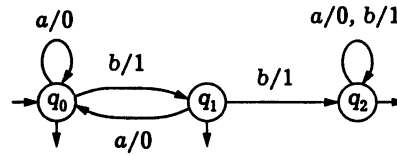


Рис. 7.37

Можно показать, что два сформулированных выше определения конечного автомата с выходом равносильны и тем самым существует *взаимно однозначное соответствие* между конечными автоматами с выходом (в смысле второго определения) и ориентированными графами, дуги которых размечены над *декартовым произведением* двух алфавитов так, как это описано в первом определении. Исходя из этого, мы можем отождествить конечные автоматы с выходом и их диаграммы.

Таким образом, каждому конечному автомату с выходом M однозначно сопоставляется детерминированный конечный автомат $M_V = (V, Q, q_0, F, \delta)$, метки дуг которого получаются „стиранием" всех выходных символов в упорядоченных парах, которыми помечены дуги исходного конечного автомата с выходом (на каждой дуге остается только ее входная метка). Этот детерминированный конечный автомат будем называть ***входной проекцией конечного автомата с выходом*** или ***V-проекцией конечного автомата с выходом*** M. Входная проекция конечного автомата с выходом, диаграмма которого изображена на рис. 7.37, показана на рис. 7.38.

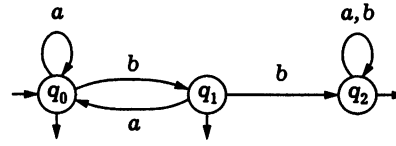


Рис. 7.38

Функцию выходов μ конечного автомата с выходом M можно доопределить до некоторого отображения μ^* из $Q \times V^*$ в W^*. Для этого фиксируем произвольно состояние q и входную цепочку $x \in V^*$. Тогда в силу детерминированности конечного автомата M_V существует единственный *путь* в M_V (и, следовательно, в диаграмме M), ведущий из q в некоторое состояние r, на котором читается цепочка x. Пусть $a \in V$ — входной символ, принадлежащий входной метке некоторой дуги, исходящей из r. Тогда положим $\mu^*(q,x) = \lambda$, если $x = \lambda$; $\mu^*(q,xa) = \mu^*(q,x)\mu(r,a)$ (заметим, что при этом для любых $q \in Q$ и $a \in V$ $\mu^*(q,a) = \mu(q,a)$, т.е. функция выходов конечного автомата с выходом есть *сужение функции μ^* на подмножество* $Q \times V$). В частности, для любой входной *цепочки* x, *допускаемой конечным автоматом* M_V, положим

$$f_M(x) = \mu^*(q_0, x). \tag{7.11}$$

Функция f_M, определенная выражением (7.11), называется ***функцией, вычисляемой конечным автоматом с выходом*** M.

Так, для конечного автомата на рис. 7.37 имеем, например, $f_M(abba) = 0110$. Можно показать, что функция f_M, вычисляемая данным конечным автоматом, есть *биекция регулярного языка* $(a+b)^*bb(a+b)^*$ в алфавите $V = \{a, b\}$ на регулярный язык $(0+1)^*11(0+1)^*$ в алфавите $W = \{0, 1\}$, такая, что для каждой цепочки x первого языка цепочка $f_M(x)$ получается заменой каждого *вхождения* символа a символом 0 и заменой каждого вхождения символа b символом 1.

Функция из V^* в W^* называется ***ограниченно детерминированной функцией*** (или ***ОД-функцией***), если она вычисляется некоторым конечным автоматом с выходом.

Замечание 7.13. Термин „ограниченно детерминированная функция“ связан с тем, что множество всех функций вида $f\colon V^* \to W^*$, вычисляемых конечными автоматами с выходом, есть собственное подмножество множества всех таких функций, которые вычисляются посредством так называемых машин Тьюринга. Машина Тьюринга является самой общей математической моделью „детерминированного преобразователя слов“, т.е. моделью, с помощью которой может быть вычислена любая функция из множества слов в одном алфавите в множество слов в другом алфавите (см. **Д.7.4**). #

Таким образом, каждый конечный автомат с выходом вычисляет некоторую ОД-функцию. С интуитивной точки зрения это значит, что конечный автомат с выходом работает как „детерминированный преобразователь“ слов (цепочек) во входном алфавите в слова (цепочки) в выходном алфавите. В отличие от обычного конечного автомата, который является чистым „распознавателем“, конечный автомат с выходом снабжен выходной лентой, на которую записывается выходная цепочка $f_M(x)$ для заданной входной цепочки x (рис. 7.39).

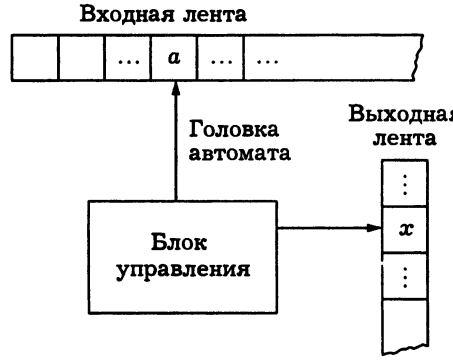


Рис. 7.39

Для конечных автоматов с выходом может быть поставлена ***задача структурного синтеза.*** Содержательно ее можно представить как задачу, аналогичную задаче реализации *булевой функции* или *булева оператора схемой из функциональных элементов.* Однако в отличие от СФЭ, реализующей булев оператор, „схема", которая реализует ОД-функцию (или, что то же, вычисляющий ее конечный автомат с выходом), должна помимо функциональных элементов, каждый из которых вычисляет некоторую булеву функцию из заданного конечного множества функций, содержать так называемые элементы задержки, позволяющие хранить информацию о текущем состоянии автомата.

Чтобы объяснить это, представим себе (на интуитивном уровне) процесс работы конечного автомата с выходом во времени.

В начальный момент времени (время дискретно!) $t = 0$ блок управления конечного автомата с выходом находится в начальном состоянии $q(0) = q_0$. Предположим, что в произвольный момент времени $t \geqslant 0$ состояние есть $q(t)$. Пусть в этот момент времени конечный автомат считывает с входной ленты символ $x(t)$. Тогда в этот же момент времени t на выходной ленте появится выходной символ $y(t)$, а его устройство управ-

ления перейдет в состояние $q(t+1)$, в котором и останется до следующего момента времени $t+1$.

При этом выполняются соотношения

$$\begin{cases} q(t+1) = \delta(q(t), x(t)), \\ y(t) = \mu(q(t), x(t)), \\ q(0) = q_0. \end{cases} \tag{7.12}$$

Эти соотношения называют ***каноническими уравнениями*** данного ***конечного автомата с выходом.***

Из канонических уравнений (7.12) вытекает, что „схема", реализующая конечный автомат с выходом, должна иметь „память" о состоянии устройства управления в предыдущий момент времени. Эта „память" реализуется с помощью так называемых ***элементов задержки***, или ***триггеров.***

Конструированию „схемы" предшествует двоичное кодирование входных и выходных символов, а также состояний устройства управления. Это значит, что каждому состоянию или символу однозначно сопоставляется *булев вектор* (*набор*) соответствующей размерности. В общем виде искомая „схема" должна содержать два блока: назовем их ***комбинационной частью*** (КЧ) и ***запоминающей частью*** (ЗЧ) (рис. 7.40). На вход комбинационной части поступают в каждый момент времени t двоичный код входного символа (булев вектор $X(t)$) и двоичный код состояния от запоминающей части (булев вектор $U(t)$), а с выхода комбинационной части снимается двоичный код выходного символа (булев вектор $Y(t)$) и двоичный код следующего состояния (булев вектор $U(t+1)$).

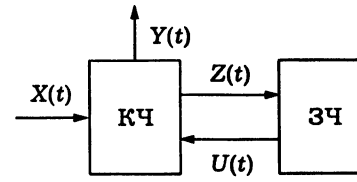


Рис. 7.40

Если через F и G обозначить булевы операторы, сопоставленные в результате указанного выше двоичного кодирования

функциям перехода и выхода соответственно, то канонические уравнения примут вид

$$\begin{cases} Z(t) = F(U(t), X(t)), \\ Y(t) = G(U(t), X(t)), \\ U(t+1) = Z(t), \\ U(0) = U_0, \\ t \geqslant 0. \end{cases} \tag{7.13}$$

Таким образом, комбинационная часть строится как некая схема из функциональных элементов, реализующая булев оператор, определяемый первыми двумя уравнениями (7.13), тогда как запоминающая часть содержит необходимое число триггеров и релизует „задержку“ $U(t+1) = Z(t)$. Каждый триггер есть устройство преобразования „одноразрядного двоичного сигнала“ (с математической точки зрения булева вектора размерности 1), состоящего в задержке этого сигнала „на один такт“. Иначе говоря, если в момент времени t на вход триггера T поступает сигнал $x(t)$, в момент $t+1$ с выхода триггера снимается сигнал $y(t+1) = x(t)$. Обычно рассматривают триггеры с двумя выходами: ***прямым*** и ***инверсным***. Прямой выход работает так, как описано выше, а инверсный выдает сигнал, являющийся *отрицанием* сигнала прямого выхода (рис. 7.41).

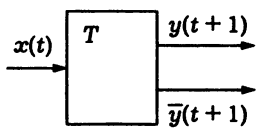


Рис. 7.41

Теперь обсудим вопрос о размерности булевых векторов, кодирующих состояния и символы обоих алфавитов. Произвольное непустое конечное множество $S = \{s_1, \ldots, s_p\}$ может быть „закодировано“ двоичными числами таким образом: нумеруем элементы S от 0 до $p-1$ и эти номера записываем в двоичной системе счисления. Нетрудно сообразить, что тогда разрядность этих чисел (или, что то же самое, размерность соответствующих булевых векторов) составит $]\log_2 p[$ (ближайшее целое, большее $\log_2 p$). Так, если $p = 15$,

то для того, чтобы закодировать числа от 0 до 14, нужны четырехразрядные двоичные коды, поскольку $]\log_2 15[= 4$. Заметим, что в общем случае, поскольку число $\log_2 p$ не будет целым, не все двоичные числа соответствующей разрядности будут использованы. Так, при $p = 15$ не будет использовано число 1111 (двоичный код числа 15).

Тогда размерности булевых векторов X (кодов входных символов), Y (кодов выходных символов), U и Z (кодов текущего и нового состояний) соответственно будут равны: $n =]\log_2 |V|[$, $m =]\log_2 |W|[$ и $k =]\log_2 |Q|[$.

Предлагается следующий алгоритм определения булевых операторов F и G в (7.13).

1. Составляется таблица для функций переходов и выходов исходного конечного автомата с выходом (табл. 7.1).

Таблица 7.1

Текущее состояние	Входной символ	Выходной символ	Новое состояние
⋮	⋮	⋮	⋮
q	a	$b = \mu(q, a)$	$r = \delta(q, a)$
⋮	⋮	⋮	⋮

2. По составленной таблице (см. табл. 7.1) строят так называемую ***структурную таблицу*** (табл. 7.2), в которой каждый символ (входной и выходной) и каждое состояние заменяются их двоичным кодом, причем так, что если набор

Таблица 7.2

U	X	Y	Z
⋮	⋮	⋮	⋮
$u_1, \ldots, u_k$	$x_1, \ldots, x_n$	$y_1, \ldots, y_m$	$z_1, \ldots, z_k$
⋮	⋮	⋮	⋮

$(u_1, \ldots, u_k)$ есть код текущего состояния q, набор $(x_1, \ldots, x_n)$ есть код входного символа a, то

$$(y_1, \ldots, y_m) = G\big((u_1, \ldots, u_k), (x_1, \ldots, x_n)\big)$$

тогда и только тогда, когда набор $(y_1, \ldots, y_m)$ есть код выходного символа $b = \mu(q,a)$, а

$$(z_1, \ldots, z_k) = F\big((u_1, \ldots, u_k), (x_1, \ldots, x_n)\big)$$

тогда и только тогда, когда набор $(z_1, \ldots, z_k)$ есть код состояния $r = \delta(q,a)$.

Структурная таблица есть не что иное, как таблица, задающая некоторый булев оператор (вообще говоря, *частичный*, так как не все векторы соответствующей размерности служат кодами элементов множеств V, W, Q) из $\mathbb{B}^{k+n}$ в $\mathbb{B}^{m+k}$. Этот оператор может быть реализован некоторой схемой из функциональных элементов (как правило, над *стандартным базисом*). Вектор Z поступает (в некоторый момент времени t) на входы k триггеров, с прямых выходов которых (в момент времени $t+1$) снимается вектор U, т.е. $U(t+1) = Z(t)$. Таким образом, если в начальный момент времени $t = 0$ с выхода запоминающей части снимается вектор U_0, код начального состояния q_0 и на вход комбинационной части поступит вектор $X(0)$, то на вход запоминающей части в этот же момент времени поступит вектор $Z(0) = F(U(0), X(0)) = U(1)$ и триггеры запоминающей части будут хранить уже информацию о новом состоянии до момента времени $t = 1$ и т.д.

В этом небольшом дополнении мы никак не можем сколько нибудь подробно и строго обсуждать математическую теорию реализации ОД-функций „схемами“ с элементами задержки*. Разумеется, нет и речи о каких-либо доказательствах. Также мы не решаем „инженерно-технические“ проблемы структурного синтеза, в частности проблемы „аппаратной реализации“

*См., например: *Яблонский С.В.*; *Гаврилов Г.П., Сапоженко А.А.*

триггеров. Наша цель здесь — показать в рамках самого элементарного изложения связь между теорией конечных автоматов и теорией булевых функций. Эта связь состоит в том, что теория булевых функций дает аппарат для структурного синтеза конечных автоматов (с выходом), т.е. для перехода от описания функции, вычисляемой конечным автоматом, к его структуре, реализующей его „схеме“, построенной на функциональных элементах и элементах задержки.

В заключение разберем простой пример структурного синтеза.

Пример 7.14. Конечный автомат с выходом задан диаграммой, изображенной на рис. 7.42. С содержательной точки зрения этот автомат работает как простейший „лексический анализатор“, распознавая все цепочки во входном алфавите $V = \{a, b, 0, 1\}$, которые начинаются с „буквы“, т.е. с символа a или b, как „правильные“, тогда как цепочки, начинающиеся с „цифры“ (т.е. с 0 или 1), классифицируются как „неправильные“, о чем выдается сообщение в виде выходного символа „?“.

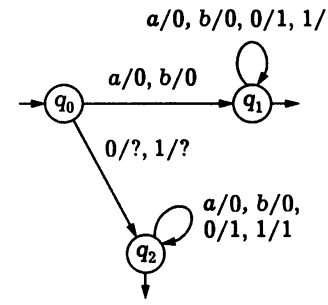


Рис. 7.42

Кодируя входной и выходной алфавиты, а также состояния, получим следующие двоичные коды:

1) входной алфавит: a — 00, b — 01, 0 — 10, 1 — 11;

2) выходной алфавит: 0 — 00, 1 — 01, ? — 10, код 11 не используется;

3) множество состояний: q_0 — 00, q_1 — 01, q_2 — 10, код 11 не используется.

Так как рассматриваемый автомат простой, мы сразу составим структурную таблицу (табл. 7.3).

Таблица 7.3

u_1	u_2	x_1	x_2	y_1	y_2	z_1	z_2
0	0	0	0	0	0	0	1
0	0	0	1	0	0	0	1
0	0	1	0	1	0	1	0
0	0	1	1	1	0	1	0
0	1	0	0	0	0	0	1
0	1	0	1	0	0	0	1
0	1	1	0	0	1	0	1
0	1	1	1	0	1	0	1
1	0	0	0	0	0	1	0
1	0	0	1	0	0	1	0
1	0	1	0	0	1	1	0
1	0	1	1	0	1	1	0

Для *частичных булевых функций* $y_1 = y_1(x_1, x_2, u_1, u_2)$, $y_2 = y_2(x_1, x_2, u_1, u_2)$, $z_1 = z_1(x_1, x_2, u_1, u_2)$, $z_2 = z_2(x_1, x_2, u_1, u_2)$ составим *карты Карно* и найдем для каждой из них *минимальную ДНФ*.

Для функции y_1 карта Карно изображена на рис. 7.43. Единственная склейка дает $y_1 = \overline{u}_1 \overline{u}_2 x_1$.

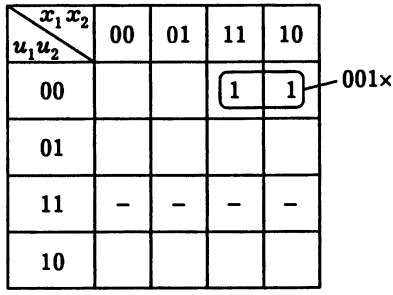


Рис. 7.43

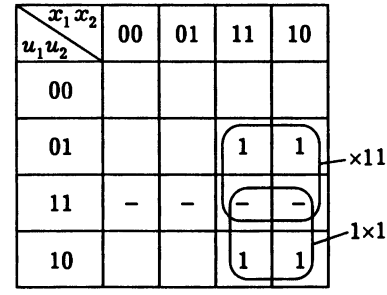


Рис. 7.44

Для второй функции получим (рис. 7.44)

$$y_2 = u_2 x_1 \vee u_1 x_1 = x_1(u_1 \vee u_2).$$

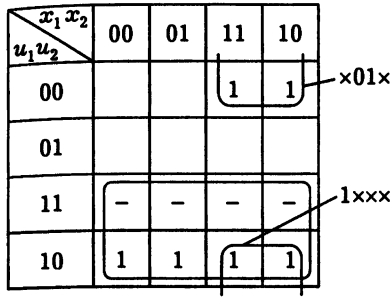


Рис. 7.45

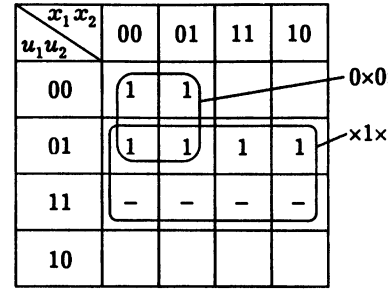


Рис. 7.46

Для функции z_1 имеем (рис. 7.45) минимальную ДНФ

$$z_1 = u_1 \vee x_1\overline{u}_2.$$

Наконец, для функции z_2 по карте Карно, изображенной на рис. 7.46, находим $z_2 = u_2 \vee \overline{x}_1\overline{u}_1$.

Структурная схема автомата представлена на рис. 7.47. Обратим внимание на то, что вместо инверторов для сигналов u_1 и u_2 мы используем сигналы, снимаемые с инверсных выходов триггеров T_1 и T_2. Заметим также, что переменная x_2 оказалась *фиктивной*. Действительно, тип входного символа („буква“, т.е. a или b, и „цифра“, т.е. 0 или 1) распознается по первому разряду входного вектора X: при $x_1 = 0$ имеем „букву“, а при $x_1 = 1$ — „цифру“.

Наконец, следует заметить, что синтезированная „структурная схема“, строго говоря, не является графом (поскольку содержит, например, „кратные дуги“, т.е. допускает несколько разных дуг между одной и той же парой вершин). Даже комбинационная часть этой схемы не может быть уже названа *схемой из функциональных элементов*, так как в вершины, помеченные переменными u_1, u_2, заходят некоторые дуги. Это будет, говоря неформально, схема из функциональных элементов, „вставленная“ в некий более общий „графовый объект“.

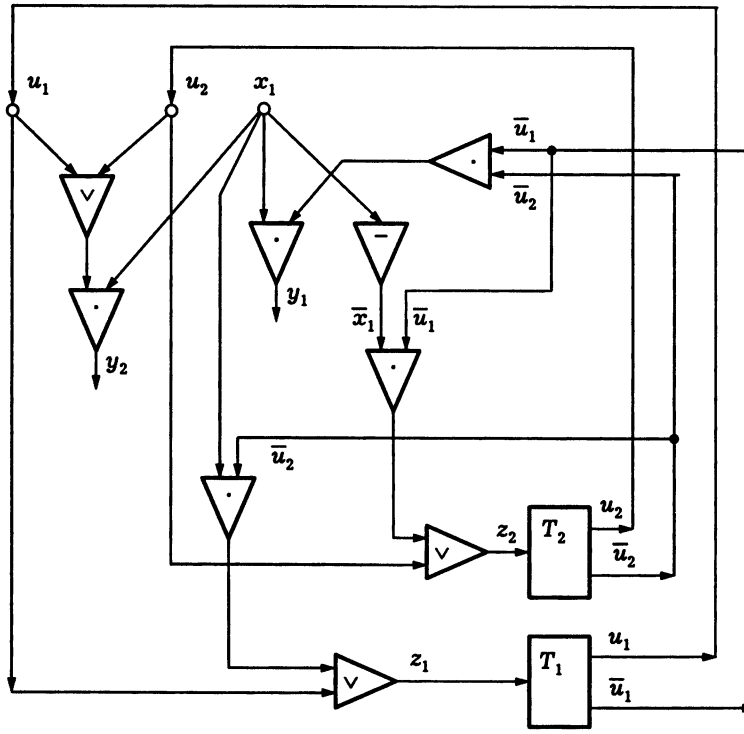


Рис. 7.47

Строгая математическая теория таких „обобщенных графов“ в этом учебнике не рассматривается*.

Дополнение 7.3. Морфизмы и конечные подстановки

Пусть V и W — некоторые алфавиты (в частности, $V = W$). ***Морфизм*** — это произвольное отображение $h\colon V^* \to W^*$, такое, что $h(\lambda) = \lambda$ и $(\forall x, y \in V^*)\ (h(xy) = h(x)h(y))$.

*В литературе для этих объектов также используются термины „сеть“, „гиперграф“, „блок-схема“ (см.: *Яблонский С.В.*).

Иначе говоря, морфизм (в данном контексте) — это *гомоморфизм свободного моноида* V^* в свободный моноид W^* (см. пример 2.7.д).

Теорема 7.11. Любой морфизм $h\colon V^* \to W^*$ однозначно определяется конечным *отображением*

$$\{(a, h(a))\colon a \in V,\ h(a) \in W^*\}. \quad \#$$

Обычно такое отображение задается в форме, напоминающей запись *правил вывода порождающей грамматики*:

$$a_1 \to h(a_1), \quad \ldots, \quad a_n \to h(a_n),$$

где $V = \{a_1, \ldots, a_n\}$. Чтобы найти образ некоторого непустого слова $x \in V^+$, достаточно вместо каждой буквы $x(i)$ подставить слово $h(x(i)) \in W^*$.

Например, если h задается в виде

$$a \to abcba, \quad b \to ba, \quad c \to \lambda,$$

то $h(abbc) = abcbababa$,

$$\begin{aligned} h(h(abbc) = h^2(abbc) = h(abcbababa) = \\ = abcba\,ba\,ba\,abcbaba\,abcba\,ba\,abcba = \\ = abc(ba)^3abc(ba)^2abc(ba)^2abcba. \end{aligned}$$

Морфизм h называется ***λ-свободным морфизмом***, если для всякого слова $x \neq \lambda$ $h(x) \neq \lambda$. Морфизм предыдущего примера не является λ-свободным.

Если $h\colon V^* \to W^*$ — морфизм, то *соответствие* h^{-1} (*обратное* к h) из W^* в V^* называют ***инверсным морфизмом*** (***инверсией морфизма***, ***обратным морфизмом***).

Таким образом, из определений сразу следует, что $\forall y \in W^*$ $h^{-1}(y) = \{x\colon h(x) = y\}$. Для рассмотренного выше примера $h^{-1}(abcbaba) = \{abc^n\colon n \geqslant 0\} = abc^*$.

Определение 7.11. Пусть $h\colon V^* \to W^*$ — морфизм. Тогда для языка $L \subseteq V^*$ язык $h(L) = \{y\colon y = h(x), x \in V^*\}$ называют ***морфизмом языка*** L, а для языка $K \subseteq W^*$ язык $h^{-1}(K) = \{x\colon h(x) \in K, x \in V^*\}$ — ***инверсным морфизмом языка*** L.

Таким образом, язык $h(L)$ есть не что иное, как *образ* языка L *при отображении* h, а $h^{-1}(K)$ — *прообраз* языка K *при отображении* h (см. **1.3**).

Соответствие $\sigma \subseteq V^* \times W^*$ называют ***конечной подстановкой***, если:

1) $\sigma(\lambda) = \{\lambda\}$;

2) для каждого $a \in V$ множество $\sigma(a)$ конечно;

3) для любых цепочек x, $y \in V^*$ $\sigma(xy) = \sigma(x)\sigma(y)$ (т.е. множество слов $\sigma(xy)$ есть *соединение языков* $\sigma(x)$ и $\sigma(y)$).

Иначе говоря, конечная подстановка — это своего рода многозначный морфизм, на любой букве алфавита принимающий лишь конечное множество значений.

Точно так же как и для морфизма, легко показать, что конечная подстановка полностью определяется своими значениями на буквах алфавита V и, следовательно, может быть задана, как и морфизм, в виде системы „правил замены“, в которой одной и той же букве алфавита V сопоставляется, вообще говоря, несколько цепочек в алфавите W. Например:

$$a \to abcba, \quad a \to ac,$$
$$b \to ba, \quad b \to cab,$$
$$c \to aa, \quad c \to \lambda, \quad c \to bab,$$

или в более короткой записи:

$$a \to abcba \,|\, ac, \quad b \to ba \,|\, cab, \quad c \to aa \,|\, \lambda \,|\, bab,$$

как мы записывали правила вывода грамматик и *системы команд конечных автоматов*.

Для нашего примера

$$\sigma(ab) = \{abcbaba, abcbacab, acba, accab\}.$$

Если $\sigma \subseteq V^* \times W^*$ — конечная подстановка, то обратное *соответствие* $\sigma^{-1} \subseteq W^* \times V^*$ называется ***инверсной*** (обратной) ***конечной подстановкой*** (или ***инверсией конечной подстановки***).

Заметим, что для фиксированного $y \in W^*$, согласно определениям обратного соответствия и *сечения* соответствия,

$$\sigma^{-1}(y) = \{x\colon (x,y) \in \sigma\} = \{x\colon y \in \sigma(x)\}.$$

Если $L \subseteq V^*$, $\sigma \subseteq V^* \times W^*$ — конечная подстановка, то

$$\sigma(L) = \{y\colon (\exists x \in L) y \in \sigma(x)\}.$$

Если же $K \subseteq W^*$, то

$$\sigma^{-1}(K) = \{x\colon (\exists y \in K) y \in \sigma(x)\} = \{x\colon \sigma(x) \cap K \neq \varnothing\}.$$

Нетрудно заметить, что, согласно определению *области определения* и *области значения соответствия*, $\sigma(L) \subseteq R(\sigma)$, а $\sigma^{-1}(L) \subseteq D(\sigma) = R(\sigma^{-1})$. Подчеркнем, что не все цепочки множества $\sigma(x)$ при $x \in \sigma^{-1}(K)$ содержатся в языке K, но найдется хотя бы одна цепочка в $\sigma(x)$, которая принадлежит K.

Формулируемая далее теорема связывает конечную подстановку с основными операциями над языками: *объединением языков*, соединением языков и *итерацией языка*.

Теорема 7.12. Если K и L — языки в алфавите V, а $\sigma \subseteq V^* \times W^*$ — конечная подстановка, то

1) $\sigma(K \cup L) = \sigma(K) \cup \sigma(L)$;
2) $\sigma(KL) = \sigma(K)\sigma(L)$;
3) $\sigma(L^*) = (\sigma(L))^*$. #

Основной результат, рассматриваемый в этом дополнении, составляет следующая теорема.

Теорема 7.13. Если $L \subseteq V^*$ и $K \subseteq W^*$ — регулярные языки, $\sigma \subseteq V^* \times W^*$ — конечная подстановка, то $\sigma(L)$ и $\sigma^{-1}(K)$ — регулярные языки в алфавитах W и V соответственно.

◀ Регулярность языка $\sigma(L)$ легко доказывается индукцией по построению регулярного выражения с привлечением теоремы 7.12, и детали этого доказательства нетрудно восстановить.

Доказательство регулярности языка $\sigma^{-1}(K)$ значительно труднее. Мы используем „технику буферов“, которая оказывается полезной в теории формальных языков при доказательстве многих утверждений. Пусть $M = (Q, W, q_0, F, \delta)$ — конечный автомат, допускающий язык K. Построим конечный автомат $M' = (Q', V, s_0, F', \delta')$ следующим образом.

1. $Q' = \{s_0\} \cup \{[q,x]\colon q \in Q, x \in W^{(k)} = W^0 \cup W^1 \cup \ldots \cup W^k\}$, где $k = \max\limits_{\substack{y \in \sigma(a) \\ a \in W}} |y|$, $s_0 \notin Q$.

С интуитивной точки зрения множество Q' состояний конечного автомата M' включает „новое“ состояние s_0 и конечное множество упорядоченных пар из $Q \times W^{(k)}$, где k — наибольшая длина среди всех длин слов из образов букв алфавита V по подстановке σ. Образно говоря, каждое состояние нового автомата определяется состоянием старого автомата (допускающего K) и содержимым „буфера“ конечной длины k для слов из W^*, длина которых не превышает k.

2. $F' = \{[q_f, \lambda]\colon q_f \in F\}$.

3. Система команд δ' содержит команды следующих видов:

(i) $s_0 a \to [q_0, x]$, где $a \in V \cup \{\lambda\}$, $x \in \sigma(a)$;

(ii) $[q, ax] \to [p, x]$ в том и только в том случае, когда δ содержит команду $qa \to p$, $a \in W$, $x \in W^{(k)}$;

(iii) $[q, \lambda]a \to [q, x]$, где $x \in \sigma(a)$, $a \in V$, $q \in Q$.

Неформально работа конечного автомата M' может быть описана следующим образом.

Если входная цепочка $y = \lambda$, то M' допускает цепочку $\lambda \in \sigma^{-1}(\lambda)$. Это соответствует команде вида (i) при $a = \lambda$ (при условии, конечно, что $q_0 \in F$, т.е. что $\lambda \in K$). Если $y \neq \lambda$, то M' прочитает символ $y(1)$ и по команде вида (i) „заполнит буфер“ какой-то цепочкой из множества $\sigma(y(1))$, т.е. перейдет в одно из состояний $[q_0, x_1]$, где $x_1 \in \sigma(y(1))$. Далее M' начинает „моделировать“ работу конечного автомата M

(согласно командам вида (*ii*)), „читая содержимое буфера“ и не обращая внимания на вход, т.е. делая *λ-такты*. Исчерпав цепочку x_1, M' перейдет в состояние $[r_1, \lambda]$, где $q_0 \underset{x_1}{\Rightarrow}^* r_1$ в M. Если $y = y(1)$ и $r_1 \in F$, то $\sigma(y) \cap K \neq \varnothing$ и $y \in \sigma^{-1}(K)$. Иначе автомат может допустить другую цепочку из $\sigma(y(1))$ и т.д. В том случае, если ни из одного состояния $[q_0, x_1]$ нельзя попасть в состояние $[q_f, \lambda]$, $q_f \in F$, однобуквенная цепочка $y \notin \sigma^{-1}(K)$ и M' „зависает“ в незаключительном состоянии.

Если цепочка y имеет второй символ $y(2)$, то по команде вида (*iii*) будет обеспечена „подкачка буфера“ некоторой цепочкой $x_2 \in \sigma(y(2))$ и M' опять начнет работать за конечный автомат M, читая буфер, и т.д.

Нетрудно видеть, что

$$[q_0, x_1] \Rightarrow^* [r_1, \lambda] \Rightarrow [r_1, x_2] \Rightarrow^* [r_2, \lambda] \Rightarrow \ldots \Rightarrow^* [r_m, \lambda],$$

$r_m \in F$ при $x_i \in \sigma(y(i))$ для всех $i = 1, \ldots, m$ и $|y| = m$ тогда и только тогда, когда цепочка $x_1 x_2 \ldots x_m$ читается на некотором пути из q_0 в r_m, т.е. когда $\sigma(y) \cap K \neq \varnothing$ и $y \in \sigma^{-1}(K)$.

Строгое доказательство равенства $L(M') = \sigma^{-1}(K)$ основано на индукции по длине пути в графе автомата. Это доказательство мы не приводим. ▶

Следствие 7.5. Если $L \subseteq V^*$ и $K \subseteq W^*$ — регулярные языки, $h : V^* \to W^*$ — морфизм, то $h(L)$ и $h^{-1}(K)$ — регулярные языки в алфавитах W и V соответственно. #

С интуитивной точки зрения доказанные результаты означают „устойчивость“ множества регулярных языков относительно преобразований, задаваемых конечными подстановками (в частности, морфизмами).

Дополнение 7.4. Машины Тьюринга

В этом дополнении мы рассмотрим *автомат*, называемый автоматом Тьюринга, или машиной Тьюринга, который является *анализирующей моделью для языков типа* 0. Одновременно

машина Тьюринга является одним из математических определений алгоритма.

Определение 7.12. ***Машина Тьюринга*** определяется кортежем вида

$$T = (Q,\ V,\ \star,\ \square,\ S,\ L,\ R,\ q_0,\ q_f,\ \delta),$$

где Q — ***конечное множество состояний***; V — ***конечный входной алфавит***; $\star \notin V$ — символ, называемый ***маркером начала ленты***; $\square \notin V$ — символ, называемый ***пустым*** (или пробелом); S, L, $R \notin V$ — символы, называемые ***символами направления движения головки***; $q_0 \in Q$ — начальное состояние; $q_f \in Q$ — заключительное состояние; δ — ***функция переходов***, являющаяся отображением вида

$$\delta\colon Q \times (V \cup \{\star, \square\}) \to 2^{\{Q \times (V \cup \{\star, \square\}) \times \{S, L, R\}\}},$$

причем значение $\delta(q, a)$, коль скоро оно определено, есть конечное (возможно, и пустое) множество упорядоченных троек из соответствующего декартова произведения.

Функция переходов может быть записана в виде ***системы команд***. Каждая команда есть слово вида

$$qa \to rb, M,$$

где $a,\ b \in V \cup \{\star, \square\}$, $M \in \{S, L, R\}$, $\to \notin V \cup \{\star, \square, S, L, R\}$.

Слово qa (до стрелки) называется ***левой частью команды***, а слово rb, M (после стрелки) — ***правой частью команды***.

Неформально работу машины Тьюринга можно представить следующим образом. Машина имеет ***устройство управления***, которое может находиться в одном из состояний множества Q, полубесконечную ленту, разделенную на ячейки, в каждой из которой может быть записан символ из алфавита $V \cup \{\star, \square\}$, причем крайняя левая ячейка хранит символ $\star$, и головку чтения-записи, которая в каждый момент дискретного

времени обозревает какую-то одну ячейку ленты. Символ □ (пробел) не следует путать с пустой цепочкой — это специальный символ, означающий пустую, т.е. не хранящую символов алфавита $V \cup \{\star\}$, ячейку ленты. Команда, записанная выше, разрешает машине Тьюринга, устройство управления которой находится в состоянии q, а головка обозревает ячейку, хранящую символ a, перевести устройство управления в состояние r, записав в обозреваемую ячейку символ b (который может и совпадать с a), и оставить головку на прежнем месте, если $M = S$, сдвинуть ее на одну ячейку влево, если $M = L$, или на одну ячейку вправо, если $M = R$.

Таким образом, в отличие от *конечных автоматов* машина Тьюринга может модифицировать содержимое ленты (т.е. является преобразователем), а также передвигать головку в любом направлении. Впредь условимся говорить о состоянии машины Тьюринга, подразумевая состояние ее устройства управления, и об обозреваемом символе, подразумевая под этим символ той ячейки ленты, которая обозревается в данный момент головкой.

Формально поведение машины Тьюринга описывается в терминах конфигураций.

Определение 7.13. ***Конфигурация машины Тьюринга*** T есть кортеж

$$(q,\ x,\ y) \in Q \times (V \cup \{\star, \square\})^* \times (V \cup \{\star, \square\})^*.$$

Из конфигурации $C = (q,\ x,\ ay)$ непосредственно выводится конфигурация $C' = (r,\ x,\ by)$, если $qa \to rb$, $S \in \delta$.

Из конфигурации $C = (q,\ xc,\ ay)$ непосредственно выводится конфигурация $C' = (r,\ x,\ cby)$, если $qa \to rb$, $L \in \delta$.

Из конфигурации $C = (q,\ x,\ acy)$ непосредственно выводится конфигурация $C' = (r,\ xb,\ cy)$, если $qa \to rb$, $R \in \delta$.

Выводом на множестве конфигураций машины Тьюринга называется произвольная последовательность конфигураций

$C_0, C_1, \ldots, C_n, \ldots$, такая, что $(\forall i \geqslant 0)(C_i \vdash C_{i+1}$, если C_{i+1} существует).

Конфигурация C' выводима из конфигурации C, если существует вывод $C = C_0 \vdash \ldots \vdash C_n = C'$. Число n называется при этом длиной вывода. Все обозначения, касающиеся выводимости на множестве конфигураций машин Тьюринга, остаются такими же, как в случае конечных и магазинных автоматов.

Конфигурация есть тройка (состояние, часть цепочки на ленте до головки, часть цепочки на ленте, первый символ которой обозревается головкой). При этом, по соглашению, любая цепочка из множества $x\square^*$ (для фиксированного $x \in \in (V \cup \{\star, \square\})^*$) отождествляется с цепочкой x, т.е. можно отбрасывать любое число пробелов в конце слова.

Определение 7.14. Машина Тьюринга называется ***детерминированной***, если из каждой конфигурации этой машины непосредственно выводится не более чем одна конфигурация.

Нетрудно видеть, что машина Тьюринга является детерминированной тогда и только тогда, когда в ее системе команд не существует двух разных команд с одной и той же левой частью.

Далее будем рассматривать только детерминированные машины Тьюринга, называя их просто машинами Тьюринга.

Определение 7.15. ***Словарная (вербальная) функция*** в алфавите V есть произвольное частичное отображение множества слов в алфавите V в себя.

Определение 7.16. Будем говорить, что машина Тьюринга T применима к слову $p \in V^*$, если из конфигурации $(q_0, \lambda, \star p)$ выводится конфигурация $(q_f, \lambda, \star s)$ для некоторого $s \in V^*$.

Слово s будем называть в этом случае ***результатом применения машины Тьюринга T к слову p*** и обозначать его $T(p)$. Факт применимости машины T к слову p будем обозначать $!T(p)$; если же T не применима к p, то будем записывать $\neg!T(p)$.

Множество всех слов p, таких, что $!T(p)$, называется ***областью применимости машины Тьюринга*** T.

Определение 7.17. Словарная функция φ в алфавите V называется ***вычислимой по Тьюрингу***, если существует такая машина Тьюринга T_φ с входным алфавитом V', содержащим V, что

$$x \in D(\varphi) \iff !T_\varphi(x) \wedge \big(T_\varphi(x) = \varphi(x)\big).$$

Пример 7.15. Рассмотрим натуральное число n как слово $0|^n$ в алфавите $\{0, |\}$, а сложение двух натуральных чисел зададим как словарную функцию ADD в алфавите $\{0, |, \dagger\}$, преобразующую слово $0|^n \dagger 0|^m$ в слово $0|^{n+m}$. Тогда функция ADD вычислима по Тьюрингу, поскольку приведенная ниже система команд определяет машину Тьюринга для сложения двух натуральных чисел:

$$T_{ADD} = (\{q_0, q_1, q_2, q_3, q_4, q_5, q_6, q_7, q8\}, \{0, |, \dagger\}, \star, \square, S, L, R, q_0, q_5, \delta),$$

где система команд δ имеет следующий вид:

$$\begin{aligned}
&q_0\star \to q_0\star, R, && (1)\\
&q_00 \to q_00, R, && (2)\\
&q_0| \to q_0|, R, && (3)\\
&q_0\dagger \to q_1|, R, && (4)\\
&q_10 \to q_1|, R, && (5)\\
&q_1| \to q_2|, R, && (6)\\
&q_2| \to q_2|, R, && (7)\\
&q_2\square \to q_3\square, L, && (8)\\
&q_3| \to q_4\square, L, && (9)\\
&q_4| \to q_8\square, L, && (10)\\
&q_8| \to q_8|, L, && (11)\\
&q_80 \to q_80, L, && (12)\\
&q_8\star \to q_5\star, S, && (13)\\
&q_1\square \to q_6\square, L, && (14)\\
&q_6| \to q_7\square, L, && (15)\\
&q_7| \to q_8\square, L. && (16)
\end{aligned}$$

Определенная таким образом машина Тьюринга, получая „на вход“ слово $0|^n \dagger 0|^m$, где $n,m \geqslant 0$, оставаясь в состоянии q_0, „бежит“ вдоль ленты слева направо до тех пор, пока не встретит символ $\dagger$ (играющий роль „разделителя“ двух слагаемых). „Увидев“ этот символ, машина перейдет в состояние q_1, заменив символ $\dagger$ „палочкой“ (|). Следующий затем символ 0 (первый символ второго слагаемого) заменяется „палочкой“. Таким образом, после „пробега“ второго слагаемого в результирующем слове будет на две „палочки“ больше, чем следует. Если второе слагаемое не равно 0, то после выполнения команды (5) будет применима команда (6), а после нее — команда (7). Тем самым будет прочитано второе слагаемое до ближайшего пробела, затем применением команд (8)–(10) будут стерты лишние „палочки“, машина вернется к маркеру начала ленты и остановится в заключительной конфигурации. Если же второе слагаемое равно 0 (есть однобуквенное слово 0), то после выполнения команды (5) применяется команда (14), и затем посредством применения команд (15) и (16) машина сотрет две „палочки“ и перейдет в заключительную конфигурацию.

Запишем тогда последовательность конфигураций для прибавления нуля к произвольному числу:

$$
\begin{aligned}
(q_0,\lambda,\star 0||\ldots|\dagger 0) &\vdash (q_0,\star,0||\ldots|\dagger 0)\vdash\\
&\vdash (q_0,\star 0,||\ldots|\dagger 0)\vdash^* (q_0,\star 0||\ldots|,\dagger 0)\vdash\\
&\vdash (q_1,\star 0||\ldots||,0)\vdash (q_1,\star 0||\ldots|||,\square)\vdash\\
&\vdash (q_6,\star 0||\ldots||,|\square)\vdash (q_7,\star 0||\ldots|,|\square\square)\vdash\\
&\vdash (q_8,\star 0||\ldots,|\square\square\square)\vdash^* (q_8,\star,0||\ldots|)\vdash\\
&\vdash (q_8,\lambda,\star||\ldots|)\vdash (q_5,\lambda,\star||\ldots|).
\end{aligned}
$$

Разумеется, предложенная машина Тьюринга для сложения двух натуральных чисел не обязана быть единственной, но это и не требуется по определению вычислимости — достаточно, чтобы нашлась хотя бы одна машина, вычисляющая данную функцию. #

Итак, используя понятие машины Тьюринга, можно дать строгое ***определение функции как процедуры*** (правила, алгоритма) вычисления значения функции по известным значениям ее аргументов. Тем самым понятие машины Тьюринга есть одно из возможных математически точных определений алгоритма.

До построения какого-либо математического определения алгоритма мы имеем дело с ***интуитивным представлением об алгоритме***. Это интуитивное понятие, не являющееся математически строгим (подобно тому как в рамках „наивной" теории множеств не является таковым понятие множества), может быть охарактеризовано следующими признаками:

1) ***признак детерминированности алгоритма*** — алгоритм есть пошагово реализуемая процедура, на каждом шаге которой однозначно определено ее продолжение (другими словами, алгоритм не допускает произвола в выборе очередного шага — в определении алгоритма через машину Тьюринга этот принцип реализуется через понятие детерминированной машины Тьюринга);

2) ***признак результативности алгоритма*** — процесс работы алгоритма с исходными данными должен завершаться через определенное конечное число шагов;

3) ***признак массовости алгоритма*** — алгоритм должен быть применим к определенному достаточно широкому множеству входных данных.

Наряду с машиной Тьюринга известны и другие уточнения понятия „алгоритм" (нормальные алгорифмы Маркова, рекурсивные функции, канонические системы Поста и т.п.). Оказывается, что все эти понятия в определенном смысле эквивалентны. Кроме того, в теории алгоритмов, которая занимается проблемами построения разных уточнений понятия алгоритма и вычислимости, а также сравнением между собой разных уточнений, основной гипотезой является гипотеза о том, что всякая функция, вычислимая в интуитивном смысле слова, т.е. такая, что существует алгоритм (в интуитивном смысле слова)

вычисления значений функции по известным значениям аргументов, будет вычислима и в соответствии с каким-либо точным определением вычислимости, например будет вычислима по Тьюрингу. Применительно к машинам Тьюринга эта гипотеза носит название ***тезиса Тьюринга***: любая вычислимая (в интуитивном смысле) словарная функция вычислима по Тьюрингу.

В пользу тезиса Тьюринга говорит то, что до сих пор не удалось придумать вычислимую функцию, которую нельзя было бы „запрограммировать“ в виде машины Тьюринга, а также то, что все известные до сих пор альтернативные уточнения понятия алгоритма можно (хотя и не так просто) свести к понятию машины Тьюринга.

Чтобы закончить обсуждение машин Тьюринга в контексте теории формальных языков, нам необходимо ввести еще два важнейших понятия — перечислимости и разрешимости.

Условимся в дальнейшем машину Тьюринга со входным алфавитом V называть ***машиной Тьюринга в алфавите*** V, а машину Тьюринга, входной алфавит которой содержит алфавит V и символы, не принадлежащие V, — ***машиной Тьюринга над алфавитом*** V. Договоримся также „кодировать“ натуральные числа словами в алфавите $N = \{0, |\}$, дав множеству $\mathbb{N}$ натуральных чисел следующее определение по индукции:

1) слово 0 есть натуральное число;

2) если известно, что слово n в алфавите N есть натуральное число, то слово $n|$ есть натуральное число;

3) никаких других натуральных чисел, кроме определенных в пп. 1 и 2, не существует.

На множестве определенных таким образом натуральных чисел (как слов) естественно вводится отношение порядка: по определению, $n \leqslant m$, если n есть префикс m. Легко определить и обычные арифметические операции, „запрограммировав“ их машинами Тьюринга (пример операции сложения разобран выше).

Определение 7.18. Непустое множество слов L в алфавите V называется ***перечислимым (по Тьюрингу)***, если существует машина Тьюринга T над алфавитом $V \cup \{0, |\}$, такая, что для всякого $n \in \mathbb{N}$ $!T(n)$ и для всякого $x \in L$ имеет место $x = T(n)$ для некоторого $n \in \mathbb{N}$.

Таким образом, перечислимость языка L означает существование алгоритма („запрограммированного" как машина Тьюринга), по любому натуральному числу n вычисляющего элемент L и такого, что любой элемент L может быть вычислен по некоторому натуральному числу. Такой алгоритм есть алгоритмический („конструктивный") аналог теоретико-множественного понятия *нумерации* как *сюръективного отображения* множества натуральных чисел на некоторое множество.

Замечание 7.14. Пустое множество слов считается перечислимым по соглашению.

Определение 7.19. Множество слов L в алфавите V называют ***разрешимым (по Тьюрингу)***, если существует такая машина Тьюринга T над алфавитом V, что для всякого $x \in V^*$ имеет место $!T(x)$ и $T(x) = \lambda$, если $x \in L$, и $T(x) \neq \lambda$, если $x \notin L$.

Другими словами, разрешимость языка L означает существование вычислимого предиката, определенного на множестве слов в алфавите V и принимающего значение 1, если элемент принадлежит L, и значение 0 в противном случае.

Без доказательства сформулируем следующую теорему.

Теорема 7.14. 1. Существует неперечислимый язык в некотором алфавите V.

2. Существует перечислимый, но неразрешимый язык (в некотором алфавите V).

3. Язык $L \subseteq V^*$ перечислим тогда и только тогда, когда он является областью применимости некоторой машины Тьюринга над алфавитом V.

4. Язык $L \subseteq V^*$ перечислим тогда и только тогда, когда он порождается некоторой грамматикой типа 0.

5. Язык $L \subseteq V^*$ разрешим тогда и только тогда, когда он порождается некоторой КЗ-грамматикой. #

Из теоремы 7.14 следует, что всякое разрешимое множество перечислимо, но обратное неверно.

В силу утверждений 3 и 4 теоремы 7.14 машины Тьюринга можно рассматривать как „распознающие автоматы" для языков типа 0: для каждого такого языка существует машина Тьюринга, применимая к словам данного языка, и только к ним. Заметим при этом, что язык типа 0 может оказаться и неразрешимым: для него может не существовать алгоритма, распознающего принадлежность ему любого наперед заданного слова. Это значит, что для грамматик типа 0 проблема принадлежности в общем случае не является разрешимой. Эта проблема, однако, как следует из утверждения 5 теоремы 7.14, разрешима для любого языка, порождаемого КЗ-грамматикой (см. **7.2**), в частности для любого КС-языка.

Утверждение 1 теоремы 7.14 означает, что существуют языки, которые не могут принципиально быть порождены какой-либо грамматикой. Примером такого языка может служить множество так называемых конструктивных действительных чисел*.

Пример перечислимого, но неразрешимого языка может быть построен следующим образом. Пусть V — алфавит, а ρ — конечное бинарное отношение на множестве V^+ непустых слов в алфавите V. Определим язык L_ρ в алфавите $V \cup \{\star, \dagger\}$, где символы $\star$ и $\dagger$ не принадлежат V:

$$L_\rho = \{x_1 \dagger y_1 \star x_2 \dagger y_2 \star \ldots \star x_n \dagger y_n\colon\ n \geqslant 1,$$
$$x_1 x_2 \ldots x_n = y_1 y_2 \ldots y_n \text{ и } (\forall i = \overline{1,n})(x_i, y_i) \in \rho\}.$$

Доказывается, что язык L_ρ в общем случае неразрешим, т.е. не существует алгоритма, который для любого наперед заданного отношения ρ (при $|V| \geqslant 2$) распознавал бы слова языка L_ρ.

*См.: *Кушнер Б.А.*

Проблема распознавания слов языка L_ρ известна в теории алгоритмов под названием ***проблемы соответствий Поста*** (над алфавитом V). Неразрешимость проблемы соответствий в общем случае не исключает того, что для каких-то конкретных алфавитов и конечных бинарных отношений на множестве непустых слов эту проблему можно решить. Например, пусть $V = \{a, b\}$, а $\rho = \{(aba,\ ab), (b,\ ab)\}$. Нетрудно показать, что в данном случае $L_\rho = \{(aba \dagger ab \star b \dagger ab)^n \colon n \geqslant 1\}$.

Вопросы и задачи

7.1. Доказать, что следующая грамматика, заданная системой правил вывода, порождает любую цепочку вида a^{n^2}, $n \geqslant 1$:

$$\begin{aligned} &S \to aCA, && Ea \to aE,\\ &A \to a^2EA \mid F, && Ca \to aC,\\ &EF \to DF, && CD \to Ca,\\ &ED \to Da^2E, && CF \to \lambda. \end{aligned}$$

7.2. Доказать, что следующая грамматика, заданная системой правил вывода, порождает любое слово в алфавите $\{a,b\}$ вида ww:

$$\begin{aligned} &S \to CD, && Aa \to aA,\\ &C \to aCA \mid bCB \mid \lambda, && Ab \to bA,\\ &D \to \lambda, && Ba \to aB,\\ &AD \to aD, && Bb \to bB,\\ &BD \to bD. \end{aligned}$$

7.3. Доказать, что любая праволинейная грамматика может быть задана эквивалентной регулярной грамматикой.

7.4. Постройте КЗ-грамматику, порождающую язык:

а) $L_1 = \{a^n b^m a^n b^m \colon m, n > 1\}$;

б) $L_2 = \{a^n b^n a^n \colon n > 1\}$.

7.5. Какой язык порождает грамматика

$$S \to aSBa \,|\, aba, \quad aB \to Ba, \quad bB \to bb?$$

7.6. Построить леволинейную грамматику для языка идентификаторов, которые должны содержать от одного до шести символов и начинаться с одной из букв i, j, k, l, m, n.

7.7. Доказать, что любая леволинейная грамматика эквивалентна некоторой праволинейной грамматике.

7.8. Дано конечное множество слов $\{u_1, \ldots, u_k\} \in V^+$. Язык L определяется как множество всех цепочек в V^*, которые не начинаются ни одним из слов $u_1, \ldots, u_k$. Построить регулярную грамматику, порождающую язык L.

7.9. Доказать, что язык $L^{+k} = \bigcup\limits_{i=k>0}^{\infty} L^i$ регулярен для любого k при условии регулярности L.

7.10. Определить, какие множества цепочек задаются регулярными выражениями:

а) $(a+b)^*c^*(b+c)$;

б) $((ab)^+(ca)^*)^*$;

в) $(a^+(b+c)^*a+b^+(a+b)^*bc)^*$.

7.11. Доказать, что для любых регулярных выражений α и β имеет место тождество $(\alpha+\beta)^* = (\alpha^*\beta^*)^*$.

7.12. Инверсия цепочки $x \in V^*$ — это цепочка символов x^R, полученная переписыванием x справа налево, т.е. если $x = a_{j1}\ldots a_{jn}$, то $x^R = a_{jn}\ldots a_{j1}$. Доказать, что если L — регулярный язык, то $L^R = \{y\colon y = x^R, x \in L\}$ — регулярный язык. Привести два варианта доказательства: 1) используя только определение регулярного языка; 2) используя конечные автоматы.

7.13. Доказать, что $\varnothing^* = \lambda$.

7.14. Написать регулярное выражение для множества цепочек в алфавите $\{0, 1\}$, содержащих четное число нулей и четное число единиц.

7.15. Найти языки, допускаемые конечными автоматами, заданными на рис. 7.48.

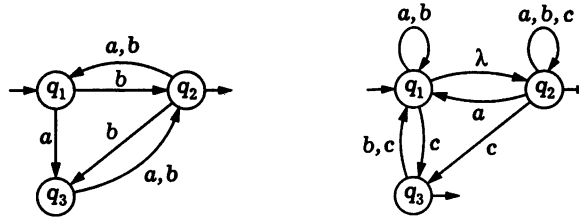


Рис. 7.48

7.16. Доказать, что любой конечный язык регулярен.

7.17. Построить конечный автомат с входным алфавитом $\{0, 1, \ldots, 9\}$, допускающий десятичные записи: а) всех четных чисел; б) нечетных чисел; в) чисел, делящихся на 4; г) чисел, делящихся на 3; д) чисел, делящихся на 9.

7.18. Построить конечный автомат, допускающий десятичные записи тех и только тех натуральных чисел, которые делятся на заданное число k.

Указание: автомат должен выполнять деление „уголком“ и в нем должно быть ровно k состояний.

7.19. Обобщенным конечным автоматом назовем ориентированный граф, размеченный над полукольцом регулярных выражений (с выделенной начальной вершиной и подмножеством заключительных вершин), в котором метка дуги может быть произвольным регулярным выражением. Определяя язык такого автомата аналогично языку обычного конечного автомата, доказать, что для любого обобщенного конечного автомата существует эквивалентный (т.е. допускающий тот же самый язык) конечный автомат.

7.20. Построить конечные автоматы, эквивалентные следующим обобщенным конечным автоматам:

а) вход q_0; выход q_3; дуги с метками $(q_0, q_0, (a+b)c)$, (q_0, q_1, a), $(q_1, q_1, (a+c)b)$, (q_1, q_3, a^*), (q_0, q_2, b^*), (q_2, q_3, b^*), $(q_2, q_2, ca(a+b)^+)$;

б) вход q_0; выход q_3; дуги $(q_0, q_2, (ab)^*)$, $(q_2, q_2, (ca)^*b)$, $(q_0, q_1, \varnothing)$, (q_1, q_3, c^+), $(q_2, q_3, ((b+a)^+c^*))$.

7.21. Решить систему линейных уравнений с регулярными коэффициентами

$$\begin{cases} x_1 = (01^* + 1)x_1 + x_2, \\ x_2 = 1x_1 + 00x_3 + 11, \\ x_3 = x_1 + x_2 + \lambda. \end{cases}$$

Для регулярного выражения, задающего компоненту решения x_3, построить допускающий его конечный автомат.

7.22. Доказать, что линейное уравнение $x = \alpha x + \beta$ с регулярными коэффициентами имеет:

а) единственное решение при $\lambda \notin \alpha$;

б) бесконечно много решений при $\lambda \in \alpha$, причем общее решение можно записать в виде $x = \alpha^*(\beta + L)$, где L — произвольный язык.

7.23. Для произвольного алфавита W определим операцию левого деления на заданную букву, а именно: для произвольных $a \in W$ и $x = x(1)x(2)\ldots x(k) \in W^*$ положим $a^{-1}x = x(2)\ldots x(k)$, если $x(1) = a$, а если $x(1) \neq a$, то считаем, что выражение $a^{-1}x$ не определено. В частности, выражение $a^{-1}\lambda$ всегда не определено. Доказать, что если L — регулярный язык (в алфавите W), то для всякого $a \in W$ язык $a^{-1}L = \{v\colon v = a^{-1}x,\ x \in L\}$ регулярен.

7.24. Аналогично операции левого деления на букву можно определить операцию левого деления на слово, полагая $w^{-1}x = y$, если $x = wy$ (иначе деление не определено). Определяя язык

$w^{-1}L$ так же, как в предыдущей задаче, доказать, что этот язык регулярен для любого регулярного L.

7.25. Определим операцию левого деления языка L_1 на язык L_2, положив

$$L_2^{-1}L_1 = \bigcup_{w \in L2} w^{-1}L_1 = \{y\colon (\exists w \in L_2)(wy \in L_1)\}.$$

Доказать, что если L_1 и L_2 регулярны, то язык ${L_2}^{-1}L_1$ регулярный.

7.26. Пусть L — регулярный язык. Доказать, что

$$INIT(L) = \{w\colon (\exists x)(wx \in L)\}$$

есть регулярный язык.

7.27. Пусть L — регулярный язык. Доказать, что

$$FIN(L) = \{w\colon (\exists x)(xw \in L)\}$$

есть регулярный язык.

7.28. Пусть L — регулярный язык. Доказать, что

$$SUB(L) = \{w\colon (\exists x)(\exists y)(xwy \in L, \text{ где } x,y,w \text{ не пусты})\}$$

есть регулярный язык.

7.29. Построить конечный автомат, допускающий все цепочки в алфавите $\{a, b\}$, не содержащие вхождений подцепочки aba.

7.30. Построить конечный автомат, допускающий все цепочки в алфавите $\{a, b\}$, которые не начинаются с ab и не кончаются ab.

7.31. Построить конечный автомат, допускающий те и только те цепочки в алфавите $\{a, b\}$, которые не допускает

следующий конечный автомат: вход q_1; выход q_3; дуги с метками (q_1, q_1, b), (q_1, q_2, a), (q_2, q_1, b), (q_2, q_3, b), (q_3, q_3, a, b), Для построенного автомата записать регулярное выражение, задающее его язык.

7.32. Построить конечный автомат, допускающий все последовательности нулей и единиц, кроме тех, которые содержат подпоследовательность 11011.

7.33. Построить конечный автомат, допускающий те и только те цепочки в алфавите $\{0, 1\}$, которые не начинаются цепочкой 01 и не заканчиваются цепочкой 11 (т.е. не разрешаются цепочки вида $01x$ и цепочки вида $y11$, каковы бы ни были цепочки $x, y \in \{0, 1\}^*$).

Указание: дополнением языка, для которого нужно построить конечный автомат, является множество всех таких цепочек нулей и единиц, которые начинаются цепочкой 01 или заканчиваются цепочкой 11. Допускающий это множество цепочек автомат строится как автомат для объединения $01(0+1)^* + (0+1)^*11$ способом, который изложен при доказательстве теоремы Клини (см. теорему 7.6).

7.34. Минимизировать по переходам автомат, изображенный на рис. 7.49.

Указание: предварительно удалить недостижимые из q_0 вершины.

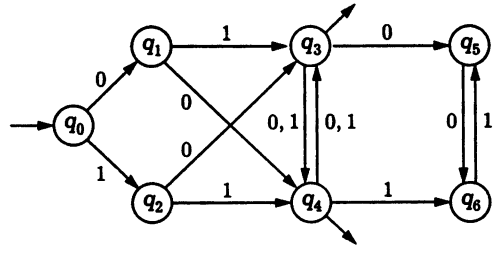


Рис. 7.49

7.35. Используя лемму о разрастании, доказать, что следующие множества не регулярны:

а) $\{0^n10^n\colon n \geqslant 1\}$;

б) $\{ww\colon w \in \{a,b\}^*\}$;

в) множество цепочек в алфавите $\{0,1\}$, являющееся наименьшим решением нелинейного уравнения $x = 0x1 + x^2 + 01$ в полукольце $L(V)$;

г) $\{a^{n^2}\colon n \geqslant 1\}$;

д) $\{w\colon w \in \{a,b,c\}\}^*$ и имеет одинаковое число символов a и b или одинаковое число символов b и c;

е) $\{0^n1^n\colon n \geqslant 0\}^*$;

ж) $\{0^n1^m\colon n \neq m,\ n,m \geqslant 0\}$;

з) множество всех цепочек нулей и единиц, в которых число нулей больше числа единиц.

7.36. Решить задачу о перечеслении путей для ориентированного графа, приведенного на рис. 7.50. Вычислить полностью матрицу путей и описать все ее компоненты словесно.

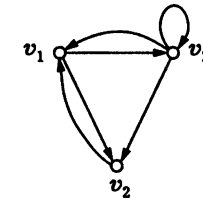


Рис. 7.50

Указание: помечая дуги графа упорядоченными парами, составить систему уравнений с регулярными коэффициентами в алфавите меток дуг для определения каждого из трех столбцов матрицы стоимостей (см. **5.6**). Для данного графа имеем систему уравнений

$$\begin{cases} x_1 = (1,2)x_2 + (1,3)x_3 + \lambda, \\ x_2 = (2,1)x_1 + (2,2)x_2 + (2,3)x_3, \\ x_3 = (3,1)x_1. \end{cases}$$

Решая ее, получим:

$$x_2 = (2,1)((1,2)x_2 + (1,3)x_3 + \lambda) + (2,2)x_2 + (2,3)x_3 =$$
$$= ((2,1)(1,2) + (2,2))x_2 + ((2,1)(1,3) + (2,3))x_3 + (2,1),$$

откуда

$$x_2 = ((2,1)(1,2) + (2,2))^*(((2,1)(1,3) + (2,3))x_3 + (2,1)).$$

Далее,

$$x_1 = (1,2)((2,1)(1,2)+(2,2))^*(((2,1)(1,3)+ \\ +(2,3))(3,1)x_1+(2,1))+(1,3)(3,1)x_1$$

и

$$x_1 = ((1,2)((2,1)(1,2)+(2,2))^*(((2,1)(1,3)+ \\ +(2,3))(3,1)+(1,3)(3,1))^*(1,2)(2,1).$$

Написанное выше регулярное выражение, полученное как значение переменного x_1, обозначает множество всех путей, начинающихся и заканчивающихся в первой вершине. Это множество можно словесно описать так: идем из первой вершины во вторую по дуге (1, 2), затем „крутимся“ сколько угодно раз (или ни разу) либо по контуру 2, 1, 2 либо по петле (2, 2) (как угодно их чередуя); далее идем в третью вершину через первую по пути 2, 1, 3 или сразу по дуге (2, 3), а из третьей вершины возвращаемся в первую — получается замкнутый путь, по которому можно „крутиться“, но не следует забывать, что можно „крутиться“ и по контуру 1, 3, 1. Кроме того, попав, как описано выше, во вторую вершину, можно сразу вернуться в первую, закончив „путешествие“.

7.37. Для натуральных чисел, заданных в виде слов в алфавите $\{0,|\}$, определить операцию умножения и построить машину Тьюринга над алфавитом $\{0,|\}$, которая для любых натуральных чисел m и n вычисляет их произведение.

7.38. Построить машину Тьюринга над алфавитом $\{a,b\}$, которая:

а) к любому слову $x \in \{a,b\}^*$ присоединяет слева (справа) заданное слово w_0;

б) распознает палиндромы в алфавите $\{a,b\}$;

в) распознает двойные слова в алфавите $\{a,b\}$;

г) удваивает произвольное слово $x \in \{a,b\}^*$, т.е. по этому слову вычисляет слово xx.

8. КОНТЕКСТНО-СВОБОДНЫЕ ЯЗЫКИ

8.1. КС-грамматики. Деревья вывода. Однозначность

Мы приступаем к изучению одного из важнейших классов формальных языков — класса *контекстно-свободных языков* (*КС-языков*).

Определение *КС-грамматик* и КС-языков было дано в **7.3**. Напомним, что *порождающую грамматику* $G = (V, N, S, P)$ называют контекстно-свободной (или КС-грамматикой), если каждое *правило вывода* из множества P имеет вид

$$A \to \gamma,$$

где $A \in N$ — *нетерминал*, а $\gamma \in (V \cup N)^*$ — *цепочка* в *объединенном алфавите грамматики*. Таким образом, неформально каждое правило вывода в КС-грамматике есть правило замены нетерминального символа цепочкой (возможно, *пустой*) в объединенном алфавите.

Каждому *выводу* в КС-грамматике, начинающемуся с нетерминального символа, однозначно сопоставляется *ориентированный граф*, являющийся *деревом* и называемый ***деревом вывода***. *Вершины* дерева вывода помечаются символами объединенного алфавита грамматики или пустой цепочкой. Подчеркнем, что дерево вывода сопоставляется не грамматике, а конкретному выводу в данной грамматике, хотя мы увидим, что нескольким различным выводам может быть сопоставлено одно и то же дерево вывода. Прежде чем давать математическое определение дерева вывода, покажем на примере, как по данному выводу в заданной КС-грамматике строится это дерево.

Пример 8.1. Рассмотрим КС-грамматику

$$G_0 = (\{a, b\},\ \{S, A, B, C\},\ S,\ P_0),$$

где множество правил вывода P_0 имеет вид:

$$\begin{aligned}
&S \to ABC, && (1)\\
&A \to BB, && (2)\\
&A \to \lambda, && (3)\\
&B \to CC, && (4)\\
&B \to a, && (5)\\
&C \to AA, && (6)\\
&C \to b. && (7)
\end{aligned}$$

В этой грамматике рассмотрим такой вывод:

$$\begin{aligned}
S \vdash_{(1)} ABC \vdash_{(2)} BBBC \vdash_{(6)}\\
\vdash_{(6)} BBBAA \vdash_{(3)} BBB\lambda A \vdash_{(4)} CCBBA \vdash^2_{(7)}\\
\vdash^2_{(7)} bbBBA \vdash^2_{(5)} bbaaA \vdash_{(2)} bbaaBB \vdash^2_{(5)} bbaaaa.
\end{aligned}$$

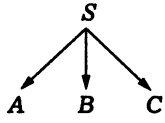


Рис. 8.1

По *шагам* этого *вывода* проследим процесс построения дерева вывода. Первому шагу соответствует дерево, изображенное на рис. 8.1.

Корень этого *дерева* (*куста*) помечен заменяемым на данном шаге нетерминалом, а метки *листьев*, перечисленные слева направо, образуют цепочку, полученную после применения правила (1) на первом шаге. Вообще же цепочку, полученную в результате такого перечисления меток листьев, называют ***кроной дерева****.

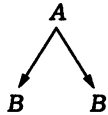


Рис. 8.2

На втором шаге применяется правило (2) и производится замена *вхождения* (единственного в данном случае) символа A цепочкой BB. Поэтому вершину дерева на рис. 8.1 с меткой A заменим кустом, как показано на рис. 8.2.

*Допуская некоторую вольность речи, кроной дерева называют и соответствующую *перестановку множества* листьев.

После двух шагов получим дерево, изображенное на рис. 8.3. Действуя аналогично, после третьего шага получим дерево, показанное на рис. 8.4.

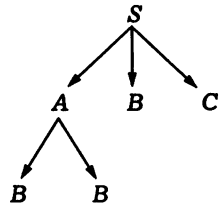


Рис. 8.3

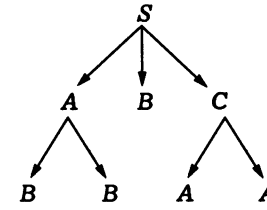


Рис. 8.4

Четвертый шаг требует более подробного комментария. Здесь в кроне дерева имеется несколько вхождений символа A. На четвертом шаге *первое вхождение* символа A в цепочку $BBAA$ заменяется пустой цепочкой. Поэтому очередной шаг в построении дерева будет состоять в том, что первый лист с меткой A в кроне дерева должен быть заменен кустом с корнем A и единственным листом с меткой λ. Тогда получим дерево, изображенное на рис. 8.5.

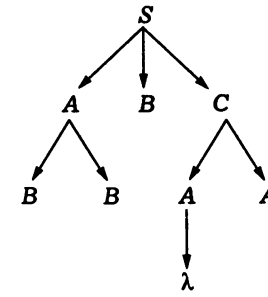


Рис. 8.5

Крона вновь полученного дерева — это цепочка $BBB\lambda A$, выведенная из аксиомы за четыре шага. Она, как элемент *свободного моноида* $(V \cup N)^*$, равна (в силу известных свойств пустой цепочки — см. **7.1**) цепочке $BBBA$. Обратим здесь внимание на то, что, хотя пустая цепочка и не является символом ни одного из алфавитов грамматики, она, как было уже сказано, может быть меткой вершины дерева вывода. Тем самым в дереве вывода фиксируются все вхождения выбрасываемых, т.е. заменяемых в процессе вывода пустой цепочкой, нетерминалов. Это равносильно выделению в выводимой цепочке некоторых вхождений в нее пустой цепочки.

На пятом шаге первое вхождение символа B в цепочку $BBBA$ заменяется цепочкой CC. Этому отвечает новое по-

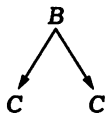


Рис. 8.6

строение на очередном дереве, состоящее в замене первого листа кроны с меткой B кустом, изображенным на рис. 8.6, после чего получим дерево, показанное на рис. 8.7.

Действуя дальше точно так же, получим в результате дерево вывода, изображенное на рис. 8.8. #

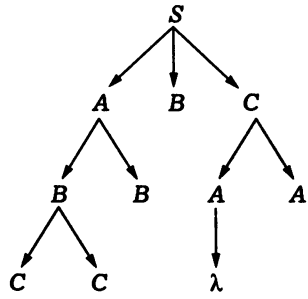


Рис. 8.7

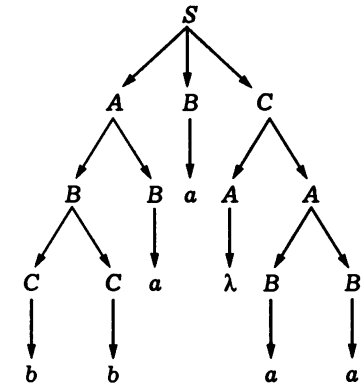


Рис. 8.8

Из разобранного примера следует, что шаги построения дерева вывода в точности соответствуют шагам самого вывода, причем после каждого шага получаем некоторое дерево (назовем его ***частичным деревом вывода***, полученным после данного шага). Начинаем построение с корня, и его меткой служит тот нетерминал, с которого начинается вывод (в частности, это может быть аксиома грамматики). Если на очередном шаге вывода применяется правило $B \to X_1 X_2 \ldots X_m$, где X_i $(1 \leqslant i \leqslant m)$ — либо символ объединенного алфавита, либо пустая цепочка, то тот лист частичного дерева вывода, полученного перед этим шагом, который имеет метку B и соответствует заменяемому вхождению символа B, заменяется кустом с корнем B и листьями $X_1, X_2, \ldots, X_m$.

Замечание 8.1. 1. Прежде всего нужно отчетливо понимать, что на каждом шаге построения дерева вывода „развер-

тывается" в куст не любая вершина, меткой которой служит некий нетерминал, а именно та вершина, которая соответствует заменяемому вхождению указанного нетерминала.

2. Совершенно необязательно рассматривать только деревья выводов терминальных цепочек из аксиомы. Дерево вывода может быть построено по выводу любой цепочки в объединенном алфавите из любого нетерминала грамматики. Так, мы могли бы на базе разобранного выше примера нарисовать дерево вывода цепочки CCa из нетерминала A, показанное на рис. 8.9.

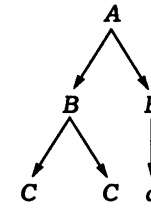


Рис. 8.9

3. Можно заметить, что разные выводы одной и той же цепочки из заданного нетерминала могут дать одно и то же дерево вывода. Так, дерево, изображенное на рис. 8.9, можно построить по двум различным выводам:

$$A \vdash BB \vdash CCB \vdash CCa$$

и

$$A \vdash BB \vdash Ba \vdash CCa. \quad \#$$

Выводы, имеющие одно и то же дерево вывода, естественно считать эквивалентными. Они различаются между собой только порядком, в котором применяются правила вывода*. Дерево вывода и служит способом наглядного изображения всех эквивалентных выводов. Кроме того, как мы увидим позже, дерево вывода является одним из основных инструментов доказательства утверждений о КС-языках и КС-грамматиках.

Теперь дадим формальное определение дерева вывода. Рассмотрим множество (ориентированных) деревьев, вершины которых помечены символами некоторого алфавита W. Ориентированное дерево, каждая вершина которого помечена символом некоторого алфавита, будем называть ***помеченным деревом.***

*Здесь, в рамках сугубо неформального изложения, мы не даем строгого определения эквивалентных выводов.

Если, кроме того, задано определенное правило перечисления меток вершин дерева, образующих некоторое подмножество вершин всего дерева, в частности листьев дерева, то такое помеченное дерево называют ***упорядоченным деревом***. Всюду в дальнейшем, изображая деревья выводов, условимся понимать их как упорядоченные деревья, метки листьев которых перечисляются всегда слева направо. Слева направо будем также перечислять и *сыновей* каждой *вершины*. Будем считать, что в машинном представлении деревьев выводов порядок „слева направо" при изображении дерева на плоскости соответствует порядку от первого элемента *списка смежности* вершины к последнему. Дерево, корень которого помечен символом A, а листья имеют метки X_1, X_2, ..., X_m, договоримся записывать в виде цепочки

$$A \downarrow\downarrow X_1X_2 \ldots X_m$$

(с двумя стрелками).

Если рассматриваемое дерево является кустом, то условимся писать

$$A \downarrow X_1X_2 \ldots X_m$$

(с одной стрелкой).

Листья дерева, обозначенного таким образом, отождествим с символами цепочки X_1, X_2, ..., X_m. Это значит, что, говоря о листе X_i, мы имеем в виду лист помеченного дерева, меткой которого является символ X_i и который в перечислении листьев слева направо получает номер i. Тем самым устанавливается взаимно однозначное соответствие между листьями помеченного дерева и вхождениями символов алфавита W в цепочку X_1, X_2, ..., X_m. Например, записывая дерево в виде $A \downarrow\downarrow BBCBC$, мы можем говорить о листе B, соответствующем первому вхождению символа B в цепочку $BBCBC$, или же, обозначив $BBCBC$ через α, о листе $\alpha(1)$. Тогда листья $\alpha(1)$, $\alpha(2)$ и $\alpha(4)$ — это разные листья дерева, хотя их меткой является один и тот же символ алфавита W.

Расширим множество помеченных деревьев, допуская в качестве меток их вершин и пустую цепочку. В этом случае, в записи $A \downdownarrows X_1X_2\ldots X_m$, представляющей дерево с корнем A и листьями X_1, X_2, ..., X_m, среди меток листьев, т.е. элементов X_i, $i = \overline{1,m}$, могут быть и пустые цепочки. Тогда, говоря о листе с меткой λ, нужно указывать, какое вхождение пустой цепочки λ в цепочку $X_1X_2\ldots X_m$ имеется в виду.

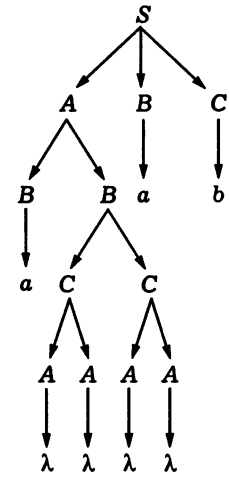


Рис. 8.10

Пример 8.2. а. Дерево вывода, изображенное на рис. 8.8, может быть обозначено так: $S \downdownarrows bbaa\lambda aa$. В нем лист с меткой λ соответствует вхождению ($bbaa$, λ, aa) пустой цепочки в цепочку $bbaaaa$.

б. Для грамматики из примера 8.1 вывод

$$S \vdash ABC \vdash BBBC \vdash aBBC \vdash aCCBC \vdash aAACBC \vdash$$
$$\vdash a\lambda ACBC \vdash a\lambda\lambda CBC \vdash a\lambda\lambda AABC \vdash a\lambda\lambda\lambda ABC \vdash$$
$$\vdash a\lambda\lambda\lambda\lambda BC \vdash a\lambda\lambda\lambda\lambda aC \vdash a\lambda\lambda\lambda\lambda ab = aab$$

имеет дерево, показанное на рис. 8.10. В этом дереве любой его лист с меткой λ соответствует одному и тому же вхождению пустой цепочки в выводимую цепочку aab, а именно вхождению (a, λ, ab), так как для любого натурального n выполняется равенство $\lambda^n = \lambda$.

в. Представленное на рис. 8.11 дерево вывода цепочки $aabaa$ может быть задано записью

$$S \downdownarrows aa\lambda\lambda b\lambda aa.$$

Оно имеет три листа с меткой λ.

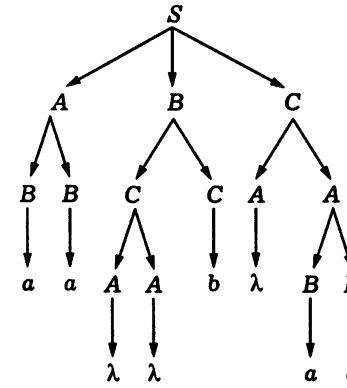


Рис. 8.11

Первые два соответствуют вхождению (aa, λ, baa), а третье — вхождению (aab, λ, aa) пустой цепочки в выводимую цепочку. #

Помеченное дерево $A \downdownarrows X_1X_2\ldots X_m$ называют ***λ-свободным***, если среди меток X_1, X_2, ..., X_m его листьев нет пустой цепочки. Из предыдущих примеров можно заключить, что не всякая цепочка, выводимая в КС-грамматике из какого-либо нетерминала, имеет λ-свободное дерево вывода.

Помеченное дерево, полученное из дерева

$$A \downdownarrows X_1\ldots X_{i-1}X_iX_{i+1}\ldots X_m$$

заменой листа $X_i \neq \lambda$ деревом $X_i \downdownarrows Y_1\ldots Y_k$, будем записывать в виде цепочки $A \downdownarrows X_1\ldots X_{i-1}[X_i \downdownarrows Y_1\ldots Y_k]X_{i+1}\ldots X_m$. Это дерево, поскольку в нем вместо листа X_i, возникают листья с метками $Y_1,\ldots,Y_k$, можно представить также в виде

$$A \downdownarrows X_1\ldots X_{i-1}Y_1\ldots Y_kX_{i+1}\ldots X_m.$$

Дерево вывода цепочки β в объединенном алфавите из нетерминала A ***в КС-грамматике*** $G = (V, N, S, P)$ — это помеченное дерево, метками вершин которого являются символы объединенного алфавита $V \cup N$ или пустая цепочка λ и которое строится по следующим правилам:

1) дерево любого вывода длины 0 состоит из единственной вершины, помеченной нетерминалом A (в данном случае цепочка $\beta = A$);

2) дерево вывода $A \vdash \lambda$ — это дерево $A \downarrow \lambda$ (в данном случае цепочка $\beta = \lambda$);

3) дерево вывода $A \vdash \alpha$, где α — непустая цепочка, есть дерево $A \downarrow \alpha$;

4) если α — цепочка в объединенном алфавите и

$$A \downdownarrows X_1X_2\ldots X_m \quad —$$

дерево некоторого вывода $\mathbf{D}$ этой цепочки из нетерминала A, где $\alpha = X_1X_2\ldots X_m$ и для каждого $i = \overline{1,m}$ $X_i \in V \cup N \cup \{\lambda\}$, а

для некоторого i $(1 \leqslant i \leqslant m)$ $X_i = B \in N$ — нетерминальный символ грамматики G, и в множестве правил вывода P грамматики G есть правило $B \to Y_1 \dots Y_k$, то дерево

$$A \downdownarrows X_1 \dots X_{i-1}[X_i \downarrow Y_1 \dots Y_k]Y_{i+1} \dots Y_m$$

есть дерево такого вывода цепочки

$$\beta = X_1 \dots X_{i-1}Y_1 \dots Y_k X_{i+1} \dots X_m$$

из нетерминала A, что его последний шаг имеет вид

$$X_1 \dots X_{i-1}BX_{i+1} \dots X_m \vdash$$
$$\vdash_{B \to Y_1 \dots Y_k} X_1 \dots X_{i-1}Y_1 \dots Y_k X_{i+1} \dots X_m,$$

а вывод цепочки $\alpha = X_1 \dots X_{i-1}BX_{i+1} \dots X_m$ из A совпадает с $\mathbf{D}$.

Все выводы данной цепочки из данного нетерминала, по которым приведенный выше алгоритм дает одно и то же дерево вывода, будем называть ***эквивалентными***. Среди эквивалентных выводов выделим два — ***левый вывод***, в котором на каждом шаге заменяется *самое левое вхождение* нетерминального символа; и ***правый вывод***, когда на каждом шаге, напротив, производится замена *самого правого вхождения* нетерминала.

Это значит, что при построении дерева вывода по левому выводу данной цепочки α из данного нетерминала A каждый раз „развертывается“ в куст самый левый лист кроны частичного дерева вывода, помеченный нетерминальным символом, а при построении дерева вывода по правому выводу то же совершается с самым правым листом кроны.

Для построенного выше в примере 8.2.а дерева вывода левый вывод имеет вид

$$S \vdash \boldsymbol{A}BC \vdash \boldsymbol{BB}BC \vdash \boldsymbol{CC}BBC \vdash$$
$$\vdash b\boldsymbol{C}BBC \vdash bb\boldsymbol{B}BC \vdash bba\boldsymbol{B}C \vdash bbaa\boldsymbol{C} \vdash$$
$$\vdash bbaa\boldsymbol{AA} \vdash bbaa\lambda\boldsymbol{A} \vdash bbaa\boldsymbol{BB} \vdash bbaaa\boldsymbol{B} \vdash bbaaaa.$$

Правый вывод той же цепочки имеет вид

$$S \vdash ABC \vdash ABAA \vdash ABABB \vdash ABABa \vdash$$
$$\vdash ABAaa \vdash AB\lambda aa \vdash Aaaa \vdash BBaaa \vdash Baaaa \vdash$$
$$\vdash CCaaaa \vdash Cbaaaa \vdash bbaaaa.$$

В обоих выводах полужирным шрифтом выделены заменяемые вхождения нетерминалов, а также удалено возникающее после применения правила $A \to \lambda$ вхождение пустой цепочки λ. Можно показать, что эквивалентные выводы различаются между собой только порядком применения правил. Сами же применяемые правила и вхождения заменяемых нетерминалов у эквивалентных выводов одинаковы.

Мы часто будем использовать условное графическое обозначение дерева вывода $A \downarrow\downarrow X_1X_2\ldots X_m$ в виде „треугольника“, одна из вершин которого помечена нетерминалом A, а противолежащая этой вершине сторона символизирует всю цепочку $\alpha = X_1X_2\ldots X_m$. Иногда будем точками (или штрихами) отмечать отдельные символы цепочки α или листья с меткой λ (рис. 8.12).

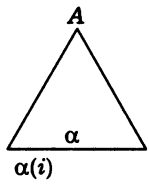


Рис. 8.12

Кроме того, по традиции деревья выводов изображаются без стрелок на дугах (хотя понимаются как ориентированные деревья).

Из определения дерева вывода понятно, что в нем *дуга* из вершины с меткой X ведет в вершину с меткой Y тогда и только тогда, когда на некотором шаге вывода применяется правило вывода $X \to \gamma_1 Y \gamma_2$, где γ_1 и γ_2 — цепочки в объединенном алфавите (возможно, пустые). Следовательно, число дуг любого *пути ненулевой длины* дерева вывода, т.е. длина этого пути, есть не что иное, как число применений правил вывода заданной КС-грамматики в некотором *фрагменте вывода*. А так как в КС-грамматике каждое применение правила вывода есть замена вхождения нетерминала некоторой цепочкой в объединен-

ном алфавите, то длина пути в дереве вывода равна числу замен нетерминалов в соответствующем фрагменте вывода. Например, для дерева, показанного на рис. 8.8, пути $S \to A \to B \to a$ отвечает фрагмент вывода $S \vdash ABC \vdash BBBC \vdash BaBC$. Обратим внимание на то, что приведенный фрагмент есть фрагмент одного из множества эквивалентных выводов, имеющих данное дерево.

Таким образом, по заданному пути в дереве вывода однозначно восстанавливается фрагмент одного из множества эквивалентных выводов. Более того, по дереву вывода, построенному по одному из множества эквивалентных выводов, можно восстановить все выводы этого множества, в частности левый и правый. Можно показать, что для этого необходимо задать определенный порядок посещения вершин дерева при *поиске в глубину* от корня. Левый (соответственно правый) вывод будет восстановлен, если каждый раз при поиске в глубину от очередной вершины задавать продолжение поиска от самого левого (соответственно самого правого) сына этой вершины.

Замечание 8.2. *Длина вывода* в общем случае не меньше, чем высота дерева этого вывода. Можно доказать, что для фиксированной КС-грамматики G существует такая константа C_G, зависящая от G, что для любого вывода $\mathbf{D}$ (начинающегося каким-либо нетерминалом) и дерева $\mathbf{T}$ этого вывода разность между длиной $l_{\mathbf{D}}$ вывода $\mathbf{D}$ и высотой $h_{\mathbf{T}}$ дерева $\mathbf{T}$ не превышает C_G: $l_{\mathbf{D}} - h_{\mathbf{T}} \leqslant C_G$. Другими словами, разность между длиной вывода и высотой дерева вывода для фиксированной грамматики ограничена сверху. Это обусловлено тем, что указанная разность определяется числом нетерминалов в правых частях правил вывода, которое в пределах заданной грамматики всегда ограничено. Это число, говоря неформально, определяет, как сильно „ветвится“ дерево вывода в процессе его „роста“. Для *линейной грамматики*, как нетрудно сообразить, высота дерева вывода равна длине вывода. Однако если не фиксировать грамматику, то разность между длиной вывода и высотой

дерева вывода может быть сколь угодно большой. Это можно подтвердить таким простым примером.

Рассмотрим семейство КС-грамматик

$$G_m = (V_m,\ N_m,\ S,\ P_m),$$

где $m \geqslant 1$, $V_m = \{a_1, \ldots, a_m\}$, $N_m = \{S, B_1, \ldots, B_m\}$, а множество правил вывода P_m имеет вид

$$S \to B_1 \ldots B_m, \quad B_1 \to a_1, \ldots, B_m \to a_m.$$

Грамматика G_m порождает единственную терминальную цепочку $a_1 \ldots a_m$. Высота дерева вывода этой цепочки равна 2 и не зависит от m (рис. 8.13), тогда как длина вывода определяется числом $m+1$. #

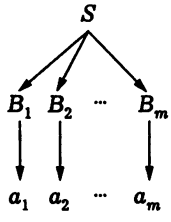


Рис. 8.13

Пусть фиксировано ориентированное дерево **T** с корнем R. Выделим в нем произвольно вершину Q, не являющуюся ни листом, ни корнем, называемую ***внутренней вершиной дерева***. Образуем *поддерево* дерева **T**, объявив его корнем вершину Q. Такое поддерево назовем ***максимальным поддеревом***, если оно содержит все вершины исходного дерева **T**, *достижимые* из вершины Q. Ясно, что *высота* любого максимального поддерева больше нуля и меньше высоты дерева **T**. Для помеченного дерева **T** нетрудно проверить справедливость следующей теоремы.

Теорема 8.1. Крона любого максимального поддерева помеченного дерева есть подцепочка кроны всего дерева.

Замечание 8.3. Сформулированное утверждение достаточно прозрачно и означает, что если в помеченном дереве **T** фиксировать произвольно внутреннюю вершину Q и рассмотреть крону максимального поддерева с корнем Q, то все листья кроны этого поддерева, расположенные между самым

левым (L) и самым правым (L') листьями, достижимыми из Q, также достижимы из Q (рис. 8.14). #

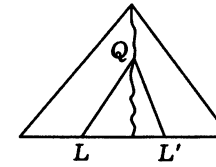


Рис. 8.14

Применяя теорему 8.1 к деревьям выводов в КС-грамматиках, получим следующий результат.

Следствие 8.1. Если в дереве вывода некоторой цепочки α из нетерминала A фиксировать произвольно внутреннюю вершину, помеченную нетерминалом B, то максимальное поддерево с корнем в фиксированной вершине (с меткой B) является деревом вывода некоторой подцепочки β цепочки α из нетерминала B, т.е. существуют такие цепочки γ_1 и γ_2, что $\alpha = \gamma_1\beta\gamma_2$ и $A \vdash^* \gamma_1 B \gamma_2 \vdash^* \gamma_1\beta\gamma_2$ (рис. 8.15). #

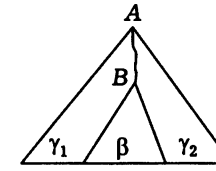


Рис. 8.15

Так, для дерева вывода цепочки $bAAa$ из нетерминала A в грамматике G_0 из примера 8.1 (рис. 8.16), фиксируя максимальное поддерево с корневой вершиной C (вторая вершина слева на первом уровне, выделена полужирным шрифтом), получим $\gamma_1 = b$, $\beta = AA$, $\gamma_2 = a$. Соответствующее этому максимальному поддереву разбиение вывода на два фрагмента, может быть, например, таким: $A \vdash BB \vdash CCB \vdash \vdash bCB \vdash bCa$ (первый фрагмент) и $C \vdash AA$ (второй фрагмент).

Описанная в следствии 8.1 „декомпозиция“ дерева вывода и соответственно вывода, по которому построено это дерево, может быть интерпретирована так: выводя цепочку α из нетерминала A, мы, получив на некотором шаге цепочку, содержащую выделенное вхождение нетерминала B, „замораживаем“ этот нетерминал и продолжаем замены нетерминалов слева и справа от B, выводя начало (подцепочку

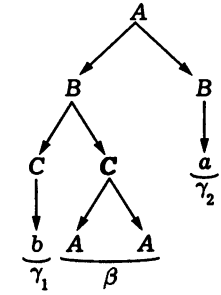


Рис. 8.16

γ_1) и конец (подцепочку γ_2) цепочки α. После этого мы „размораживаем“ нетерминал B и „довыводим“ из него середину цепочки α — подцепочку β.

Понятие дерева вывода связано с понятием однозначности КС-грамматики.

Определение 8.1. КС-грамматику называют ***однозначной***, если каждая цепочка порождаемого ею языка имеет единственное дерево вывода.

Поскольку по дереву вывода цепочки α из нетерминала A однозначно строится как левый, так и правый вывод данной цепочки, то определение 8.1 можно сформулировать иначе: КС-грамматика однозначна, если каждая цепочка порождаемого ею языка имеет единственный левый и единственный правый вывод.

Рассмотрим один хрестоматийный пример неоднозначной грамматики.

Пример 8.3. Зададим грамматику GE следующим образом:

$$GE = (\{a, b, \textbf{if}, \textbf{then}, \textbf{else}\}, \{S\}, S, \\ S \to \textbf{if}\ b\ \textbf{then}\ S\ \textbf{else}\ S \mid \textbf{if}\ b\ \textbf{then}\ S \mid a).$$

Эта грамматика описывает „абстрактный синтаксис“ операторов условного перехода в некотором „паскалеобразном“ языке программирования. Термин „абстрактный синтаксис“ в данном случае означает, что мы отвлекаемся от структуры как операторов языка, так и логических условий.

Терминальный символ a означает произвольный оператор (или последовательность операторов), а терминальный символ b — призвольное логическое условие. Кроме того, терминалами грамматики являются также лексемы **if**, **then** и **else**, формирующие „скелет“ любого оператора условного перехода.

Определенная таким образом грамматика неоднозначна, что усматривается из такого примера: цепочка

$$\textbf{if } b \textbf{ then if } b \textbf{ then } a \textbf{ else } a$$

имеет два дерева вывода (рис. 8.17 и 8.18).

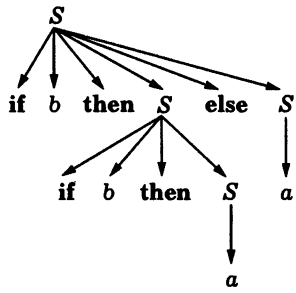


Рис. 8.17

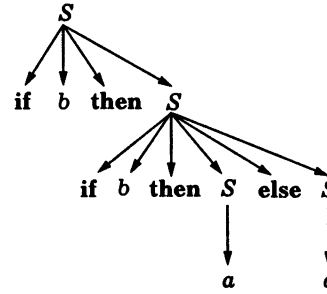


Рис. 8.18

Заметим (неформально), что наличие нескольких деревьев вывода одной и той же цепочки имеет отношение к *семантике* рассматриваемого языка. В данном конкретном примере понятно, что каждому дереву вывода приведенной выше цепочки отвечает свое смысловое прочтение этой цепочки: в первом случае **else** мы относим к первому **then**, а во втором — ко второму. В итоге получаем две совершенно разные интерпретации оператора условного перехода. Данный пример неоднозначности известен поэтому под названием „кочующее **else**“.

Исходную грамматику, однако, можно „исправить“, построив эквивалентную ей однозначную грамматику:

$$GE_1 = (\{a, b, \textbf{if}, \textbf{then}, \textbf{else}\}, \{S_1, S_2\}, S_1,$$
$$S_1 \to \textbf{if } b \textbf{ then } S_2 \textbf{ else } S_1 \,|\, \textbf{if } b \textbf{ then } S_1 \,|\, a,$$
$$S_2 \to \textbf{if } b \textbf{ then } S_2 \textbf{ else } S_2 \,|\, a).$$

В общем случае алгоритма преобразования произвольной неоднозначной КС-грамматики к эквивалентной однозначной не существует. #

Рассмотрим в заключение интересный пример неоднозначности в естественном языке (точнее, в художественном тексте). В известном стихотворении А.С. Пушкина „К вельможе“, адресованном князю Юсупову, владельцу Архангельского, есть такая строка:

Посланник молодой увенчанной жены...

(Речь идет о том, что князь был послан Екатериной Великой к Вольтеру, великому французскому философу и писателю, состоявшему с Екатериной в переписке.)

Процитированную строчку можно прочитать двояко:

1) посланник (молодой увенчанной жены),

2) (посланник молодой) увенчанной жены,

т.е. в первом случае эпитет „молодой“ отнесен к „жене“, а во втором — к „посланнику“. Чисто синтаксически эта коллизия неразрешима, но внеязыковый контекст данной фразы, учитывающий совершенно определенные биографические обстоятельства персонажей, позволяет с высокой долей вероятности предположить, что правильным будет второе прочтение.

В то же время следующую анекдотическую фразу нельзя рассматривать как пример неоднозначности, ибо она не соответствует грамматическим нормам русского литературного языка:

Сдавайте мусор дворнику, который накопился.

Приведенные примеры, между прочим, обнаруживают, что понятие однозначности необходимо анализировать в связи с семантикой языка и что „смысл“ следует придавать не цепочке языка, а дереву ее вывода (см. **Д.8.2**).

8.2. Приведенная форма КС-грамматики

При изучении свойств КС-грамматик и КС-языков иногда бывает целесообразно преобразовать КС-грамматику к эквивалентной КС-грамматике, правила вывода которой удовлетворяют определенным дополнительным ограничениям. В этом

разделе мы рассмотрим алгоритмы некоторых таких преобразований.

Определение 8.2. *Правило вывода КС-грамматики* $G = (V, N, S, P)$ называют:

1) ***λ-правилом***, если его *правая часть — пустая цепочка*;

2) ***цепным***, если оно имеет вид $A \to B$, где A и B — *нетерминалы* грамматики.

Теорема 8.2. Любая КС-грамматика $G = (V, N, S, P)$ может быть преобразована к эквивалентной КС-грамматике, множество правил вывода которой не содержит λ-правил, кроме, может быть, правила $S \to \lambda$, и не содержит цепных правил. #

Мы опишем алгоритмы удаления λ-правил и цепных правил из множества правил КС-грамматики, опуская строгое обоснование этих алгоритмов.

Алгоритм удаления λ-правил. Предположим, что нам известно множество N_e всех нетерминалов грамматики G, из которых выводится пустая цепочка, т.е.

$$N_e = \{A\colon A \vdash^* \lambda\}.$$

Тогда в правой части каждого правила $A \to \xi$ выделяются все *вхождения* нетерминалов из множества N_e (если таких вхождений нет, то правило пропускается) так, что цепочка ξ представляется в виде

$$\xi = \alpha_1 B_1 \alpha_2 B_2 \ldots \alpha_m B_m \alpha_{m+1},$$

где B_1, B_2, ..., B_m — все нетерминалы из N_e, входящие в ξ. Заметим, что они не обязаны быть различными, так как мы выделяем все вхождения таких нетерминалов в правую часть правила.

После этого к рассматриваемому правилу добавляются все правила вида

$$A \to \alpha_1 X_1 \alpha_2 X_2 \ldots \alpha_m X_m \alpha_{m+1},$$

где для каждого $i = \overline{1,m}$ X_i есть или символ B_i, или пустая цепочка.

Иначе говоря, к любому правилу исходной грамматики, правая часть которого содержит нетерминалы из множества N_e добавляются все правила, которые могут быть получены из исходного путем замены (в его правой части) любого числа вхождений нетерминалов из N_e пустой цепочкой. При этом сохраняется и само исходное правило и появляется, в частности, правило

$$A \to \alpha_1\alpha_2 \ldots \alpha_m\alpha_{m+1},$$

вовсе не содержащее вхождений указанных нетерминалов в правой части. Подчеркнем еще раз, что каждое правило исходной грамматики, правая часть которого не содержит вхождений нетерминалов из N_e, без всяких изменений переносится в систему правил новой грамматики.

Если в процессе описанного выше преобразования системы правил исходной грамматики получаются правила вида $A \to A$ или $A \to \lambda$ (при $A \neq S$), то они игнорируются.

Остается обсудить метод вычисления множества N_e. Это множество вычисляется рекуррентно, а именно на первом шаге вычисляют множество N_0 таких нетерминалов A грамматики G, что существует в P λ-правило $A \to \lambda$. Таким образом, множество N_0 — это множество всех нетерминалов грамматики, из которых пустая цепочка выводится за один шаг.

На следующем шаге вычисляем множество N_1, включая в него, во-первых, все нетерминалы множества N_0 и, во-вторых, все такие нетерминалы B, что в P имеется правило вывода $B \to \gamma$, где γ — непустая цепочка нетерминалов из множества N_0, т.е. $\gamma \in \in N_0^+$. Это означает, что N_1 состоит из всех таких нетерминалов A грамматики G, что дерево вывода пустой цепочки из A, имеет высоту, не большую 2 (рис. 8.19).

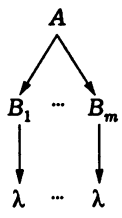


Рис. 8.19

В общем случае множество $N_i \subseteq N$ вычисляют по формуле

$$N_0 = \{A\colon \text{ в } P \text{ есть правило } A \to \lambda\},$$
$$N_i = N_{i-1} \cup \{B\colon \text{ в } P \text{ есть правило } B \to \xi, \text{ где } \xi \in N_{i-1}^{+}\}.$$

Множество N_i — это множество всех таких нетерминалов A грамматики G, что дерево вывода пустой цепочки λ из A имеет высоту, не большую $i+1$.

Так как множество N всех нетерминалов грамматики G конечно, то описанный выше рекуррентый процесс оборвется на некотором шаге, т.е. для некоторого $j \geqslant 0$ получим $N_j = N_{j+1}$. Тогда полагаем

$$\mathbb{N}_e = N_j \text{ для наименьшего } j, \text{ такого, что } N_j = N_{j+1}.$$

Далее, если аксиома S принадлежит множеству N_e, то в системе правил P оставляют правило $S \to \lambda$, остальные λ-правила удаляют*, а множество правил вывода исходной грамматики преобразуют согласно описанному выше алгоритму.

Можно доказать, что полученная таким образом КС-грамматика без λ-правил эквивалентна исходной грамматике G. Это объясняется тем, что „потерю“ вывода пустой цепочки из нетерминала $A \in N_e$ (посредством применения λ-правил) мы „компенсируем“ добавлением к множеству правил вывода всех таких правил, что замена A на λ уже произведена прямо в их правых частях.

Пример 8.4. Рассмотрим грамматику с множеством правил

$$S \to aSbT \,|\, bTaT \,|\, ab, \quad T \to baaST \,|\, TT \,|\, \lambda.$$

В данном случае $N_0 = \{T\}$. Так как единственное правило вывода этой грамматики, правая часть которого — непустая

*Исключение для λ-правила, левой частью которого служит аксиома грамматики, связано с тем, чтобы при переходе к новой грамматике не „потерять“ пустую цепочку, которую порождает исходная грамматика.

цепочка нетерминалов из N_0, есть правило $T \to TT$, то

$$N_1 = N_0 \cup \{T\} = \{T\} = N_0,$$

т.е. $N_e = \{T\}$, и наша грамматика принимает вид

$$S \to aSbT \,|\, aSb \,|\, bTaT \,|\, bTa \,|\, baT \,|\, ba \,|\, ab,$$
$$T \to baaST \,|\, baaS \,|\, TT. \quad \#$$

Алгоритм удаления цепных правил. Этот алгоритм во многом аналогичен алгоритму удаления λ-правил.

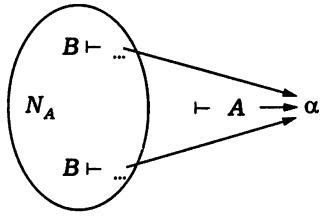


Рис. 8.20

Пусть существует вывод нетерминала U из нетерминала T, в котором применяются только цепные правила. Тогда такой вывод назовем ***цепным выводом***.

Предположим теперь, что для каждого нетерминала A грамматики G известно множество N_A всех таких нетерминалов B, что существует цепной вывод A из B. Тогда, удалив все цепные правила, мы должны предусмотреть непосредственную замену всех нетерминалов из N_A любой цепочкой α, которая служит правой частью некоторого правила $A \to \alpha$ (рис. 8.20).

Это соображение лежит в основе алгоритма преобразования множества правил КС-грамматики с целью удаления цепных правил.

Вычислив множество N_A для каждого $A \in N$, удалить все цепные правила, добавив при этом для каждой цепочки α, такой, что в P есть правило $A \to \alpha$, и для каждого $B \in N_A$ правило $B \to \alpha$. После этого вместо каждого вывода вида $\underbrace{B \vdash \ldots \vdash A}_{\text{цепной вывод}} \vdash \alpha$ (при $B \in N_A$) в исходной грамматике в новой грамматике получим вывод $B \vdash \alpha$.

Множества нетерминалов N_A для каждого $A \in N$ вычисляются рекуррентно (как и в рассмотренном выше алгоритме

удаления λ-правил вычислялось множество N_e):

$$N_A^0 = \{B\colon \text{ в } P \text{ есть правило } B \to A\};$$
$$N_A^i = N_A^{i-1} \cup \{C\colon \text{ в } P \text{ есть правило } C \to D, \text{ где } D \in N_A^{i-1}\}.$$

Таким образом, множество N_A^i — это множество всех таких нетерминалов B, что существует цепной вывод длины не больше $i+1$ нетерминала A из B. „Насыщение“ этой рекуррентной процедуры происходит для первого номера j, такого, что $N_A^j = N_A^{j+1}$, и тогда принимают $N_A = N_A^j$.

Пример 8.5. Дана грамматика с множеством правил

$$E \to E+T \,|\, T,$$
$$T \to T*F \,|\, F,$$
$$F \to (E) \,|\, a,$$

аксиомой E и терминальным алфавитом $\{a, (,), +, *\}$. Имеем $N_E^0 = \varnothing$, следовательно, и $N_E = \varnothing$ (множество нетерминалов данной грамматики, из которых применением одних цепных правил выводится нетерминал E, пусто). Далее, $N_T^0 = \{E\}$, а так как нет ни одного цепного правила с правой частью E, то $N_T^1 = N_T^0 = N_T$; $N_F^0 = \{T\}$; $N_F^1 = \{T, E\} = N_F$.

Удаляя цепные правила, необходимо, согласно описанному выше алгоритму, ко всем правилам с левой частью E добавить правило $E \to T*F$, чтобы, сокращая вывод $E \vdash T \vdash T*F$ и убирая из него применение цепного правила $E \to T$, сразу вывести из E цепочку $T*F$.

Аналогично к правилам с левой частью F надо добавить правила $T \to (E)$, $T \to a$, $E \to (E)$ и $E \to a$. В итоге получим грамматику

$$E \to E+T \,|\, T*F \,|\, (E) \,|\, a,$$
$$T \to T*F \,|\, a \,|\, (E),$$
$$F \to (E) \,|\, a. \quad \#$$

Можно доказать, что описанный алгоритм удаления цепных правил дает грамматику, эквивалентную исходной.

Замечание 8.4. К цепным правилам относится и правило $A \to A$. Если такое правило находится в множестве правил грамматики или появляется при удалении λ-правил (см. пример 8.4), то оно должно быть просто удалено из множества правил. #

Обратимся к примеру, из которого станет ясно, что удаление λ-правил может привести к появлению в множестве правил заданной КС-грамматики цепных правил.

Пример 8.6. Рассмотрим грамматику G_0 из примера 8.1. Для удобства перепишем ее определение:

$$G_0 = (\{a, b\},\ \{S, A, B, C\},\ S,\ P_0),$$

где множество правил вывода P_0 имеет вид

$$\begin{array}{ll} S \to ABC, & (1) \\ A \to BB, & (2) \\ A \to \lambda, & (3) \\ B \to CC, & (4) \\ B \to a, & (5) \\ C \to AA, & (6) \\ C \to b. & (7) \end{array}$$

Удаляем λ-правила. Множество $N_0 = \{A\}$ (правило (3)). Далее, $N_1 = \{A, C\}$, поскольку в правиле (6) правая часть есть непустая цепочка нетерминалов из N_0; $N_2 = \{A, C, B\}$ (правило (4)), $N_3 = \{A, C, B, S\}$. Так как N_3 совпало с множеством всех нетерминалов грамматики, то $N_3 = N_e$.

Согласно описанному выше алгоритму удаления λ-правил, получим следующую грамматику:

$$\begin{array}{l} S \to ABC \,|\, AB \,|\, AC \,|\, BC \,|\, A \,|\, B \,|\, C \,|\, \lambda, \\ A \to BB \,|\, B, \\ B \to CC \,|\, C \,|\, a, \\ C \to AA \,|\, A \,|\, b. \end{array}$$

Мы видим, что в результате удаления λ-правил получилось много цепных правил (которых не было в исходной грамматике).

Применение алгоритма удаления цепных правил даст следующие множества N_A^i:

$$N_S^0 = N_S^e = \varnothing;$$
$$N_A^0 = \{S, C\}, \quad N_A^1 = \{S, C, B\}, \quad N_A^2 = \{S, C, B, A\};$$
$$N_B^0 = \{S, A\}, \quad N_B^1 = \{S, A, C\}, \quad N_B^2 = \{S, A, C, B\};$$
$$N_C^0 = \{S, A\}, \quad N_C^1 = \{S, A, B\}, \quad N_A^2 = \{S, A, B, C\}.$$

Таким образом, $N_A^e = N_B^e = N_C^e = \{S, A, B, C\}$ (множество всех нетерминалов грамматики).

Новое множество правил, не содержащее цепных правил, будет иметь следующий вид:

$$S \to ABC|AB|AC|BC|\lambda|BB|CC|AA|a|b;$$
$$A \to BB|CC|AA|a|b;$$
$$B \to CC|BB|AA|a|b;$$
$$C \to AA|BB|CC|a|b.$$

Замечание 8.5. Теорема 8.2 позволяет утверждать, что любую КС-грамматику можно преобразовать к эквивалентной КС-грамматике, множество правил вывода которой содержит только *КЗ-правила*, т.е. правила вида $\alpha A\beta \to \alpha\gamma\beta$ при $\alpha = \beta = \lambda$ и $\gamma \neq \lambda$. Единственным исключением может быть правило $S \to \lambda$ для аксиомы S. Можно показать, что добавление к множеству правил *КЗ-грамматики* правила, предусматривающего замену аксиомы пустой цепочкой, не приводит к существенному изменению класса языков, порождаемых КЗ-грамматиками. Единственное отличие языка, порождаемого такой грамматикой, от *КЗ-языка* состоит в том, что он может содержать пустую цепочку, тогда как ни один КЗ-язык, будучи *неукорачивающим языком*, пустой цепочки не содержит.

Следовательно, если в множестве правил КЗ-грамматики разрешить кроме КЗ-правил также и правило $S \to \lambda$ (где S — аксиома грамматики), то любую КС-грамматику можно преобразовать к эквивалентной КС-грамматике, являющейся частным случаем КЗ-грамматики (с правилом $S \to \lambda$).

Определение 8.3. Нетерминальный символ A КС-грамматики $G = (V, N, S, P)$ называют ***бесполезным***, если из него не выводится ни одна терминальная цепочка, т.е. не существует такого $x \in V^*$, что $A \vdash_G^* x$.

Бесполезный нетерминал КС-грамматики может быть удален (вместе со всеми правилами, в которые он входит) без изменения языка, порождаемого данной грамматикой. Следующая теорема утверждает существование алгоритма удаления бесполезных нетерминальных символов.

Теорема 8.3. Для каждой КС-грамматики $G = (V, N, S, P)$ может быть построена эквивалентная ей КС-грамматика, не содержащая бесполезных нетерминальных символов.

◀ Алгоритм удаления бесполезных нетерминалов состоит в рекуррентном вычислении множества $N_u \subseteq N$ всех нетерминалов грамматики G, из которых выводится какая-либо терминальная цепочка.

Сначала вычисляем множество N_0 всех нетерминалов, из которых терминальная цепочка выводится за один шаг. Для этого просматриваем множество правил P, отмечая в нем все правила вида $A \to x$, где $x \in V^*$. Другими словами,

$$N_0 = \{A\text{: в } P \text{ есть правило } A \to x,\ x \in V^*\}.$$

Затем вычисляем множество N_1, добавляя к N_0 множество всех таких нетерминалов B, что существует правило вывода в P вида $B \to \beta$, где β — цепочка, содержащая как терминалы, так и нетерминалы из множества N_0. Таким образом, в N_1 попадут все нетерминалы B, такие, что существует вывод

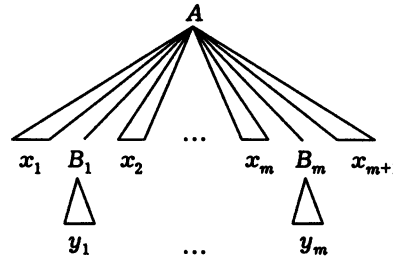


Рис. 8.21

терминальной цепочки из B, и высота дерева этого вывода не больше 2 (рис. 8.21).

В общем случае для множества N_i ($i \geqslant 1$) получаем формулу

$$N_i = N_{i-1} \cup \{A\text{: в } P \text{ есть правило } A \to \alpha, \text{ где } \alpha \in (N_{i-1} \cup V)^*\}.$$

Это множество содержит все такие нетерминалы B, что высота дерева вывода любой терминальной цепочки из B не превышает $i+1$.

Для наименьшего j, такого, что $N_j = N_{j+1}$, полагаем $N_u = N_j$. Из построения множества N_u следует, что это множество всех таких нетерминалов A грамматики G, что существует вывод (ненулевой длины) $A \vdash^+ x$ для некоторого $x \in V^*$, что и требовалось доказать. Тогда все нетерминалы из $N \setminus N_u$ бесполезны. ▶

В частности, если $S \notin N_u$, то $L(G) = \varnothing$, и наоборот. Таким образом решается ***проблема пустоты для КС-грамматик***, которая формулируется следующим образом: для данной КС-грамматики выяснить, пуст ли язык, ею порождаемый. Из предыдущих рассмотрений ясно, что для ответа на этот вопрос достаточно вычислить множество бесполезных нетерминалов грамматики и проверить, находится ли среди них аксиома грамматики. Язык, порождаемый КС-грамматикой, пуст тогда и только тогда, когда ее аксиома бесполезна.

Пример 8.7. Пусть множество P правил вывода КС-грамматики $G = (\{a, b\}, \{S, A, B, C\}, S, P)$ имеет вид

$$S \to aA \,|\, bB,$$
$$A \to bAa,$$
$$B \to aB \,|\, bS \,|\, a \,|\, b,$$
$$C \to BaA.$$

Применяя алгоритм из доказательства теоремы 8.3, получаем $N_0 = \{B\}$, так как в P имеются только два правила, правые части которых — терминальные цепочки $B \to a$ и $B \to b$. С целью вычисления множества N_1 найдем те правила, правые части которых кроме терминалов содержат только нетерминальный символ B: это будут правила $S \to bB$ и $B \to aB$, т.е. $N_1 = \{B, S\}$. Так как есть только одно правило, правая часть которого содержит вхождение символа S, а именно правило $B \to bS$, то $N_2 = N_1 = N_u = \{S, B\}$. Символы A и C бесполезны. Все правила, содержащие вхождения этих символов, можно удалить.

В итоге получим грамматику с таким множеством правил:

$$S \to bB,$$
$$B \to aB \,|\, bS \,|\, a \,|\, b.$$

Определение 8.4. Символ X объединенного алфавита $V \cup N$ КС-грамматики $G = (V, N, S, P)$ называют ***недостижимым***, если из аксиомы грамматики не выводится ни одна цепочка, содержащая вхождение символа X, т.е. не существует таких цепочек $\alpha, \beta \in (V \cup N)^*$, что $S \vdash_G^* \alpha X \beta$.

Из самого понятия недостижимого символа ясно, что такие символы можно удалить из грамматики (удалив все правила, которые их содержат) без изменения порождаемого ею языка. Займемся разработкой алгоритма удаления недостижимых символов.

Введем в рассмотрение понятие графа КС-грамматики.

Определение 8.5. Назовем ***графом КС-грамматики*** $G = (V, N, S, P)$ *ориентированный граф*, множество *вершин* которого равно $V \cup N \cup \{\lambda\}$, а множество *дуг* определяется следующим образом. Дуга из вершины с меткой A ведет в вершину B тогда и только тогда, когда правило $A \to \alpha B \beta$ $(\alpha, \beta \in (V \cup N)^*)$ принадлежит множеству P правил вывода грамматики G.

Понятие графа КС-грамматики ни в коем случае нельзя путать с понятием дерева вывода в КС-грамматике, так как дерево вывода строится по конкретному выводу, а граф КС-грамматики строится по множеству правил вывода и характеризует саму грамматику в целом.

Пример 8.8. Граф грамматики из примера 8.7 представлен на рис. 8.22. #

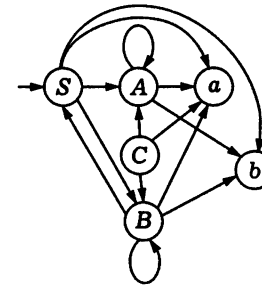


Рис. 8.22

Опираясь на определение 8.4, можно показать, что символ X КС-грамматики $G = (V, N, S, P)$ достижим тогда и только тогда, когда в графе грамматики существует путь из вершины S грамматики G в вершину X, и задача распознавания достижимости символов в КС-грамматиках сводится тем самым к задаче распознавания достижимости в ориентированных графах из заданной вершины. Для ее решения достаточно воспользоваться, например, алгоритмом *поиска в ширину* (см. **5.5**). В примере 8.8 (см. рис. 8.22) символ C не достижим.

Удаление бесполезных нетерминалов может привести к появлению недостижимых символов.

Пример 8.9. Модифицируем грамматику из примера 8.7, добавив к множеству ее терминалов новый символ d и к множеству ее правил вывода правило $A \to Cd$.

Граф модифицированной грамматики приведен на рис. 8.23. Нетерминалы A и C бесполезны, хотя и достижимы. Удаляя их вместе со всеми правилами, в которые они входят, получим КС-грамматику, граф которой изображен на рис. 8.24. Терминал d стал недостижимым.

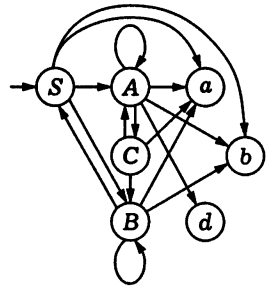


Рис. 8.23

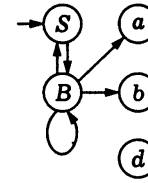


Рис. 8.24

Определение 8.6. КС-грамматика считается заданной в ***приведенной форме***, если множество ее правил вывода не содержит λ-правил и цепных правил, множество ее нетерминалов не содержит бесполезных нетерминалов, а объединенный алфавит не содержит недостижимых символов.

Рассмотренные выше алгоритмы любую КС-грамматику позволяют преобразовать к эквивалентной КС-грамматике, заданной в приведенной форме. Заметим, что поскольку удаление λ-правил может привести к появлению цепных правил (см. пример 8.6), а удаление бесполезных нетерминалов — к появлению недостижимых символов, то построение приведенной формы КС-грамматики нужно проводить в такой последовательности: 1) удалить λ-правила, 2) удалить цепные правила, 3) удалить бесполезные нетерминалы, 4) удалить недостижимые символы.

Замечание 8.6. К числу важных преобразований КС-грамматики относят также преобразование, состоящее в удалении аксиомы из правых частей правил вывода. Можно до-

казать, что любая КС-грамматика преобразуется к эквивалентной КС-грамматике, не содержащей вхождений аксиомы в правые части правил вывода. Действительно, для этого достаточно ввести новый нетерминал S_1 (отличный от всех нетерминалов исходной грамматики), объявить его аксиомой и добавить правило $S_1 \to S$, после чего появившееся цепное правило и недопустимые λ-правила, которые могут при этом возникнуть, удаляются согласно приведенным выше алгоритмам.

Пример 8.10. Грамматика с множеством правил

$$S \to aSa|bSb|a|b|\lambda,$$

порождающая множество всех *палиндромов* в алфавите $\{a, b\}$ (см. пример 7.5.в), преобразуется после удаления аксиомы из правых частей правил вывода в грамматику с аксиомой S_1 и следующим множеством правил:

$$S_1 \to S;$$
$$S \to aSa|bSb|a|b|\lambda.$$

Появились, как мы видим, цепное правило и, поскольку символ S в новой грамматике уже не является аксиомой, λ-правило $S \to \lambda$, которое следует удалить. Выполняя это, получим окончательно следующее множество правил:

$$S_1 \to aSa|bSb|aa|bb|a|b|\lambda,$$
$$S \to aSa|bSb|aa|bb|a|b. \quad \#$$

Заметим, что если КС-грамматика задана в приведенной форме и правые части правил не содержат вхождений аксиомы, то единственное допустимое правило с пустой правой частью (левой частью его является аксиома), если оно существует, используется только для порождения пустой цепочки. Исходная грамматика из примера 8.10 этим свойством не обладает, так

как правило $S \to \lambda$ используется всякий раз при выводе слова четной длины.

Кроме того, КС-грамматика, заданная в приведенной форме и не содержащая вхождений аксиомы в правых частях правил вывода, обладает еще одним интересным свойством: дерево вывода любой непустой цепочки является *λ-свободным*. Таким образом, без ограничения общности можно считать деревья выводов непустых цепочек в КС-грамматиках λ-свободными.

8.3. Лемма о разрастании для КС-языков

Для *КС-языков* выполняется „лемма о разрастании“, аналогичная лемме о разрастании для *регулярных языков* (см. **7.8**). Она формулирует необходимое условие того, что язык является контекстно-свободным.

Теорема 8.4 (лемма о разрастании для КС-языков). Для любого КС-языка L существует натуральная константа k (зависящая от L), такая, что любая *цепочка* $z \in L$, *длина* которой $|z| > k$, представима в виде *соединения* пяти *цепочек*: $z = uxwyv$, где $|xy| > 0$, $|xwy| \leqslant k$, и каждая цепочка $z_n = ux^n wy^n v$, $n \geqslant 0$, принадлежит L.

◀ Так как L — КС-язык, то существует *КС-грамматика* $G = (V, N, S, P)$, порождающая язык L, т.е. $L = L(G)$.

Фиксируем произвольно *нетерминал* $A \in N$ и рассмотрим множество $\mathcal{D}(A)$ всех *выводов*, начинающихся с A, таких, что *высота дерева вывода* не превышает числа $|N| + 1$ нетерминалов грамматики G, увеличенного на единицу. Множество $\mathcal{D}(A)$ конечно, так как высота дерева вывода, а следовательно, и его длина ограничены сверху (см. замечание 8.2). Таким образом, каждое множество $\mathcal{D}(A)$ при $A \in N$ есть конечное множество конечных выводов. Множество выводов $\mathcal{D}$ определим как *объединение множеств* $\mathcal{D}(A)$ по всем $A \in N$. Ясно, что и это множество выводов конечно, так как конечно множество нетерминалов грамматики. Введем подмножество $\mathcal{A}$ всех таких

цепочек в объединенном алфавите, что они являются последними цепочками выводов из $\mathcal{D}$. Иначе говоря, $\mathcal{A}$ — это множество всех таких цепочек α в объединенном алфавите, что существует дерево вывода α (из некоторого нетерминала грамматики G), высота которого не больше $|N|+1$. В силу конечности множества $\mathcal{D}$ конечно и множество $\mathcal{A}$. Тогда положим

$$k = \max_{\alpha \in \mathcal{A}} |\alpha|,$$

т.е. введем константу k как наибольшую длину среди всех цепочек множества $\mathcal{A}$.

Пусть теперь $z \in L$, причем $|z| > k$. Тогда $S \vdash_G^* z$, и в силу определения константы k любое дерево вывода цепочки z имеет высоту, большую $|N|+1$. Фиксируем какое-нибудь дерево T вывода цепочки z.

Выделим в дереве T ***максимальное поддерево*** с ***корнем*** R, высота которого равна $|N|+1$ (рис. 8.25).

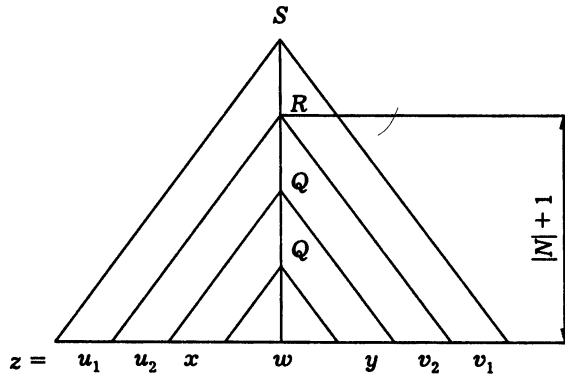


Рис. 8.25

Тогда, согласно следствию 8.1, получим (для некоторых терминальных цепочек u_1 и v_1)

$$S \vdash^* u_1 R v_1 \vdash^* u_1 z_1 v_1, \tag{8.1}$$

где $z = u_1 z_1 v_1$, т.е. существует вывод цепочки $z_1 = u_2 x w y v_2$, являющейся подцепочкой цепочки z, из нетерминала R, такой, что высота дерева этого вывода равна $|N| + 1$.

Напомним, что высота дерева есть максимальная *длина пути* в этом дереве из корня в *лист*. Так как длина пути в дереве вывода КС-грамматики равна числу замен нетерминалов в некотором фрагменте вывода (см. **8.1**), то при выводе цепочки z_1 из R число замен нетерминалов строго больше числа всех нетерминалов грамматики G. Это означает, что в выводе z_1 из R какой-то нетерминал Q повторяется дважды (возможно, что $Q = R$) и на некотором пути из вершины R в один из листьев будут по крайней мере две разные вершины, помеченные одним и тем же нетерминалом Q. Тогда рассмотрим два максимальных поддерева дерева T, корневые вершины которых являются указанными выше вершинами с меткой Q (см. рис. 8.25). Выделению этих максимальных поддеревьев соответствует выделение фрагментов вывода z_1 из R, таких, что для некоторых терминальных цепочек u_2, v_2, x, y, w выполняется выводимость

$$R \vdash^* u_2 Q v_2 \vdash^* u_2 x Q y v_2 \vdash^* u_2 x w y v_2, \tag{8.2}$$

где $z_1 = u_2 x w y v_2$.

Соотношения (8.2) показывают, что имеют место выводимости

$$Q \vdash^* x Q y \tag{8.3}$$

и

$$Q \vdash^* w. \tag{8.4}$$

Это значит, что существует дерево вывода цепочки xwy из нетерминала Q („большое“ дерево с корневой вершиной Q на рис. 8.25) и в нем — максимальное поддерево с корневой вершиной, помеченной тем же символом Q, являющееся деревом вывода цепочки w из Q („маленькое“ дерево с корневой вершиной Q на рис. 8.25). При этом, так как „большое“ дерево с

корневой вершиной Q является поддеревом дерева с корневой вершиной R высотой $|N|+1$, то его высота будет не больше чем $|N|+1$, откуда в силу выбора константы k имеем $|xwy| \leqslant k$.

Объединяя (8.1) и (8.2), получим такое представление вывода цепочки z из аксиомы S:

$$S \vdash^* u_1 R v_1 \vdash^* \vdash^* u_1 u_2 Q v_2 v_1 \underbrace{\vdash^* u_1 u_2 x Q y v_2 v_1} \vdash^* u_1 u_2 x w y v_2 v_1. \quad (8.5)$$

Полагая $u = u_1 u_2$ и $v = v_2 v_1$, получим $z = uxwyv$, и тем самым мы доказали представление цепочки z в виде соединения определенных пяти цепочек.

Из (8.5) следует, что, получив (при выводе z из S) цепочку $u_1 u_2 Q v_2 v_1$, можно с учетом (8.4) сразу из нетерминала Q вывести цепочку w, и тогда

$$S \vdash^* u_1 R v_1 \vdash^* u_1 u_2 Q v_2 v_1 \vdash^* u_1 u_2 w v_2 v_1 = uwy \quad (8.6)$$

(в выводе (8.5) выбрасываем последовательность шагов над фигурной скобкой).

В то же время вывод (8.5) можно с учетом (8.3) модифицировать так, что после получения цепочки $u_1 u_2 Q v_2 v_1 = uQv$ многократно (не менее одного раза) повторять вывод цепочки xQy из Q:

$$S \vdash^* u_1 R v_1 \vdash^* u_1 u_2 Q v_2 v_1 = \\ = uQv \vdash^* uxQyv \vdash^* uxxQyyv \vdash^* \ldots \vdash^* \\ \vdash^* ux^n Q y^n v \vdash^* ux^n w y^n v \quad (8.7)$$

(в выводе (8.5) последовательность шагов над фигурной скобкой повторяем n раз).

Итак, в силу (8.6) цепочка $z_0 = uwv \in L$, а в силу (8.7) и цепочка $z_n = ux^n w y^n v$ для любого $n > 0$ принадлежит языку L. Тем самым мы доказали, что $(\forall n \geqslant 0)(ux^n w y^n v \in L)$.

Осталось доказать, что длина цепочки xy не меньше 1 (или, что равносильно, цепочки x и y не являются одновременно пустыми).

Согласно теореме 8.2, можно считать без ограничения общности, что в множестве правил вывода грамматики G нет ни λ-правил (кроме, быть может, правила $S \to \lambda$), ни цепных правил. Кроме того, в силу замечания 8.6 можно считать, что правило $S \to \lambda$ применяется только при выводе пустой цепочки, а поскольку высота дерева цепочки z строго больше 1, то $z \neq \lambda$ и правило $S \to \lambda$ не применяется при выводе z из аксиомы S.

Тогда, предполагая, что $x = y = \lambda$, получим $Q \vdash^* Q$, а это возможно лишь при условии применения в соответствующем выводе цепных правил или правил с пустой правой частью, что ввиду сказанного выше не может иметь места. Итак, цепочка xy не пуста, т.е. $|xy| > 0$, что и завершает доказательство теоремы. ▶

Следствие 8.2. В любом бесконечном КС-языке существует последовательность слов, длины которых образуют возрастающую арифметическую прогрессию.

◀ Пусть КС-язык L бесконечен. Тогда в нем нет цепочки наибольшей длины. Следовательно, найдется цепочка $z \in L$, длина которой больше константы k, определяемой леммой о разрастании. Согласно этой лемме, цепочка z может быть представлена в виде $z = uxwyv$ для некоторых цепочек u, x, w, y, v, таких, что $|xwy| \leqslant k$, $|xy| > 0$. Тогда последовательность $\{z_n = ux^n wy^n v\}_{n \geqslant 0}$ является искомой, так как длины ее цепочек образуют возрастающую арифметическую прогрессию со знаменателем $|xy|$. ▶

Замечание 8.7. Любой конечный язык $\{u_1, \ldots, u_m\}$ (в каком-то алфавите V) порождается КС-грамматикой с множеством правил $S \to u_1 | \ldots | u_m$. Дерево вывода любой цепочки в этой грамматике имеет высоту, не большую 1 (высоту 0 име-

ет дерево вывода нулевой длины аксиомы S из себя самой, а каждая терминальная цепочка, т.е. цепочка u_i, $i = \overline{1, m}$, имеет дерево вывода высоты 1). Следовательно, в данном случае константа k равна наибольшей длине цепочки u_i и цепочки, длина которой была бы больше k, не существует.

Таким образом, для конечного языка утверждение леммы о разрастании тривиально выполняется, а именно, не существует цепочек, длины которых превосходят константу k.

Пример 8.11. а. Язык $\{a^{n^2}: n \geqslant 0\}$ не является контекстно-свободным, поскольку из длин принадлежащих ему слов нельзя образовать арифметическую прогрессию: $(n+1)^2 - n^2 = = 2n + 1$.

б. Язык $\{a^s: s$ — простое число$\}$ не является КС-языком по той же причине.

в. Рассмотрим язык $\{a^n b^n c^n \mid n \geqslant 0\}$. Длины его слов образуют арифметическую прогрессию, но тем не менее этот язык не является контекстно-свободным. Действительно, предположим противное. Тогда для любого достаточно большого n слово $a^n b^n c^n$ можно представить в виде $uxwyv$. Проанализируем, как могут располагаться в слове цепочки x, w и y. Поскольку условие леммы о разрастании должно выполняться для всех достаточно длинных слов языка, т.е. слов, длина которых строго больше константы k из леммы, то в целях проверки невыполнения условия можно брать слова, длина которых заведомо превышает указанную константу*. Для анализируемого в этой задаче языка примем, что $n > k$. Тогда возможны следующие случаи:

*Предполагая, что язык является контекстно-свободным, мы тем самым в силу леммы о разрастании предполагаем, что оказывается фиксированной какая-то константа k, о которой идет речь в условии леммы. Мы не можем знать значения этой константы, но это нам и не нужно: достаточно знать, что вследствие нашего предположения константа k как-то фиксирована, и мы можем быть уверены в том, что в бесконечном КС-языке найдутся цепочки, длины которых превысят k.

1) $xwy = \alpha^l$, где $l \leqslant k < n$, а α обозначает одну из букв a, b, c (т.е. цепочка xwy целиком расположена или в „зоне“ символов a, или в „зоне“ символов b, или в „зоне“ символов c);

2) $xwy = a^l b^m$ или $xwy = b^l c^m$, где $l + m \leqslant k < n$ и $l, m > 0$ (т.е. цепочка xwy расположена „на стыках зон“ различных символов и содержит ненулевое число символов каждой из „зон“). В силу выбора числа n никакие другие способы расположения подцепочки xwy в цепочке $a^n b^n c^n$ невозможны (отсюда следует, что подходящий выбор числа n позволил нам уменьшить число вариантов расположения подцепочки xwy в цепочке языка).

Теперь если мы станем повторять („накачивать“) цепочки x и y в первом случае, то начнет расти число одного из символов a, b или c, тогда как число остальных двух символов будет оставаться прежним, и получаемые при этом цепочки уже не будут принадлежать заданному языку. Во втором же случае при „накачке“ возникнут цепочки, содержащие вхождение цепочки bd или cb, что также недопустимо (символы a, b, c начнут „перемешиваться“). Получается, что мы не можем представить любую достаточно длинную цепочку $a^n b^n c^n$ в виде $uxwyv$ так, чтобы при этом выполнялось условие леммы о разрастании. Следовательно, исходный язык не является контекстно-свободным. #

В целом техника доказательства с помощью леммы о разрастании для КС-языков „отрицательных“ утверждений типа „данный язык не является контекстно-свободным“ аналогична технике доказательства утверждений о нерегулярности языков с использованием леммы о разрастании для регулярных языков.

Обратим внимание на различие структур „накачиваемых“ цепочек в регулярных и контекстно-свободных языках.

Лемма о разрастании для регулярных языков утверждает, что любая достаточно длинная цепочка регулярного языка представима как соединение трех подцепочек, из которых средняя подцепочка ограничена сверху по длине и не пуста, ее можно „накачивать“, повторяя любое число раз, в том числе

и ни разу, т.е. можно выбросить вообще, и такая „накачка“ не выводит за пределы данного языка (рис. 8.26).

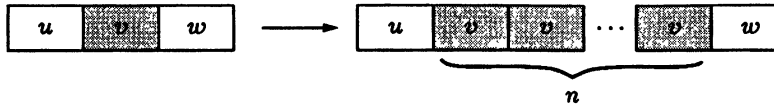


Рис. 8.26

Лемма о разрастании для КС-языков утверждает в аналогичной ситуации, что цепочка языка представима как соединение пяти подцепочек, причем если рассмотреть три средние подцепочки, то „накачиваются“ края — первая и третья из них, а самая середина не меняется (рис. 8.27).

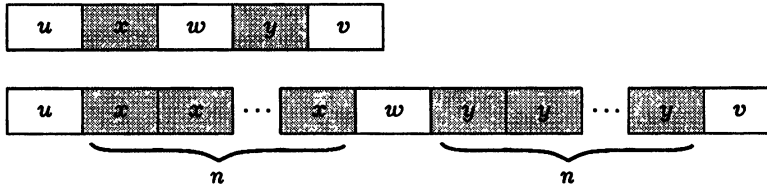


Рис. 8.27

Обратим внимание на то, что ограничение сверху длины подцепочки xwy константой k, вытекающее из леммы о разрастании, очень важно и, не используя его, вообще говоря, нельзя доказать, что данный язык не является контекстно-свободным. Рассмотрим в этой связи такой пример.

Пример 8.12. Пусть язык L определен как множество всех цепочек в алфавите $\{a,b\}$ вида $a^nb^ma^nb^m$, $n \geqslant 0$. Докажем, что он не является контекстно-свободным.

Предполагая, что L — КС-язык, получим, что для достаточно больших n и m цепочка $a^nb^ma^nb^m$ допускает представление в виде $uxwyv$. Выберем числа n и m так, что они больше, чем определенная леммой о разрастании константа k для языка L. Анализируем варианты „размещения“ подцепочки xwy в цепочке $a^nb^na^nb^m$: очевидно, что „накачка“ невозможна, если

цепочки x и y обе целиком находятся в „зоне“ какого-то одного символа, a или b; если же, скажем, $x = a^p b^s$ при p, $s \neq 0$, т.е. цепочка x находится „на стыке зон“ символов a и b, то при „накачке“ возникнет более чем одно вхождение подцепочки ba в цепочку языка L, что противоречит определению этого языка; если же $x = b^p a^s$ при p, $s \neq 0$, то при „накачке“ возникнет более двух вхождений подцепочки ab, что также невозможно. Аналогично анализируется размещение цепочки y „на стыках зон“. В силу того что n, $m > k$, а $|xwy| \leqslant k$, никакие другие варианты расположения подцепочки xwy невозможны.

Если ограничение на длину этой подцепочки опустить, то окажется возможным такое ее вхождение в цепочку $a^m b^n a^m b^n$, что $x = a^s$, $w = a^p b^m a^q$ и $y = a^r$ для некоторых положительных s, p, q, r. Тогда при $s = r$ (чего априори нельзя отвергать, так как условия леммы о разрастании не требуют, чтобы длины цепочек x и y были обязательно различными) „накачка“ цепочек x и y дала бы „синхронное“ увеличение числа символов a при неизменном числе символов b, что не вывело бы нас за пределы языка L. Таким образом, нам тогда не удалось бы доказать, что определяемое леммой о разрастании представление цепочки $a^m b^n a^m b^n$ не может иметь места. #

Рассмотрим еще один пример, важный для понимания леммы о разрастании и ее применений.

Пример 8.13. Зададим язык

$$L' = \{a^n b^n c^p \colon n, p \geqslant 0 \text{ и } p \geqslant n\}.$$

Здесь при размещении „накачиваемых“ цепочек в „зоне“ символов c „накачка“ не выводит за пределы языка L' (так как в его цепочках число символов c может на сколько угодно превышать число символов a и b). Тем не менее и в этом случае условие леммы о разрастании не выполняется.

Действительно, для достаточно большого n возьмем цепочку $a^n b^n c^n \in L'$ (т.е. цепочку языка L' при $n = p$). Если в представлении такой цепочки в виде $a^n b^n c^n = uxwyv$ (согласно лемме о

разрастании) цепочка $xwy = c^l$ при $0 < l \leqslant n$ (т.е. „накачиваемые“ цепочки располагаются целиком в „зоне“ символов c), то цепочка $uwv = a^n b^n c^{n-|xy|} \notin L'$, так как $|xy| \geqslant 1$ и $n - |xy| < n$, т.е. число символов c окажется меньше, чем число символов a и b. Таким образом, несмотря на то что „накачка“ цепочек x, y в данном случае возможна, нельзя их выбросить, оставаясь в пределах языка. Невозможность других вариантов расположения подцепочки xwy в цепочке языка L' доказывается точно так же, как и в предыдущих примерах. #

Итак, нужно помнить, что выполнение условий леммы о разрастании предполагает и возможность выбрасывания „накачиваемых“ цепочек — все цепочки $z_n = ux^n wy^n v$ из условия леммы при $n \geqslant 0$ должны оставаться в языке L, и если условие леммы выполняется при всех положительных n, но не выполняется при $n = 0$, то этого достаточно для признания того, что цепочка $z = uxwyv \in L$ не удовлетворяет условию леммы о разрастании.

Аналогичная ситуация имеет место, конечно, и при использовании леммы о разрастнии для регулярных языков.

8.4. Магазинные автоматы

Автоматы с магазинной памятью, или ***магазинные автоматы*** (сокращенно — ***МП-автоматы***), образуют класс *распознающих моделей* для *КС-языков* точно так же, как *конечные автоматы* являются распознающими моделями в классе *регулярных языков*.

Понятие МП-автомата является частным случаем общего интуитивного понятия *автомата*, которое мы ввели в **7.5**. Как и всякий автомат, МП-автомат — это устройство, имеющее *блок управления*, *входную ленту*, *головку* и *блок внутренней памяти* в виде ***магазина МП-автомата*** (или *стека*).

Блок управления в каждый момент времени находится в одном из конечного множества Q *состояний*.

Входная лента предполагается „полубесконечной“, т.е. она имеет начало („левый край“), но не имеет конца. Лента разделена на ячейки, пронумерованные от крайней левой ячейки натуральными числами, начиная с единицы; в каждой ячейке может быть записан символ алфавита V, называемого ***входным алфавитом МП-автомата***. В каждый момент времени читающая головка обозревает (или читает) некоторую ячейку входной ленты.

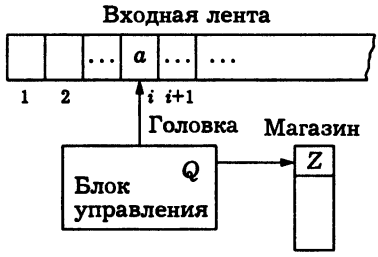


Рис. 8.28

Если в этой ячейке записан некоторый *входной символ*, т.е. символ входного алфавита, то его называют ***обозреваемым*** (в данный момент) ***входным символом*** (рис. 8.28).

Предполагается, что если на ленте записана некоторая непустая цепочка $x = x(1)x(2)\ldots x(k)$ во входном алфавите, то ее символы последовательно записаны в ячейках от первой до k-й, без пропусков, так, что символ $x(i)$ записан в i-й ячейке (рис. 8.29).

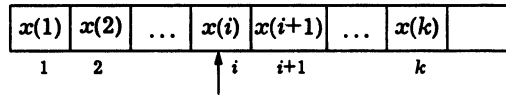


Рис. 8.29

МП-автомат может читать цепочку x, продвигая головку только в одном направлении, слева направо, по шагам, причем за каждый шаг головка переходит от обозреваемой ячейки к следующей. Если в какой-то момент времени обозревается символ $x(i)$, $1 \leqslant i \leqslant k$, записанной на ленте цепочки x, то *непрочитанной частью входной цепочки* x будет подцепочка $x(i+1)\ldots x(k)$. В частности, при $i = k$ непрочитанная часть совпадает с пустой цепочкой, и тогда говорят, что цепочка x полностью прочитана МП-автоматом (рис. 8.30).

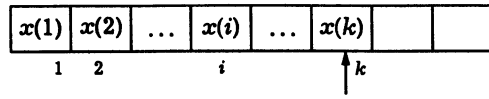


Рис. 8.30

Магазин МП-автомата устроен и работает следующим образом. Как и входная лента, магазин является полубесконечным и разделенным на пронумерованные ячейки, в каждой из которых может быть записан символ алфавита Г, называемого ***магазинным алфавитом МП-автомата***. Входной и магазинный алфавиты МП-автомата могут пересекаться и даже совпадать друг с другом. Символы алфавита Г называют ***магазинными символами***. Первую ячейку магазина называют его ***верхней ячейкой*** (иногда просто ***верхом магазина****). Символ, в данный момент записанный в верхней ячейке магазина, называют ***верхним символом магазина***.

Единственная операция, которую МП-автомат может реализовать с магазином, состоит в следующем: верхний символ Z магазина заменяется некоторой цепочкой γ магазинных символов. При этом если цепочка γ непустая, то она записывается в первых m ячейках, где $m = |\gamma|$ (*длине цепочки* γ), так, что ее первый символ становится верхним символом магазина.

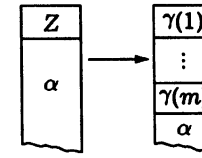


Рис. 8.31

Если до замены Z цепочкой γ под верхней ячейкой** (т.е. в ячейках, начиная со второй) была записана какая-то цепочка α, то после замены она сдвигается „в глубь“ магазина и оказывается записанной уже в ячейках, начиная с $(m+1)$-й (рис. 8.31).

Если же верхний символ Z заменяется пустой цепочкой λ, то после такой замены верхним символом магазина становится

*Таким образом, в неформальном описании МП-автомата лента читается „слева направо“, а магазин — „сверху вниз“.

**Полагают, что, как и символы цепочек на входной ленте, символы цепочек, записанных в магазин, располагаются в последовательных ячейках „сверху вниз“ без пропусков ячеек.

первый символ цепочки α (записанной под верхней ячейкой магазина), т.е. цепочка α „поднимается“ на одну ячейку.

В случае, когда α — пустая цепочка, замена верхнего символа магазина пустой цепочкой γ приводит к опустошению магазина (ни в одной из ячеек магазина не записан магазинный символ). Заметим, что, по определению, с пустым магазином МП-автомат не может производить никаких операций.

Описанная выше операция с магазином составляет основу поведения МП-автомата, его, образно говоря, „динамику“. Эта „динамика“ определяется ***системой команд МП-автомата***, которая, аналогично *системе команд конечного автомата*, определяется как конечное множество δ команд, каждая из которых записывается в виде

$$qaZ \to r\gamma, \tag{8.8}$$

где q и r — состояния из множества Q; a — входной символ или пустая цепочка; Z — магазинный символ; γ — цепочка магазинных символов (может быть и пустая).

Если в системе команд δ МП-автомата M есть команда (8.8), то M может в данный момент времени выполнить эту команду, если и только если в данный момент времени его блок управления находится в состоянии q, головка обозревает входной символ a (при условии, что $a \neq \lambda$), а верхним символом магазина является символ Z. Выполнение команды (8.8) состоит в замене верхнего символа магазина цепочкой γ (как это описано выше), переходе блока управления в состояние r (которое может и совпадать с состоянием q) и в продвижении головки на одну ячейку вправо (если a в команде (8.8) не есть пустая цепочка λ) (рис. 8.32).

Если же в команде (8.8) $a = \lambda$, то такую команду МП-автомат может выполнить всякий раз, когда его блок управления окажется в состоянии q, а верхним символом магазина будет символ Z. В этом случае выполнение команды никак не

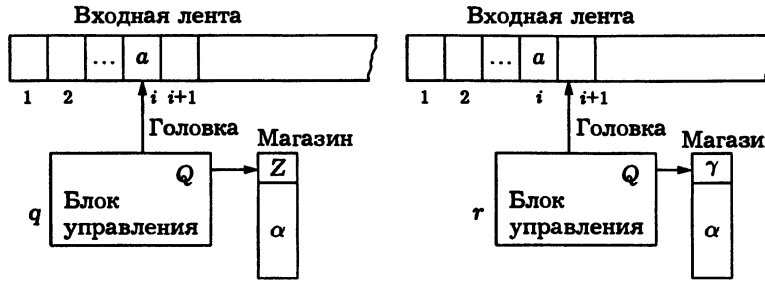


Рис. 8.32

зависит от содержимого входной ленты, а после ее выполнения головка остается на прежнем месте.

Рассмотренную выше процедуру выполнения команды вида (8.8) называют ***шагом*** (или ***тактом***) ***работы МП-автомата*** (при $a = \lambda$ такт работы называют ***λ-тактом***).

Предположим теперь, что в множестве состояний Q блока управления МП-автомата M с системой команд δ фиксировано некоторое ***начальное состояние*** q_0 и подмножество ***заключительных состояний*** F.

Пусть в какой-то начальный момент времени блок управления МП-автомата M находится в начальном состоянии q_0, на ленте записана непустая цепочка x, головка обозревает первую ячейку ленты и, следовательно, первый символ цепочки x является обозреваемым, а в магазине записан только один специальный символ Z_0, называемый ***начальным символом магазина***.

Тогда, если МП-автомат M, выполняя некоторую последовательность команд из δ, прочитывает полностью цепочку x, в результате чего блок управления переходит в некоторое заключительное состояние $q_f \in F$, говорят, что МП-автомат M допускает (распознает, воспринимает) цепочку x, которую называют ***допустимой цепочкой МП-автомата***. Множество всех допустимых цепочек МП-автомата M образует ***язык, допускаемый*** этим ***МП-автоматом***.

Далее мы докажем, что язык любого МП-автомата является КС-языком и, наоборот, любой КС-язык есть язык некоторого МП-автомата. В этом смысле мы и говорим об МП-автоматах как о „распознавателях“ КС-языков: всякий МП-автомат допускает те и только те цепочки входных символов, которые порождаются некоторой *КС-грамматикой*. Но чтобы доказывать какие-либо утверждения об МП-автоматах, нужно сначала дать строгое математическое определение этого понятия, не апеллируя уже к наглядным, но не определенным формально идеям ленты, магазина, головки и т.п. Формализация изложенного выше описания проводится во многом аналогично построению определения порождающей грамматики.

Определение 8.7. МП-автомат задается упорядоченной семеркой

$$M = (Q,\ V,\ \Gamma,\ q_0,\ F,\ Z_0,\ \delta),$$

где Q — конечное множество состояний; V — алфавит, называемый входным алфавитом; Γ — алфавит, называемый магазинным алфавитом; $q_0 \in Q$ — начальное состояние; $F \subseteq Q$ — подмножество заключительных состояний; $Z_0 \in \Gamma$ — начальный магазинный символ; δ — система команд, определенная как конечное множество команд, каждая из которых записывается в виде (8.8):

$$qaZ \to r\gamma,$$

где знак „$\to$“ (стрелка) — внешний символ (не принадлежащий ни одному из указанных алфавитов); $q, r \in Q$; $a \in V \cup \{\lambda\}$; $Z \in \Gamma$; $\gamma \in \Gamma^*$.

Замечание 8.8. Нелишне обратить внимание на то, что упоминание о ленте и магазине в приведенном выше формальном определении совершенно необязательно. Формально для определения МП-автомата достаточно задать два алфавита, конечное множество состояний, выделив в нем начальное состояние и подмножество заключительных состояний, а также

систему команд. Образы ленты, магазина, да и сам термин „состояние“, являются метафорами, которые позволяют связать формальное определение с его содержательной интерпретацией, идеей некоего „устройства“ для чтения слов, с описания которого мы начали этот параграф. #

Любую цепочку в алфавите V будем называть входной цепочкой, а сами элементы входного алфавита — входными символами; точно так же любую цепочку в алфавите Г будем называть магазинной цепочкой, а символы этого алфавита — магазинными символами. Упорядоченную тройку до знака „$\rightarrow$“ (стрелки) в записи команды называют ***левой частью команды***, а упорядоченную пару после знака „$\rightarrow$“ — ***правой частью команды***.

Пример 8.14. Рассмотрим МП-автомат

$$M_1 = (\{q_0, q_1, q_2\},\ \{0, 1\},\ \{Z, 0\},\ q_0,\ \{q_0\},\ Z, \delta_1)$$

с таким множеством команд δ_1:

$$q_0 0Z \rightarrow q_1\ 0Z,$$
$$q_1 00 \rightarrow q_1\ 00,$$
$$q_1 10 \rightarrow q_2 \lambda,$$
$$q_2 10 \rightarrow q_2 \lambda,$$
$$q_2 \lambda Z \rightarrow q_0 \lambda.$$

На рис. 8.33 проиллюстрировано выполнение первых двух команд. При записи системы команд этого конкретного МП-автомата мы в правых частях первых двух команд отделили магазинную цепочку от состояния пробелом для того, чтобы не возник соблазн прочитать левую и правую части этих команд одинаково. Таким приемом записи мы хотим подчеркнуть, что левая часть команды МП-автомата — упорядоченная тройка, а правая — упорядоченная пара.

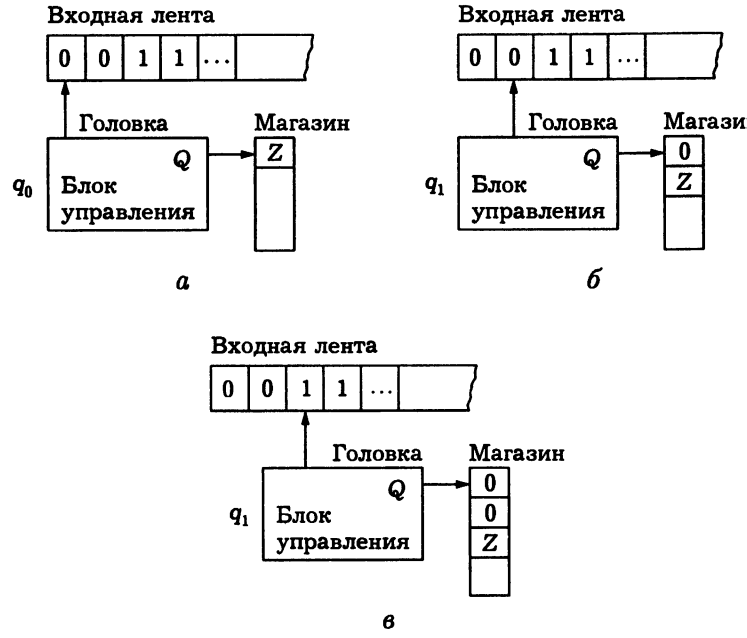


Рис. 8.33

Замечание 8.9. Систему команд МП-автомата можно понимать как способ определения некоторой *конечной функции*, которую называют ***функцией переходов МП-автомата*** (по аналогии с *функцией переходов конечного автомата*). Эта функция может быть определена как отображение вида

$$\delta\colon Q \times (V \cup \{\lambda\}) \times \Gamma \to \mathbf{Fin}(Q \times \Gamma^*),$$

где использовано обозначение $\mathbf{Fin}(A)$ для множества всех конечных подмножеств множества A. #

„Динамику“ МП-автомата, т.е. процесс перехода его из одного состояния в другое в соответствии с обозреваемыми символами на ленте и содержимым магазина, удобнее всего описывать в терминах *конфигураций*. Для этого общее понятие конфигурации автомата, которое на интуитивном уровне

было введено в **7.5**, необходимо уточнить применительно к МП-автомату и определить также отношение выводимости на множестве конфигураций.

Определение 8.8. ***Конфигурацией МП-автомата*** M называют упорядоченную тройку вида (q, x, β), где $q \in Q$ — состояние, $x \in V^*$ — входная цепочка, β — магазинная цепочка.

Конфигурацию $C' = (r, w, \gamma\alpha)$ называют ***непосредственно выводимой*** из конфигурации $C = (q, aw, Z\alpha)$, если в множестве команд МП-автомата есть команда (8.8). Отношение непосредственной выводимости на множестве конфигураций МП-автомата M обозначаем $\vdash_M$, используя запись $C \vdash_M C'$ (и опуская, если это не вредит пониманию, обозначение самого МП-автомата).

Итак, непосредственная выводимость

$$(q, aw, Z\alpha) \vdash_M (r, w, \gamma\alpha) \tag{8.9}$$

имеет место тогда и только тогда, когда в системе δ команд МП-автомата M есть команда (8.8), которую в этом случае называют ***применимой к конфигурации*** $(q, aw, Z\alpha)$.

Таким образом, неформально конфигурация показывает состояние блока управления, непрочитанную часть входной цепочки (первый символ этой цепочки, если она не пуста, является обозреваемым) и содержимое магазина (первый символ магазинной цепочки β, если она не пуста, есть верхний символ магазина). Если вторая компонента конфигурации — пустая цепочка, то это означает, что входная цепочка прочитана полностью. Если же пуста третья компонента, то это означает, что пуст магазин.

На рис. 8.33 изображены (в терминах приведенного выше содержательного описания) две конфигурации МП-автомата из примера 8.14, причем вторая конфигурация непосредственно выводится из первой. Понятие непосредственной выводимости, таким образом, формализует данное выше понятие такта (или

шага) работы МП-автомата. Заметим, что если в (8.9) и в команде (8.8) $a = \lambda$, т.е. имеет место λ-такт, то команда (8.8) применима к любой конфигурации, у которой (говоря неформально) состояние блока управления есть q, а верхний символ магазина — Z, причем, подчеркнем это, независимо от содержимого входной ленты (т.е. для любой входной цепочки w). Если же a есть входной символ, то применимость команды к конфигурации определяется состоянием, обозреваемым символом на ленте и верхним символом магазина.

Любую конфигурацию вида (q_0, x, Z_0) называют ***начальной***, а любую конфигурацию вида (q_f, λ, λ), где $q_f \in F$, — ***заключительной***. Обратим внимание на то, что заключительная конфигурация — это конфигурация с пустым магазином. Множество всех заключительных конфигураций находится, таким образом, во взаимно однозначном соответствии с множеством заключительных состояний МП-автомата. Из заключительной конфигурации не может быть выведена ни одна конфигурация, так как к ней не применима ни одна команда. Конфигурацию МП-автомата, не являющуюся заключительной и к которой не применима ни одна команда, называют ***тупиковой***.

Определение 8.9. ***Выводом на множестве конфигураций МП-автомата*** M называют последовательность C_0, $C_1, \ldots, C_n, \ldots$ (конечную или бесконечную) таких его конфигураций, что для любого $i \geqslant 0$ имеет место $C_i \vdash C_{i+1}$, если C_{i+1} существует.

Если вывод конечен и C_n — его последняя конфигурация, то число n называют ***длиной вывода***. В этом случае будем говорить, что вывод $C_0, C_1, \ldots, C_n$ связывает конфигурацию C_0 с конфигурацией C_n.

Конфигурацию C' называют ***выводимой*** из конфигурации C, обозначая это как $C \vdash^* C'$, если существует вывод, связывающий C с C', т.е. вывод $C_0, C_1, \ldots, C_n$, такой, что $C_0 = C$, а $C_n = C'$.

В частности, при $n = 0$ получаем, что любая конфигурация выводится сама из себя; при $n = 1$ получаем непосредственную выводимость C' из C. В общем случае говорят, что конфигурация C' выводится из конфигурации C за n шагов, записывая при этом $C \vdash^n C'$. Желая подчеркнуть, что существует вывод ненулевой длины конфигурации C' из конфигурации C, т.е. $C \vdash^n C'$ и $n > 0$, записывают $C \vdash^+ C'$. #

Таким образом, понятие выводимости для конфигураций МП-автомата вводится аналогично таковому для грамматики. И к тому же сохраняется полная аналогия с понятием *достижимости* в *ориентированных графах.*

В терминах конфигураций может быть дано и строгое определение языка МП-автомата.

Определение 8.10. Входную цепочку x называют допустимой цепочкой МП-автомата M (см. определение 8.7), если на множестве конфигураций M существует вывод, связывающий начальную конфигурацию (q_0, x, Z_0) с заключительной конфигурацией (q_f, λ, λ), где $q_f \in F$, т.е.

$$(q_0,\ x,\ Z_0) \vdash^*_M (q_f,\ \lambda,\ \lambda).$$

Язык, допускаемый МП-автоматом M (или просто язык МП-автомата M), — это множество всех его допустимых цепочек.

МП-автоматы M_1 и M_2 называют ***эквивалентными***, если их языки совпадают, т.е. $L(M_1) = L(M_2)$.

Любой вывод на множестве конфигураций МП-автомата, связывающий начальную конфигурацию $C_0^x = (q_0, x, Z_0)$ с одной из заключительных, называют ***допускающей последовательностью конфигураций для цепочки*** x. Цепочка x принадлежит языку $L(M)$ тогда и только тогда, когда для нее существует допускающая последовательность конфигураций. Тем не менее может оказаться так, что даже в том случае, когда $x \in L(M)$, из начальной конфигурации C_0^x можно вывести отнюдь не только заключительную конфигурацию.

Пример 8.15. Пусть МП-автомат

$$M_2 = (\{q_0, q_1, q_2\},\ \{a, b\},\ \{Z, a, b\},\ q_0, \{q_2\},\ Z,\ \delta_2)$$

определен множеством команд δ_2:

$$\begin{aligned}
&q_0aZ \to q_0aZ, && (1)\\
&q_0bZ \to q_0bZ, && (2)\\
&q_0aa \to q_0aa, && (3)\\
&q_0aa \to q_1\lambda, && (4)\\
&q_0ab \to q_0ab, && (5)\\
&q_0ba \to q_0ba, && (6)\\
&q_0bb \to q_0bb, && (7)\\
&q_0bb \to q_1\lambda, && (8)\\
&q_1aa \to q_1\lambda, && (9)\\
&q_1bb \to q_1\lambda, && (10)\\
&q_1\lambda Z \to q_2\lambda. && (11)
\end{aligned}$$

Можно доказать, что этот МП-автомат допускает зеркальный язык $\{xx^R \in \{a, b\}^*\}$, где x^R обозначает *инверсию цепочки* x.

Приведем допускающую последовательность конфигураций для цепочки $abba$:

$$(q_0,\ abba,\ Z) \vdash_{(1)} (q_0,\ bba,\ aZ) \vdash_{(6)} (q_0,\ ba,\ baZ) \vdash_{(8)}$$
$$\vdash_{(8)} (q_1,\ a,\ aZ) \vdash_{(9)} (q_1,\ \lambda,\ Z) \vdash_{(11)} (q_2,\ \lambda,\ \lambda)$$

(справа внизу под значками непосредственной выводимости подписаны номера применяемых команд).

К начальной конфигурации $(q_0,\ abba,\ Z)$ применима только команда (1). После выполнения этой команды первый символ входной цепочки будет прочитан, а в магазине вместо одного символа Z окажется цепочка aZ, т.е. символ a станет верхним символом магазина.

К полученной конфигурации $(q_0,\ bba,\ aZ)$ применима только команда (6), в результате выполнения которой будет прочитан еще один символ входной цепочки, т.е. ее второй символ b, и в магазине окажется цепочка baZ.

К третьей конфигурации записанного выше вывода применимы две команды: (7) и (8). Выбирая команду (8), мы тем самым читаем третий символ входной цепочки, а верхний символ магазина, совпавший с указанным символом входной цепочки, „выталкиваем“ из магазина, после чего верхним символом магазина окажется уже символ a, т.е. возникнет конфигурация (q_1, a, aZ). Заметим также, что после выполнения команды (8) МП-автомат M_2 окажется в новом состоянии q_1.

Применив к конфигурации (q_1, a, aZ) единственную применимую к ней команду (9), получим конфигурацию (q_1, λ, Z), т.е. входная цепочка прочитана полностью, а в магазине остался символ Z.

Применяя опять-таки единственно возможную для такой конфигурацию команду (11), мы убираем символ Z из магазина, тем самым опустошая его, и переходим в заключительное состояние q_2.

В то же время, если бы после третьей конфигурации было выбрано „неудачное продолжение“, т.е. была бы применена команда (7), и автомат продолжал бы записывать в магазин вместо того, чтобы сравнивать магазин с лентой, то получился бы вывод

$$(q_0, ba, baZ) \vdash_{(7)} (q_0, a, bbaZ) \vdash_{(5)} (q_0, \lambda, abbaZ).$$

Последняя конфигурация в этом выводе характеризуется тем, что входная цепочка прочитана, магазин не пуст, состояние не является заключительным и не применима ни одна команда, т.е. полученная конфигурация $(q_0, \lambda, abbaZ)$ МП-автомата M_2 является тупиковой. #

Рассмотренный пример показывает, что МП-автомат может попасть в тупиковую конфигурацию, читая даже „правильную“ цепочку, т.е. цепочку, принадлежащую его языку. Это связано с недетерминированностью МП-автомата, т.е. с тем, что из данной конфигурации можно непосредственно вывести,

вообще говоря, более чем одну конфигурацию*. Тот факт, что входная цепочка принадлежит языку МП-автомата, означает лишь, что существует допускающая последовательность конфигураций для этой цепочки, но не всякая последовательность конфигураций, которая возникает при чтении этой цепочки, будет допускающей. Задача эффективного поиска допускающей последовательности конфигураций — одна из основных задач разработки алгоритмов синтаксического анализа на базе МП-автоматов.

Как уже отмечалось, основное свойство языков, допускаемых МП-автоматами, состоит в том, что эти языки образуют класс, совпадающий с классом КС-языков. Но перед тем как доказывать этот основной результат теории МП-автоматов, аналогичный теореме Клини для конечных автоматов, установим одно весьма полезное свойство МП-автомата, состоящее в том, что „хвост" входной цепочки и „дно" магазина не влияют на работу МП-автомата с началом входной цепочки и верхом магазина, так как входная цепочка читается строго символ за символом без возвратов и забеганий вперед, а магазин обновляется только сверху.

Приведем строгую формулировку.

Теорема 8.5. Пусть в МП-автомате

$$M = (Q,\ V,\ \Gamma,\ q_0,\ F,\ Z_0,\ \delta)$$

на множестве конфигураций для некоторых $q, r \in Q$, $x, y \in V^*$ и $\alpha, \beta \in \Gamma^*$ имеет место выводимость

$$(q,\ xy,\ \alpha) \vdash_M^* (r,\ y,\ \beta). \tag{8.10}$$

*МП-автомат называют ***детерминированным***, если из каждой его конфигурации непосредственно выводится не более чем одна конфигурация. Можно доказать, что, в отличие от конечного автомата, для МП-автомата невозможна процедура детерминизации и класс языков, допускаемых детерминированными МП-автоматами, строго включается в класс языков, допускаемых произвольными МП-автоматами.

Тогда для произвольных $z \in V^*$, $\gamma \in \Gamma^*$ имеет место выводимость

$$(q,\ xyz,\ \alpha\gamma) \vdash_M^* (r,\ yz,\ \beta\gamma). \tag{8.11}$$

Обратно, если для некоторых q, $r \in Q$, x, y, $z \in V^*$ и α, β, $\gamma \in \Gamma^*$ имеет место выводимость (8.11), то выполняется и выводимость (8.10).

◀ Индукцией по длине вывода, связывающего конфигурацию $(q,\ xy,\ \alpha)$ с конфигурацией $(r,\ y,\ \beta)$ в (8.10) докажем, что если эта выводимость имеет место, то имеет место и выводимость (8.11).

Для вывода нулевой длины утверждение тривиально.

Пусть доказываемое верно для любой длины вывода, связывающего конфигурации в (8.10), не превышающей $n-1$.

Предположим, что

$$(q,\ xy,\ \alpha) \vdash_M^n (r,\ y,\ \beta).$$

Рассмотрим первый шаг соответствующего вывода. Пусть он имеет вид

$$(q,\ ax'y,\ Z\alpha') \vdash_M (q',\ x'y,\ \sigma\alpha'),$$

где $x = ax'$, $\alpha = Z\alpha'$, $a \in V \cup \{\lambda\}$ и $Z \in \Gamma$.

Это значит, что на этом шаге применена команда

$$qaZ \to q'\sigma. \tag{8.12}$$

Согласно предположению индукции, для произвольных $z \in V^*$ и $\gamma \in \Gamma^*$ имеет место выводимость

$$(q',\ x'yz,\ \sigma\alpha'\gamma) \vdash^{n-1} (r,\ yz,\ \beta\gamma). \tag{8.13}$$

Рассмотрим конфигурацию $(q,\ xyz,\ \alpha\gamma) = (q,\ ax'yz,\ Z\alpha'\gamma)$ и применим к ней команду (8.12). Получим

$$(q,\ ax'yz,\ Z\alpha'\gamma) \vdash_M (q',\ x'yz,\ \sigma\alpha'\gamma).$$

Отсюда в силу (8.13), будем иметь

$$(q,\ ax'yz,\ Z\alpha'\gamma) \vdash (q',\ x'yz,\ \sigma\alpha'\gamma) \vdash_M^{n-1} (r,\ yz,\ \beta\gamma),$$

т.е. при выводимости (8.10) за n шагов имеет место и выводимость (8.11) также за n шагов, что и требовалось доказать.

Аналогично (но уже индукцией по длине вывода (8.11)) доказывается, что если имеет место выводимость (8.11), то имеет место и выводимость (8.10). ▶

МП-автомат M называют ***эквивалентным КС-грамматике*** G, если язык, допускаемый M, совпадает с ***языком, порождаемым грамматикой*** G, т.е. если $L(M) = L(G)$.

Опишем алгоритм построения по заданной КС-грамматике эквивалентного ей МП-автомата, а также алгоритм построения по заданному МП-автомату КС-грамматики, которой он эквивалентен.

Алгоритм построения МП-автомата по КС-грамматике. Для исходной КС-грамматики $G = (V,\ N,\ S,\ P)$ определим МП-автомат $M_G = (\{q\},\ V,\ V \cup N,\ q,\ \{q\},\ S,\ \delta)$ (с единственным состоянием q), система команд δ которого строится следующим образом: для каждого правила $A \to \alpha$, принадлежащего P, в δ записывается команда $q\lambda A \to q\alpha$ и для каждого $a \in V$ — команда $qaa \to q\lambda$. Никаких других команд в системе команд δ нет.

Пример 8.16. Пусть дана грамматика

$$G = (\{a, b, c\},\ \{S\},\ S,\ P),$$

множество правил вывода P которой есть

$$S \to aSbS \,|\, aS \,|\, c.$$

Тогда эквивалентный ей МП-автомат задается следующей системой команд:

$$q\lambda S \to qaSbS \,|\, qaS \,|\, qc,$$
$$qaa \to q\lambda,$$
$$qbb \to q\lambda,$$
$$qcc \to q\lambda$$

(мы использовали сокращенную запись нескольких команд с одинаковыми левыми частями по аналогии с тем, как мы это делаем при записи множества правил вывода грамматики).

В системе команд МП-автомата, который строится по заданной КС-грамматике так, как это описано выше, мы видим два „сорта“ команд: 1) команды, „моделирующие“ применение правил, исходной грамматики (в данном примере это команды первой строки); при выполнении каждой такой команды МП-автомат делает λ-такт, т.е. не продвигает головку по входной ленте; 2) команды „сравнения“, согласно которым МП-автомат должен убирать верхний символ магазина всякий раз, когда он совпадает с обозреваемым символом на ленте.

Тогда, читая входную цепочку, такой МП-автомат „моделирует“ левый вывод этой цепочки в исходной грамматике, применяя каждый раз, когда верхним символом магазина оказывается нетерминал, команду „первого сорта“ и всякий раз, когда верхним символом магазина оказывается терминал исходной грамматики, „команду сравнения“.

Прочитаем цепочку $aacbc$:

$$(q,\ aacbc,\ S) \vdash (q,\ aacbc,\ aSbS) \vdash (q,\ acbc,\ SbS) \vdash$$
$$\vdash (q,\ acbc,\ aSbS) \vdash (q,\ cbc,\ SbS) \vdash (q,\ cbc,\ cbS) \vdash$$
$$\vdash (q,\ bc,\ bS) \vdash (q,\ c,\ S) \vdash (q,\ c,\ c) \vdash (q,\ \lambda,\ \lambda).$$

Нетрудно видеть, что этот вывод „моделирует“ левый вывод в грамматике:

$$S \vdash aSbS \vdash aaSbS \vdash aacbS \vdash aacbc,$$

т.е. МП-автомат, строя допускающую последовательность конфигураций для входной цепочки, на каждом шаге, когда верхний символ магазина не совпадает с очередным символом входной цепочки, должен „угадать“ применяемое правило (выбрать в данном примере одну из трех первых команд). Неправильный выбор приведет к тупику, например:

$$(q,\ aacbc,\ S) \vdash (q,\ aacbc,\ aS) \vdash (q,\ acbc,\ S) \vdash (q,\ acbc, c). \quad \#$$

Докажем, что рассмотренный алгоритм дает МП-автомат, эквивалентный исходной КС-грамматике.

Теорема 8.6. МП-автомат M_G эквивалентен КС-грамматике G.

◀ Индукцией по длине n вывода *терминальной цепочки* x из *нетерминала* A докажем, что если $A \vdash_G^* x$, то

$$(q,\ x,\ A) \vdash_{M_G}^* (q,\ \lambda,\ \lambda).$$

Пусть $n = 1$, т.е. $A \vdash^1 x$; тогда в P есть правило $A \to x$ и, следовательно, в δ есть команда $q\lambda A \to qx$. В таком случае при $x = \lambda$ имеет место выводимость

$$(q,\ \lambda,\ A) \vdash (q,\ \lambda,\ \lambda),$$

и требуемый вывод на множестве конфигураций МП-автомата M_G построен. Если же цепочка x непустая, то тогда, расписывая ее по символам, т.е. полагая $x = x(1)\dots x(k)$, $k \geqslant 1$, получим

$$(q,\ x(1)\dots x(k),\ A) \vdash (q,\ x(1)\dots x(k),\ x(1)\dots x(k)).$$

Из последней конфигурации за k шагов посредством применения команд вида $qaa \to q\lambda$, где $a \in V$, выводится конфигурация $(q,\ \lambda,\ \lambda)$, и, таким образом, $(q,\ x,\ A) \vdash^{|x|+1} (q,\ \lambda,\ \lambda)$.

Пусть теперь доказываемое верно для любого n, не превосходящего некоторого $m - 1$ для $m \geqslant 2$, пусть $A \vdash^m x$ и первый

шаг вывода длины m цепочки x из нетерминала A имеет вид $A \vdash X_1X_2\ldots X_k$, где $k \geqslant 1$ и для каждого $i = \overline{1, k}$ символ X_i есть символ объединенного алфавита грамматики*. Далее, из цепочки $X_1X_2\ldots X_k$ должна быть выведена терминальная цепочка x. Это значит, что для каждого $i = \overline{1, k}$ из символа X_i выводится какая-то подцепочка цепочки x (в частности, если этот символ является терминалом, он будет одним из символов цепочки x). Таким образом, для каждого $i = \overline{1, k}$ выполняется $X_i \vdash^{m_i} x_i$ $(m_i \geqslant 0)$, и $x = x_1x_2\ldots x_k$.

Для $X_i \in N$ длина вывода m_i подцепочки x_i не может превышать $m - 1$. Следовательно, согласно предположению индукции, имеем

$$(q,\ x_i,\ X_i) \vdash^* (q,\ \lambda,\ \lambda),$$

а для $X_i \in V$ $(m_i = 0$ и, следовательно, $X_i = x_i)$, согласно построению МП-автомата M_G,

$$(q,\ x_i,\ x_i) \vdash^* (q,\ \lambda,\ \lambda).$$

Тогда в силу теоремы 8.5

$$(q,\ x_1x_2\ldots x_k,\ X_1X_2\ldots X_k) \vdash^* (q,\ x_2\ldots x_k,\ X_2\ldots X_k) \vdash^* \\ \vdash^* (q,\ x_k,\ X_k) \vdash (q,\ \lambda,\ \lambda).$$

Кроме того, так как $A \vdash X_1X_2\ldots X_k$, то в P есть правило вывода $A \to X_1X_2\ldots X_k$, откуда, согласно построению МП-автомата M_G, в δ есть команда

$$q\lambda A \to qX_1X_2\ldots X_k$$

и

$$(q,\ x,\ A) \vdash (q,\ x,\ X_1X_2\ldots X_k),$$

*Цепочка $X_1X_2\ldots X_k$ не может быть пустой, так как тогда она окажется последней цепочкой рассматриваемого вывода и его длина будет равна 1, а мы предположили, что его длина равна $m > 1$.

а окончательно

$$(q,\ x,\ A) \vdash^* (q,\ \lambda,\ \lambda).$$

Следовательно, если $x \in L(G)$, то $S \vdash^* x$ и $(q,\ x,\ S) \vdash^* (q,\ \lambda,\ \lambda)$, т.е. $x \in L(M_G)$.

Итак, мы доказали, что $L(G) \subseteq L(M_G)$.

Чтобы доказать обратное включение, сначала индукцией по длине n вывода на множестве конфигураций МП-автомата M_G докажем, что

$$(q,\ x,\ A) \vdash^*_{M_G} (q,\ \lambda,\ \lambda)$$

влечет $A \vdash^*_G x$ (для произвольных $A \in N$ и $x \in V^*$).

Пусть $n = 1$, т.е.

$$(q,\ x,\ A) \vdash (q,\ \lambda,\ \lambda).$$

Согласно построению МП-автомата M_G, это может быть тогда и только тогда, когда $x \in V$ и $A = x$, т.е. $x \vdash^* x$.

Пусть доказываемое верно для любого $n \leqslant m-1$, где $m \leqslant 2$, и

$$(q,\ x,\ A) \vdash^m (q,\ \lambda,\ \lambda), \tag{8.14}$$

причем первый шаг соответствующего вывода имеет вид

$$(q,\ x,\ A) \vdash (q,\ x,\ X_1X_2\ldots X_k). \tag{8.15}$$

Это значит, что в системе команд δ есть команда $q\lambda A \to$ $\to qX_1X_2X_k$ и, следовательно, правило в множестве правил вывода P грамматики G есть правило $A \to X_1X_2\ldots X_k$, по которому указанная команда построена. Из (8.14) и (8.15) следует, что имеет место выводимость

$$(q,\ x,\ X_1X_2\ldots X_k) \vdash^* (q,\ \lambda,\ \lambda). \tag{8.16}$$

Используя индукцию по длине магазинной цепочки $X_1X_2\ldots X_k$, можно доказать, что из (8.16) следует существование таких

входных цепочек $x_1, x_2, \ldots, x_k$, что $x = x_1 x_2 \ldots x_k$ и имеет место выводимость

$$\begin{aligned}(q,\, x_1x_2\ldots x_k,\, X_1X_2\ldots X_k) &\vdash^{m_1} \\ &\vdash^{m_1} (q,\, x_2\ldots x_k,\, X_2\ldots X_k) \vdash^{m_2} \\ &\vdash^{m_2} (q, x_3\ldots x_k, X_3\ldots X_k) \vdash^{m_3} \ldots \vdash^{m_{k-1}} \\ &\vdash^{m_{k-1}} (q, x_k, X_k) \vdash^{m_k} (q, \lambda, \lambda) \qquad (8.17)\end{aligned}$$

для некоторых m_i, таких, что $1 \leqslant m_i < m-1$, $i = \overline{1,k}$.

Вывод (8.17) можно прокомментировать так: чтобы достичь заключительной конфигурации, M_G должен выбросить все символы $X_1, X_2, \ldots, X_k$ из магазина. Предположим, что к тому моменту, когда X_1 покинет магазин (чтобы уже больше туда не вернуться!), будет прочитано некоторое *начало* входной *цепочки* x — цепочка x_1, когда X_2 покинет магазин, будет прочитана следующая подцепочка x_2 и так до тех, пока наконец, автомат не прочитает цепочку x_k — *конец цепочки* x.

Из теоремы 8.5 следует, что существуют также выводы

$$\begin{aligned}&(q,\, x_1,\, X_1) \vdash^{m_1} q;\lambda;\lambda, \\ &(q,\, x_2,\, X_2) \vdash^{m_2} (q,\, \lambda, \lambda), \\ &\ldots\ldots\ldots\ldots\ldots \\ &(q,\, x_k,\, X_k) \vdash^{m_k} (q,\, \lambda,\, \lambda).\end{aligned} \qquad (8.18)$$

Все числа m_i при $i = \overline{1,k}$ не больше $m-1$. Тогда согласно предположению индукции, из (8.18) следует, что для каждого $i = \overline{1,k}$ в грамматике G имеет место $X_i \vdash_G^* x_i$, и в силу того, что в P есть правило вывода $A \to X_1X_2\ldots X_k$ (см. (8.15)),

$$A \vdash X_1X_2\ldots X_k \vdash^* x_1x_2\ldots x_k = x,$$

т.е. $A \vdash_G^* x$, что и требовалось доказать.

Отсюда, если $x \in L(M_G)$, то $x \in L(G)$, т.е. $L(M_G) \subseteq L(G)$, и в силу доказанного выше языки $L(G)$ и $L(M_G)$ совпадают. ▶

Алгоритм построения КС-грамматики по МП-автомату. Дадим сначала неформальную мотивировку той конструкции, которая будет приведена ниже. Будем рассматри-

вать МП-автомат как „игрока“, ставящего цели следующего вида: „находясь в состоянии q и имея верхний символ магазина Z, перейти в состояние s“. Условимся записывать такую цель в виде тройки $[qZs]$. Как может наш „игрок“ достичь поставленной цели? Если в множестве команд автомата („правил игры“) есть команда

$$qaZ \to rX_1X_2\ldots X_k,$$

где для каждого i X_i — магазинный символ ($X_i \in \Gamma$), то после выполнения этой команды МП-автомат перейдет в состояние r.

Если $r = s$, то цель достигнута, иначе ставим цель $[rX_1s_1]$, а достигнув ее, ставим цель $[s_1X_2s_2]$ и т.д. Достигнув цели $[s_{k-1}X_ks]$, „игрок“ достигает и цели $[qZs]$. Так как рассматривается допуск с пустым магазином, то символы X_i должны по очереди покинуть магазин, и только в этом случае может быть достигнута „глобальная“ цель МП-автомата: $[q_0Z_0q_f]$, $q_f \in F$ („находясь в начальном состоянии и имея верхним символом магазина начальный магазинный символ, попасть в одно из заключительных состояний, опустошив магазин“). Так как, вообще говоря, „игрок“ не знает наперед последовательности состояний $s_1, s_2, \ldots, s_{k-1}$, ведущих к цели, он должен перебрать все такие последовательности.

Эти неформальные соображения лежат в основе следующей конструкции.

Пусть дан МП-автомат

$$M = (Q, V, \Gamma, q_0, F, Z_0, \delta).$$

Определим КС-грамматику $G_M = (V, N, S, P)$, терминальный алфавит которой совпадает со входным алфавитом МП-автомата M, следующим образом.

Нетерминальный алфавит N грамматики есть множество, находящееся во взаимно однозначном соответствии с множеством всех упорядоченных троек вида (q, Z, s), где $q, s \in Q$, $Z \in \Gamma$, пополненное символом S, не принадлежащим ни одному из множеств Q, V, Γ и объявляемым аксиомой грамматики.

Упорядоченные тройки указанного вида записывают обычно как $[qZs]$, интуитивно понимая каждую такую тройку как охарактеризованную выше цель.

Таким образом, $N = \{[qZs]\colon q, s \in Q \text{ и } Z \in \Gamma\} \cup \{S\}$.

Множество правил вывода P грамматики G_M строится так:

а) если команда $qaZ \to rX_1X_2\ldots X_k$, $k \geqslant 1$, принадлежит системе команд δ, то в P записываются все правила вида

$$[qZs_k] \to a[rX_1s_1][s_1X_2s_2]\ldots[s_{k-1}X_ks_k]$$

для любой последовательности k состояний $s_1, \ldots, s_k$ множества Q (тем самым к P добавляется $|Q|^k$ правил на каждую команду указанного вида);

б) для каждой команды $qaZ \to r\lambda$ в δ в P добавляется правило $[qZr] \to a$;

в) для каждого $q_f \in F$ в P вводится правило $S \to [q_0Z_0q_f]$;

г) никаких других правил в P, кроме определенных пп. а–в, нет.

Пример 8.17. Для МП-автомата

$$M = (\{q_0, q_1, q_2\},\ \{a, b\},\ \{a, b, Z\},\ q_0,\ \{q_0\},\ Z,\ \delta)$$

с множеством команд δ, имеющих вид

$$q_0aZ \to q_1aZ,$$
$$q_1aa \to q_1aa,$$
$$q_1ba \to q_2\lambda,$$
$$q_2ba \to q_2\lambda,$$
$$q_2\lambda Z \to q_0\lambda,$$
$$q_0\lambda Z \to q_0\lambda,$$

построим эквивалентную ему КС-грамматику. Можно доказать, что этот МП-автомат допускает язык $\{a^nb^n\colon n \geqslant 0\}$. Заметим, что МП-автомат M допускает пустую цепочку, применяя к начальной конфигурации (q_0, λ, Z) последнюю коман-

ду. Построенная по данной системе команд грамматика будет иметь следующий вид:

$$
\begin{aligned}
&S \to [q_0 Z q_0],\\
&[q_0 Z s_2] \to a[q_1 a s_1][s_1 Z s_2],\ s_1, s_2 \in \{q_0, q_1, q_2\},\\
&[q_1 a s_2] \to a[q_1 a s_1][s_1 a s_2],\ s_1, s_2 \in \{q_0, q_1, q_2\},\\
&[q_1 a q_2] \to b,\\
&[q_2 a q_2] \to b,\\
&[q_2 Z q_0] \to \lambda,\\
&[q_0 Z q_0] \to \lambda.
\end{aligned}
$$

Подчеркнем, что во второй и третьей строчках имеется не по одному, а по $3^2 = 9$ правил (число всех последовательностей двух состояний из трехэлементного множества состояний). Выведем в этой грамматике цепочку $aabb$:

$$
\begin{aligned}
S \vdash [q_0 Z q_0] \vdash a[q_1 a q_2][q_2 Z q_0] \vdash aa[q_1 a q_2][q_2 a q_2][q_2 Z q_0] \vdash\\
\vdash aab[q_2 a q_2][q_2 Z q_0] \vdash aabb[q_2 Z q_0] \vdash aabb.
\end{aligned}
$$

На втором шаге этого вывода мы применяем то правило вывода из девяти правил второй строки, которое получается при подстановке вместо s_1 состояния q_2, а вместо s_2 состояния q_0. Мы „угадываем“ эти состояния, зная (по системе команд исходного МП-автомата), что достичь заключительного состояния по прочтении (непустой) входной цепочки наш МП-автомат может только из состояния q_2. В то же время, если бы мы на втором шаге использовали правило второй строки, получающееся подстановкой $s_1 = q_1$, $s_2 = q_2$, возник бы *бесполезный нетерминал* $[q_1 a q_1]$ и наш вывод зашел бы в тупик.

Аналогично на третьем шаге используется то правило из девяти правил третьей строки, в котором $s_1 = s_2 = q_2$. После этого применяем по очереди правила четвертой, пятой и шестой строк, завершая вывод.

Если мы теперь в построенном выводе „считаем“ по шагам магазинные символы, заключенные в квадратных скобках

между состояниями, то получим Z, aZ, aaZ, aZ, Z, т.е. получим изменение содержимого магазина (не считая последнего шага, когда происходит его окончательное опустошение), представленное в следующем выводе на множестве конфигураций МП-автомата M:

$$(q_0,\ aabb,\ Z) \vdash (q_1,\ abb,\ aZ) \vdash (q_1,\ bb,\ aaZ) \vdash$$
$$\vdash (q_2,\ b,\ aZ) \vdash (q_2,\ \lambda,\ Z) \vdash (q_0,\ \lambda,\ \lambda).$$

Этот вывод есть не что иное, как допускающая последовательность конфигураций для цепочки $aabb$.

Читая же последовательности состояний в квадратных скобках, мы получим в итоге ту последовательность состояний, которую проходит МП-автомат, допуская написанную выше цепочку.

Действительно, после первого шага вывода в грамматике получим последовательность q_0, q_0, что можно интерпретировать так: „из состояния q_0 перейти (вернуться) в это же состояние q_0, прочитав входную цепочку“.

После второго шага будем иметь q_0, q_1, q_2, q_0, что означает: „чтобы вернуться в q_0, сначала нужно попасть в q_2 через q_1“).

После третьего шага получим q_0, q_1, q_1, q_2, q_2, q_0. Это и есть результирующая последовательность состояний, так как все следующие шаги МП-автомата связаны с „выталкиванием“ символов из магазина и не приводят к возникновению новых целей. #

Как правило, грамматика, которая указанным выше способом строится по МП-автомату, оказывается очень громоздкой, содержит много бесполезных и *недостижимых символов*. Это связано с тем, что в ней фигурируют произвольные последовательности состояний МП-автомата фиксированной длины. Что касается разобранного примера 8.17, то в этом случае легко написать грамматику для языка $\{a^n b^n\colon n \geqslant 0\}$ непосредственно:

$$S \to aSb \,|\, ab \,|\, \lambda.$$

По этой грамматике можно построить МП-автомат, используя алгоритм из первой части доказательства основной теоремы, значительно более простой, чем исходный. Его система команд будет иметь следующий вид:

$$qaS \to qSb \,|\, qb,$$
$$qaa \to q\lambda,$$
$$qbb \to q\lambda,$$
$$q\lambda S \to q\lambda.$$

В общем же случае проблема распознавания эквивалентности двух МП-автоматов (в отличие от такой же проблемы для конечных автоматов) не разрешима, и не существует общего алгоритма „упрощения“ (в определенном смысле, „минимизации“) МП-автомата, хотя, как мы только что видели, в конкретных случаях это вполне возможно.

Теорема 8.7. МП-автомат M эквивалентен грамматике G_M.

◀ Индукцией по длине n вывода в M докажем, что $Z \in \Gamma$, $(q,\ x,\ Z) \vdash^n (r,\ \lambda,\ \lambda)$ для любых $q, r \in Q$, $x \in V^*$ влечет $[qZr] \vdash^* x$ в G_M.

Если $n = 1$, т.е. $(q,\ x,\ Z) \vdash (r,\ \lambda,\ \lambda)$, то $x \in V \cup \{\lambda\}$, и в δ есть команда $qZx \to r\lambda$, откуда в P есть правило $[qZr] \to x$, и $[qZr] \vdash^* x$.

Пусть доказываемое верно для каждого $n \leqslant m-1$, где $m > 1$, и пусть $(q,\ x,\ Z) \vdash^m (r,\ \lambda,\ \lambda)$, причем первый шаг соответствующего вывода имеет вид

$$(q,\ x,\ Z) \vdash (p,\ y,\ X_1X_2\ldots X_k),$$

где $x = ay$ для некоторого $a \in V \cup \{\lambda\}$.

Аналогично доказательству теоремы 8.6 (см. (8.17)) доказывается, что тогда найдутся такие цепочки $x_1x_2\ldots x_k$ и такая

последовательность состояний $s_1, \ldots, s_{k-1}$, что $y = x_1x_2\ldots x_k$ и

$$(p,\ x_1x_2\ldots x_k,\ X_1X_2\ldots X_k) \vdash^{m_1} (s_1,\ x_2\ldots x_k,\ X_2\ldots X_k) \vdash^{m_2} \\ \vdash^{m_2} \ldots \vdash^{m_{k-1}} (s_{k-1},\ x_k,\ X_k) \vdash^{m_k} (r,\ \lambda,\ \lambda),$$

где для любого $i = \overline{1, k}$ выполняется $0 \leqslant m_i \leqslant m-1$. Поэтому в силу теоремы 8.5 для любого $i = \overline{1, k-1}$

$$(s_{i-1},\ x_i,\ X_i) \vdash^{m_i} (s_i,\ \lambda,\ \lambda),$$

где $s_0 = p$, а $s_k = r$, и, согласно предположению индукции,

$$[s_{i-1}X_is_i] \vdash^* x_i.$$

Следовательно, согласно построению грамматики G_M, имеет место выводимость

$$[qZr] \vdash_{G_M} a[pX_is_i][s_iX_2s_2]\ldots[s_{k-1}X_kr] \vdash^*_{G_M} \\ \vdash^*_{G_M} ax_1x_2\ldots x_k = ay = x,$$

что и требовалось доказать.

Пусть цепочка x допускается МП-автоматом M. Тогда

$$(q_0,\ x,\ Z_0) \vdash^*_M (q_f,\ \lambda,\ \lambda),$$

где $q_f \in F$ — одно из заключительных состояний МП-автомата M. Согласно только что доказанному, в этом случае для грамматики G_M выполняется $[q_0Z_0q_f] \vdash^*_{G_M} x$. Но так как в множестве правил вывода грамматики G_M есть правило $S \to \\ \to [q_0Z_0q_f]$, то мы получим $S \vdash_{G_M} [q_0Z_0q_f] \vdash^*_{G_M} x$, т.е. $x \in L(G_M)$. Итак, $L(M) \subseteq L(G_M)$.

Для доказательства обратного включения докажем сначала, что $[qZr] \vdash^*_{G_M} x$ влечет $(q,\ x,\ Z) \vdash^*_M (r,\ \lambda,\ \lambda)$ для любых $q, r \in Q$, $x \in V^*$ и $Z \in \Gamma$. Снова проведем индукцию по длине вывода (в грамматике G_M). При $[qZr] \vdash x$ получаем правило $[qZr] \to x$ в P и, следовательно, команду $qxZ \to r\lambda$ в δ, т.е. $(q,\ x,\ Z) \vdash (r,\ \lambda,\ \lambda)$.

Если же $[qZr] \vdash^m x$ для некоторого $m \geqslant 1$, то $x = ay$ для некоторого $a \in V \cup \{\lambda\}$ и

$$[qZr] \vdash a[pX_1s_1][s_1X_2s_2]\dots[s_{k-1}X_kr],$$

причем для всех $i = \overline{1,k}$

$$[s_{i-1}X_is_i] \vdash^{m_i} x_i,$$

где $s_0 = p$, $s_k = r$ и $1 \leqslant m_i \leqslant m-1$, так что $y = x_1x_2\dots x_k$.

Согласно предположению индукции, тогда для каждого такого i

$$(s_{i-1}, x_i, X_i) \vdash^* (s_i, \lambda, \lambda).$$

Но так как указанный выше первый шаг вывода в грамматике возможен только при наличии команды в МП-автомате $qaZ \to$ $\to pX_1X_2\dots X_k$, то

$$\begin{aligned}(q,\, x,\, Z) \vdash (p,\, y,\, X_1X_2\dots X_k) \vdash^* \\ \vdash^* (s_1,\, x_2\dots x_k,\, X_2\dots X_k) \vdash^* \\ \vdash^* (s_{k-1},\, x_k,\, X_k) \vdash^* (r,\, \lambda,\, \lambda).\end{aligned}$$

Если теперь цепочка x порождается грамматикой G, т.е. $S \vdash^*_{G_M} x$, то первый шаг вывода x из S, согласно определению системы правил грамматики G_M, будет иметь вид $S \vdash [q_0Z_0q_f]$ для некоторого $q_f \in F$, и, следовательно, $[q_0Z_0q_f] \vdash^*_{G_M} x$. Тогда в силу только что доказанного $(q_0,\, x,\, Z_0) \vdash^*_M (q_f,\, \lambda,\, \lambda)$, т.е. $x \in$ $\in L(M)$. Итак, $L(G_M) \subseteq L(M)$, а поскольку обратное включение уже доказано, то $L(M) = L(G_M)$ ▶

Из доказанных теорем 8.6 и 8.7 получаем следующую теорему.

Теорема 8.8. Язык является контекстно-свободным тогда и только тогда, когда он допускается некоторым МП-автоматом.

Замечание 8.10. Существует одна полезная модификация построения МП-автомата по КС-грамматике. Вернемся

к примеру 8.16. Можно заметить, что в этом примере правая часть каждого правила грамматики начинается некоторым терминалом. Учет этой особенности позволяет найти другой МП-автомат, который, как нетрудно показать, тоже эквивалентен данной грамматике. Система команд этого автомата имеет следующий вид:

$$qaS \to qSbS \mid qS,$$
$$qcS \to q\lambda,$$
$$qaa \to q\lambda,$$
$$qbb \to q\lambda,$$
$$qcc \to q\lambda.$$

Его преимущество в том, что он „видит“ первый непрочитанный символ входной цепочки и, следовательно, имеет меньше альтернатив при выборе команды: например, если очередной символ есть b, то ни одна из команд первых двух строк не может быть применена. Тогда этот автомат имеет меньше возможностей попасть в тупик.

В общем случае, если правая часть любого правила грамматики имеет вид $a\xi$, где $a \in V$, МП-автомат, эквивалентный данной грамматике, определяется командами вида

$$qaA \to q\xi,$$
$$qbb \to q\lambda, \quad b \in V,$$

(первая — для правила $A \to a\xi$). В такой модификации МП-автомат записывает в магазин „хвост“ правой части правила, следующей за первым терминалом.

Можно доказать, что любая КС-грамматика может быть определена правилами такого вида (так называемая нормальная форма Грейбах*).

*См.: *Ахо А., Ульман Дж.*

8.5. Алгебраические свойства КС-языков

В этом параграфе мы рассмотрим некоторые *операции* над *КС-языками*, относительно которых *множество* всех КС-языков *замкнуто*. Напомним (см. **7.6**), что множество всех *регулярных языков* (в произвольно фиксированном *алфавите*) замкнуто относительно операций (конечного) *объединения*, *соединения*, *пересечения*, *дополнения* (до *универсального языка*) и *итерации*. В этом разделе мы докажем прежде всего что множество КС-языков замкнуто относительно операций объединения, соединения и итерации. Потом мы увидим, что пересечение двух КС-языков, вообще говоря, не является КС-языком, но можно доказать более слабое утверждение: пересечение КС-языка с регулярным языком есть КС-язык.

Начнем с определения новой операции над языками — операции суперпозиции.

Пусть $V = \{a_1, \ldots, a_n\}$ — произвольный алфавит, каждой букве a_i которого сопоставлен однозначно некоторый алфавит V_{a_i} (не исключено, что все алфавиты V, V_{a_1}, ..., V_{a_n} совпадают); в алфавите V фиксируем произвольно язык $L \subseteq V^*$, а в каждом из алфавитов V_{a_i} — язык $L_{a_i} \subseteq V_{a_i}^*$.

Рассмотрим множество всех таких цепочек в объединении алфавитов $V_{a_1} \cup \ldots \cup V_{a_n}$, которые могут быть получены из цепочек языка L следующим образом: берем произвольную цепочку $b_1 \ldots b_k \in L$ и каждую букву b_j в ней заменяем произвольной цепочкой $x_j \in L_{b_j}$:

$$\begin{array}{cccc} b_1 & b_2 & \ldots & b_k \\ \downarrow & \downarrow & & \downarrow \\ x_1 & x_2 & \ldots & x_k, \end{array}$$

в результате чего получаем цепочку $x_1 x_2 \ldots x_k$.

Множество всех цепочек вида $x_1 x_2 \ldots x_k$, получаемых описанным выше способом, называют ***суперпозицией*** (или ***подстановкой***) ***языков*** L_{a_1}, ..., L_{a_n} ***в язык*** L и обозначают $S(L, L_{a_1}, \ldots, L_{a_n})$. При этом полагают, что если пустая цепоч-

ка λ принадлежит языку L, то она принадлежит суперпозиции $S(L, L_{a_1}, \ldots, L_{a_n})$.

При определении суперпозиции на алфавиты V, V_{a_1}, ..., V_{a_n} не накладывается никаких ограничений. Они могут и попарно не пересекаться, а могут и все совпадать. В любом случае суперпозиция $S(L, L_{a_1}, \ldots, L_{a_n})$ есть язык в алфавите $V_{a_1} \cup \ldots \cup V_{a_n}$. Точно так же среди языков L_{a_i}, „подставляемых“ в язык L, может оказаться и сам язык L, т.е. язык можно подставлять сам в себя, причем многократно.

Заметим, что если язык L не содержит пустую цепочку, суперпозиция $S(L, L_{a_1}, \ldots, L_{a_n})$ может ее содержать. Это возможно, если каждый язык L_{a_i} содержит пустую цепочку. Тогда, беря произвольно $x \in L$ и заменяя каждый ее символ пустой цепочкой, получаем пустую цепочку как элемент суперпозиции. Если же пустая цепочка принадлежит L, то, по определению, суперпозиция обязана ее содержать.

Еще заметим, что для разных вхождений одной и той же буквы b_j цепочки $x \in L$ совершенно не обязательно заменять эту букву одной и той же цепочкой языка L_{b_j}: каждому вхождению может быть сопоставлена своя цепочка этого языка.

Пример 8.18. Зададим язык L в алфавите $V = \{a, b, c\}$ как множество $\{a^n b^n c^n : n \geqslant 1\}$. Букве a алфавита V сопоставим алфавит $V_a = \{0, 1\}$, выбрав в качестве языка L_a множество $\{0^n 1^n\} n \geqslant 0$. Букве $b \in V$ сопоставим алфавит $V_b = \{a\}$ и в этом алфавите — язык $L_b = \{a^{n^2} : n \geqslant 0\}$. Наконец, букве c алфавита V сопоставим алфавит $V_c = \{a, b, c\} = V$ и в качестве языка L_c возьмем язык L, пополненный пустой цепочкой, т.е. $L_c = \{a^n b^n c^n : n \geqslant 0\}$.

Возьмем цепочку $aabbcc \in L$ и первый символ a заменим цепочкой $0011 \in L_a$, второй символ a — цепочкой $01 \in L_a$, первый символ b — цепочкой $a^{729} \in L_b$ (при $n = 27$), второй символ b — пустой цепочкой, первый символ c — пустой цепочкой, второй символ — цепочкой $abc \in L_c$. В результате получим следующую цепочку, принадлежащую суперпозиции

$S(L, L_a, L_b, L_c)$:

$$001101a^{729}abc = 001101 \underbrace{aa\ldots a}_{729 \text{ раз}} abc.$$

Проделывая аналогичные действия с любой другой цепочкой языка L или же производя другие замены символов той же самой цепочки, будем получать все новые цепочки определенной выше суперпозиции. В данном случае будем иметь $\lambda \in S(L, L_a, L_b, L_c)$, так как каждый из языков L_a, L_b и L_c содержит пустую цепочку.

Пример 8.19. Пусть алфавит V есть русский алфавит $\{А, Б, \ldots, Э, Ю, Я\}$, а язык L — множество всех слов русского языка. Каждой букве русского алфавита сопоставим язык, состоящий из единственной однобуквенной цепочки α, которая стоит рядом с указанной русской буквой на клавиатуре компьютера. Так, $L_А = \{F\}$, $L_Б = \{<\}$ и т.д. Все эти языки можно считать языками в латинском алфавите, дополненном определенными специальными символами, такими, как [,], {, }, <, > и т.п. Тогда, если мы в качестве исходной цепочки языка L возьмем слово *УЧЕБНИК*, то получим такую цепочку в суперпозиции $S(L, L_А, \ldots, L_Я)$: $EXT < YBR$. #

Разобранные примеры показывают, что суперпозицию можно рассматривать как своего рода „кодирование“ цепочек языка L с заменой в них каждой буквы цепочкой другого языка. Действительно, такая замена является одним из простейших способов „шифровки“ текстов.

Замечание 8.11. Суперпозицию можно определить через *соответствие*. Введем соответствие $\sigma \subseteq V \times (L_{a_1} \cup \ldots \cup L_{a_n})$ так, что для каждого $i = \overline{1, n}$ упорядоченная пара (a_i, y) принадлежит σ тогда и только тогда, когда $y \in L_{a_i}$. В этом случае

$$S(L, L_{a_1}, \ldots, L_{a_n}) = \bigcup_{x \in L \setminus \{\lambda\}} \sigma(x(1)) \ldots \sigma(x(k)) \cup (\{\lambda\} \cap L),$$

где $x = x(1) \ldots x(k)$. #

В терминах суперпозиции могут быть представлены и такие операции над языками как объединение, соединение и итерация.

Объединение. Пусть L_1 и L_2 — два произвольных языка в каком-то алфавите W. Возьмем произвольный двухбуквенный алфавит $V = \{a, b\}$ и язык L зададим как $L = V$ (т.е. язык L состоит из двух однобуквенных цепочек a и b). Рассмотрим суперпозицию $S(L, L_a, L_b)$, где $L_a = L_1$, $L_b = L_2$. Докажем, что эта суперпозиция совпадает с объединением $L_1 \cup L_2$. Действительно, если мы возьмем цепочку $a \in L$, то ее единственную букву мы можем, согласно определению суперпозиции, заменить любой цепочкой языка L_1. Точно так же, строя суперпозицию, вместо цепочки $b \in L$ получим все цепочки языка L_2. Итак,

$$L_1 \cup L_2 = S(\{a, b\}, L_1, L_2).$$

Соединение. Снова фиксируем произвольно в произвольном алфавите W языки L_1 и L_2. Опять-таки произвольно выберем двухбуквенный алфавит $\{a, b\}$, определив язык $L = \{ab\}$. Положим, как и выше, $L_a = L_1$, $L_b = L_2$. Тогда суперпозиция $S(L, L_a, L_b) = S(\{ab\}, L_a, L_b)$ совпадет с соединением $L_1 L_2$. В самом деле, символ a единственной цепочки ab языка L можно при построении суперпозиции заменять любой цепочкой языка L_1, а символ b — любой цепочкой языка L_2. В результате записанная выше суперпозиция совпадет с множеством всех цепочек, представимых как соединение xy, где $x \in L_1$, $y \in L_2$. Но это и есть соединение $L_1 L_2$. Итак,

$$L_1 L_2 = S(\{ab\}, L_1, L_2).$$

Итерация. Рассуждая аналогично, можно показать, что итерация L^* языка L равна

$$L^* = S(a^*, L),$$

а позитивная итерация

$$L^+ = S(a^+, L)$$

(для любых алфавита W, языка $L \subseteq W^*$ и однобуквенного алфавита $V = \{a\}$).

Основное свойство суперпозиции применительно к КС-языкам выражает следующая важная теорема.

Теорема 8.9. Если языки L_{a_i}, $i = \overline{1, n}$, контекстно-свободные, язык L также контекстно-свободный, то суперпозиция $S(L, L_{a_1}, \ldots, L_{a_n})$ есть КС-язык.

◀ Пусть $V = \{a_1, \ldots, a_n\}$ и каждый язык L_{a_i}, $i = \overline{1, n}$, задается КС-грамматикой $G_{a_i} = (V_{a_i}, N_{a_i}, S_{a_i}, P_{a_i})$, а язык L задается КС-грамматикой $G = (V, N, S, P)$. Без ограничения общности можно предполагать, что нетерминальные алфавиты всех указанных грамматик попарно не пересекаются (так как всегда можно провести *процедуру переименования нетерминалов* любой *грамматики*).

Построим следующую грамматику:

$$G' = (V', N', S, P'),$$

где $V' = V_{a_1} \cup \ldots \cup V_{a_n}$, $N' = N \cup N_{a_1} \cup \ldots \cup N_{a_n}$, $P' = P_{a_1} \cup \ldots \cup P_{a_n} \cup \{A \to \alpha'$: для любого правила $A \to \alpha$ в $P\}$.

Здесь цепочка α' получается из цепочки α следующим образом: все нетерминалы в α остаются без изменений, а каждое вхождение каждого терминала $b \in V$, т.е. $b = a_i$ для некоторого $i = \overline{1, n}$, заменяется аксиомой S_b грамматики G_b (заметим, что, согласно этим правилам замены, для любых цепочек α, β в объединенном алфавите грамматики G $(\alpha\beta)' = \alpha'\beta'$ и $\lambda' = \lambda$).

Неформально система правил P' получается объединением всех множеств правил грамматик G_{a_i} и добавлением „перекрашенных“ правил грамматики G. „Перекрашивание“ выражается в том, что каждый терминал a_i грамматики G заменяется

аксиомой S_{a_i} грамматики G_{a_i}, а нетерминалы остаются в неприкосновенности. В частности, для любой цепочки $\beta \in N^*$ имеет место $\beta' = \beta$.

Докажем теперь, что грамматика G', контекстно-свободная по построению, является искомой, т.е. она порождает суперпозицию $S(L, L_{a_1}, \ldots, L_{a_n})$.

Прежде всего докажем, что для произвольных нетерминала $A \in N$ и цепочки $\gamma \in (V \cup N)^*$ из $A \vdash_G^* \gamma$ следует $A \vdash_{G'}^* \gamma'$, т.е. если в грамматике G из нетерминала A выводится цепочка γ в объединенном алфавите, то в грамматике G' из того же нетерминала A выводится „двойник“ цепочки γ, получаемый из последней заменой каждого ее терминального символа b аксиомой соответствующей грамматики $G_b = G_{a_i}$ для некоторого $i = \overline{1, n}$.

Доказательство проведем индукцией по длине l вывода цепочки γ из нетерминала A. Для вывода нулевой длины утверждение тривиально, так как $A \vdash^0 A = A'$. Пусть утверждение доказано при всех $l \leqslant m-1$, и пусть $A \vdash_G^m \gamma$. Тогда существует цепочка ξ, такая, что $A \vdash_G^{m-1} \xi \vdash_G \gamma$. Согласно предположению индукции, отсюда следует, что $A \vdash_{G'}^* \xi'$, а так как $\xi \vdash_G \gamma$, то существует такое правило вывода $B \to \beta$ в множестве P, что $\xi = \xi_1 B \xi_2$ (для некоторых $\xi_1, \xi_2 \in (V \cup N)^*$) и $\gamma = \xi_1 \beta \xi_2$. Следовательно, $\xi' = \xi_1' B \xi_2'$, а так как в множестве правил вывода P' есть правило $B \to \beta'$, то $\xi' \vdash_{G'} \xi_1' \beta' \xi_2' = \gamma'$. Итак, окончательно получаем $A \vdash_{G'}^* \xi' \vdash_{G'} \gamma'$.

Из доказанного следует, что для любой цепочки $x \in L = L(G)$, т.е. такой цепочки x, что $S \vdash_G^* x$, выполняется $S \vdash_{G'}^* x'$. Если $x = \lambda$, то тогда из $\lambda \in L$ следует, что $\lambda \in L(G')$. Если $x \neq \lambda$, тогда для некоторого $k > 0$ $x = x(1) \ldots x(k)$ и $x' = S_{x(1)} \ldots S_{x(k)}$, а так как для каждого $j = \overline{1, k}$ выполняется $S_{x(j)} \vdash_{G'}^* y_j$, какова бы ни была цепочка $y_j \in L_{x(j)}$, то $x' \vdash_{G'}^* y_1 \ldots y_k$. Но последняя цепочка есть произвольная цепочка суперпозиции $S(L, L_{a_1}, \ldots, L_{a_n})$ (не считая пустой цепочки, непосредственно выводимой из аксиомы S), так как она получена из произвольной непустой цепочки x языка L путем замены в ней каждого

символа $x(j)$ произвольной цепочкой языка $L_{x(j)}$. Поскольку из $S \vdash_G \lambda$ следует $S \vdash_{G'} \lambda$, любая цепочка суперпозиции $S(L, L_{a_1}, \ldots, L_{a_n})$ порождается грамматикой G'.

Докажем теперь, что если цепочка w в алфавите V' (терминальном алфавите грамматики G') выводится из аксиомы S в грамматике G', то она принадлежит суперпозиции $S(L, L_{a_1}, \ldots, L_{a_n})$. Для этого сначала докажем, что для любых нетерминала $A \in N$ и цепочки $\gamma \in (V \cup N)^*$ из $A \vdash_{G'}^* \gamma'$ следует $A \vdash_G^* \gamma$.

Опять воспользуемся индукцией по длине вывода, но на этот раз вывода цепочки γ' из нетерминала A в грамматике G'. Случай вывода нулевой длины, как и выше, тривиален. Пусть для некоторого $m > 0$ выполняется $A \vdash_{G'}^m \gamma'$. Тогда существует такая цепочка μ, что $A \vdash_{G'}^{m-1} \mu \vdash_{G'} \gamma'$. Каждый символ цепочки γ' есть либо одна из аксиом S_{a_i}, $i = \overline{1, n}$, либо нетерминал из N. Так как КС-грамматики G и G_{a_i}, $i = \overline{1, n}$, в силу результатов, полученных в **8.2**, можно считать заданными в *приведенной форме*, то правые части правил вывода каждой из рассматриваемых грамматик не содержат аксиомы. Кроме того, нетерминальные алфавиты N и N_{a_i} всех этих грамматик попарно не пересекаются.

Отсюда следует, что цепочка γ' не может быть непосредственно выведена из цепочки μ применением какого-либо правила из множества P_{a_i}, т.е. какого-либо правила вывода грамматики G_{a_i}, а выводится из μ применением некоторого правила вида $B \to \beta'$, построенного, как указано в определении грамматики G', по правилу $B \to \beta$ системы правил грамматики G. Тогда мы можем написать: $\gamma' = \gamma_1' \beta' \gamma_2'$ (для некоторых цепочек γ_1 и γ_2 в алфавите $V \cup N$), $\mu = \gamma_1' B \gamma_2' = (\gamma_1 B \gamma_2)'$. Это значит, что $\mu = \xi'$ при $\xi = \gamma_1 B \gamma_2$, т.е. $A \vdash_{G'}^{m-1} \xi'$. Согласно предположению индукции, отсюда следует, что $A \vdash_G^* \xi$, а так как $\xi = \gamma_1 B \gamma_2$ и в множестве правил вывода грамматики G содержится правило $B \to \beta$, то $\xi \vdash_G \gamma_1 \beta \gamma_2 = \gamma$ и окончательно $A \vdash_G^* \gamma$. В частности, отсюда следует, что для произвольной цепочки $x \in V^*$, такой, что $S \vdash_{G'}^* x'$, имеет место $S \vdash_G^* x$, т.е. $x \in L$.

Теперь докажем, что произвольная цепочка α, выводимая в грамматике G' из аксиомы S, имеет вид $\alpha_1\alpha_2\ldots\alpha_m$ $(m \geqslant 1)$, где для каждого $j = \overline{1,m}$ цепочка α_j либо пуста, либо для некоторого символа $b_j \in V$ выводится из аксиомы S_{b_j} грамматики G_{b_j}, т.е. $S_{b_j} \vdash^*_{G_{b_j}} \alpha_j$, либо является некоторой цепочкой нетерминалов из N. Действительно, сама аксиома S удовлетворяет этому требованию. Далее, пусть доказываемое справедливо для любой цепочки, выводимой в G' за любое число шагов, не превышающее $l-1$ для некоторого $l > 1$.

Предположим, что $S \vdash^l_{G'} \alpha$. Тогда, как это мы уже делали в аналогичных ситуациях, рассмотрим последний шаг этого вывода, т.е. для некоторой цепочки γ будем иметь

$$S \vdash^{l-1}_{G'} \gamma \vdash_{G'} \alpha.$$

Согласно предположению индукции, цепочка γ представима в виде

$$\gamma = \gamma_1\gamma_2\cdots\gamma_m,$$

где $\gamma_j = \lambda$, $j = \overline{1,m}$, или $S_{b_j} \vdash^*_{G_{b_j}} \gamma_j$ для некоторого $b_j \in V$, или $\gamma \in N^*$. Так как к цепочке γ применяется некоторое правило грамматики G' (на последнем шаге вывода цепочки α из аксиомы S), то в γ входит по крайней мере один нетерминал грамматики G'. Если это есть некоторый символ $A \in N$, то он может входить только в одну из таких подцепочек γ_j, которая состоит из нетерминалов алфавита N. После замены A посредством некоторого правила вида $A \to \beta'$, соответствующего правилу $A \to \beta$ грамматики G, получится цепочка, в которую могут входить только нетерминалы множества N и аксиомы S_b грамматик G_b для каких-либо $b \in V$.

Следовательно, если цепочка α на последнем шаге ее вывода из аксиомы получена применением некоторого правила $A \to \beta'$, то она также может быть представлена соединением цепочек такого же вида, что и цепочки γ_j.

Если же $S_{b_j} \vdash^*_{G_{b_j}} \gamma_j$ для некоторого $b_j \in V$ и на указанном шаге применяется правило из множества P_{b_j}, то оно, ввиду того что все множества N и N_b (для различных $b \in V$) попарно не пересекаются, может быть применено только к такой подцепочке γ_j, которая выводится из аксиомы S_{b_j}, и тогда в цепочке α получим подцепочку, опять-таки выводимую из S_{b_j}, т.е. и в этом случае* цепочка α образуется соединением подцепочек того же вида, что и цепочки γ_j.

Пусть теперь цепочка $w \in V'^*$ такова, что $S \vdash^*_{G'} w$. Если $w = \lambda$ и $S \vdash_{G'} \lambda$, то это означает наличие правила вывода $S \to \lambda$ в множестве P' и, следовательно, в множестве P, т.е. $\lambda \in L$ и $\lambda \in \mathcal{S}(L, L_{a_1}, \ldots, L_{a_n})$. Таким образом в этом случае пустая цепочка, порождаемая грамматикой G', действительно принадлежит указанной суперпозиции. Иначе в силу доказанного выше свойства любой цепочки, выводимой в грамматике G' из аксиомы, цепочка w, не содержащая по условию вхождений нетерминалов из N, допускает представление в виде $w_1 w_2 \ldots w_k$, где для каждого $j = \overline{1, k}$ имеет место выводимость $S_{b_j} \vdash^*_{G_{b_j}} w_j$ (возможно для некоторых j, что $w_j = \lambda$; в частности, может оказаться, что и $w = \lambda$). Таким образом,

$$S \vdash^*_{G'} S_{b_1} S_{b_2} \ldots S_{b_k} \vdash^*_{G'} w_1 w_2 \ldots w_k,$$

где для каждого $j = \overline{1, k}$ цепочка w_j принадлежит языку L_{b_j}. Так как при этом $S \vdash^*_G b_1 b_2 \ldots b_k$, т.е. $b_1 b_2 \ldots b_k \in L$, то $w \in \in \mathcal{S}(L, L_{a_1}, \ldots, L_{a_n})$.

Итак, доказав, что любая цепочка w в терминальном алфавите V' грамматики G', выводимая из аксиомы, есть элемент

*Заметим, что в приведенном выше доказательстве существенно используется тот факт, что нетерминальные алфавиты N, N_{a_1}, ..., N_{a_n} попарно не пересекаются. Если бы это было не так, то в некоторой подцепочке γ_j, выводимой из аксиомы S_{b_j} для некоторого $b_j \in V$, мы могли бы заменить вхождение нетерминала из N_{b_j}, применив правило вывода „чужой“ грамматики G_{a_k}, $a_k \neq b_j$, и получить в составе цепочки α подцепочку, не выводимую ни из одной „дочерней“ аксиомы S_b, $b \in V$.

суперпозиции $S(L, L_{a_1}, \ldots, L_{a_n})$, мы тем самым завершили доказательство теоремы. ▶

Следствие 8.3. Семейство КС-языков замкнуто относительно операций объединения, соединения, итерации и позитивной итерации.

◀ Доказательство получается немедленно из приведенных выше определений операций объединения, соединения, итерации и позитивной итерации как частных случаев суперпозиции. ▶

Заметим, что объединение бесконечного семейства КС-языков не является, вообще говоря, КС-языком. Так, язык

$$\{a^n b^n c^n : n \geqslant 0\},$$

не являющийся КС-языком (см. **8.3**), может быть представлен в виде объединения счетного семейства, каждый элемент которого есть язык $\{a^n b^n c^n\}$ для фиксированного натурального n, т.е. язык, состоящий из одной цепочки, а следовательно, регулярный и контекстно-свободный.

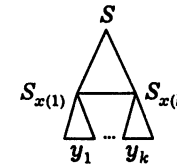


Рис. 8.34

Замечание 8.12. Вывод цепочки w из суперпозиции $S(L, L_{a_1}, \ldots, L_{a_n})$ (если она не является пустой цепочкой, непосредственно выводимой из аксиомы S) в грамматике G', построенной в доказательстве теоремы 8.9, можно провести по „двухуровневой“ схеме, приведенной на рис. 8.34. Сначала порождаем, используя „перекрашенные“ правила грамматики G, т.е. правила вида $A \to \gamma'$ для $A \in N$ и $\gamma \in (V \cup N)^*$, „двойника“ произвольной цепочки $x \in L$, т.е. цепочку $S_{x(1)} \ldots S_{x(k)}$. Из нее затем, используя правила „дочерних“ грамматик, порождаем саму цепочку w, из каждой аксиомы $S_{x(j)}$ выводя некоторую цепочку y_j так, что $w = y_1 \ldots y_k$.

Пример 8.20. Рассмотрим конструкцию из доказательства теоремы 8.9 на конкретном примере.

Язык L определим как язык, порождаемый грамматикой G со следующим множеством правил вывода:

$$S \to ab \,|\, aSb \,|\, SS.$$

Языки L_a и L_b определим так:

$$L_a = \{0^n 1^n \colon n \geqslant 0\},$$
$$L_b = \{x \colon x \in \{a, b\} \text{ и } x = x^R\}$$

(т.е. L_b есть язык палиндромов в алфавите $\{a, b\}$). Запишем множества правил вывода грамматик G_a и G_b, порождающих языки L_a и L_b соответственно:

$$S \to 0S1 \,|\, 01 \,|\, \lambda$$

(грамматика G_a) и

$$S \to aSa \,|\, bSb \,|\, aa \,|\, bb \,|\, a \,|\, b \,|\, \lambda$$

(грамматика G_b).

Преобразуя грамматики G, G_a и G_b к приведенной форме (убирая аксиому из правых частей правил вывода) и переименовывая нетерминалы так, чтобы множества N, N_a и N_b попарно не пересекались, построим, как описано в доказательстве теоремы 8.9, множество P' правил вывода грамматики

$$G' = \big(\{0, 1, a, b\},\ \{S, T, S_a, T_a, S_b, T_b\},\ S,\ P'\big),$$

порождающей суперпозицию $\mathcal{S}(L, L_a, L_b)$:

$$\begin{cases} S \to S_a S_b \,|\, S_a T S_b \,|\, TT, \\ T \to S_a S_b \,|\, S_a T S_b \,|\, TT, \\ S_a \to 0T_a 1 \,|\, 01 \,|\, \lambda, \\ T_a \to 0T_a 1 \,|\, 01, \\ S_b \to aT_b a \,|\, bT_b b \,|\, aa \,|\, bb \,|\, a \,|\, b \,|\, 0 \,|\, \lambda, \\ T_b \to aT_b a \,|\, bT_b b \,|\, aa \,|\, bb \,|\, a \,|\, b \,|\, 0. \end{cases} \tag{8.19}$$

Рассмотрим дерево вывода цепочки $010011abab$, изображенное на рис. 8.35. Если при порождении этой цепочки использовать „двухуровневую“ схему, описанную в замечании 8.12, то получим такой вывод:

$$S \vdash S_aTS_b \vdash S_aS_aS_bS_b \vdash 01S_aS_bS_b \vdash$$
$$\vdash 010T_a1S_bS_b \vdash 010011S_bS_b \vdash 010011aT_baS_b \vdash$$
$$\vdash 010011abaS_b \vdash 010011abab.$$

Сначала порождается „двойник“ цепочки $abab \in L$, а затем на место первого символа (замененного аксиомой S_a „дочерней“ грамматики G_a) подставляется цепочка 01, на место второго символа — цепочка 0011, на место третьего — aba, на место четвертого — цепочка b.

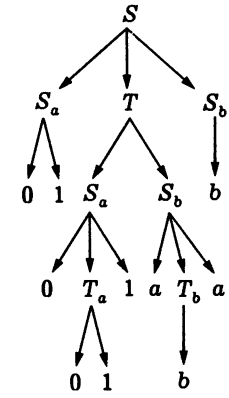


Рис. 8.35

Заметим, что в общем случае построения вывода в грамматике G' (если не обязательно придерживаться „двухуровневой“ стратегии) „дочерняя“ грамматика может начать „работать“, как только в выводе появляется ее аксиома. Так, по изображенному на рис. 8.35 дереву вывода может быть построен следующий левый вывод:

$$S \vdash S_aTS_b \vdash 01S_aS_bS_b \vdash 010T_a1S_bS_b \vdash$$
$$\vdash 010011S_bS_b \vdash 010011aT_baS_b \vdash$$
$$\vdash 010011abaS_b \vdash 010011abab.$$

Но если мы пренебрежем требованием, чтобы нетерминальные алфавиты грамматик G, G_a и G_b попарно не пересекались, и отождествим, скажем, нетерминалы T_a, T_b и T, положив $T_a = T_b = T$, то сможем вывести цепочку, не принадлежащую суперпозиции $S(L, L_a, L_b)$. Например,

$$S \vdash S_aTS_b \vdash 01TS_b \vdash 01bbS_b \vdash 01bba.$$

В самом деле, цепочка 01 может быть получена применением либо правила $S_a \to 01$, либо правила $T_a \to 01$; цепочка bb — применением правила $T_b \to bb$ или $S_b \to bb$, цепочка a — применением правила $S_b \to a$ или $T_b \to a$. Можно легко убедиться в том, что в этом случае единственная цепочка, состоящая из аксиом „дочерних" грамматик, т.е. символов S_a, S_b, из которой может быть выведена цепочка $01bba$, равна $S_aS_bS_b$. Но эта цепочка не выводима из аксиомы S. Если же мы, анализируя цепочку $01bba$, стали бы рассматривать в качестве правой части правила вывода не цепочку bb, а цепочку b, то получили бы подцепочку ba, не являющуюся правой частью ни одного правила вывода, и тогда имели бы S_aS_bba — цепочку, не выводимую из S.

Мы вывели „плохую" цепочку здесь потому, что „перепутав" нетерминалы грамматик, позволили „вклиниться" „дочерней" грамматике G_b в работу грамматики „верхнего уровня". Аналогично можно построить пример, когда разные „дочерние" грамматики „мешают" друг другу.

Итак, требование, согласно которому нетерминальные алфавиты грамматик G и всех G_{a_i} попарно не пересекаются, является существенным. #

Исследуем теперь вопрос о пересечении КС-языков. Введем обозначение: если дан алфавит $\{b_1, \ldots, b_k\}$, то через $[b_1^n \ldots b_k^n]$ обозначим язык $\{b_1^n \ldots b_k^n \colon n \geqslant 0\}$.

Теорема 8.10. Класс КС-языков не замкнут относительно пересечения.

◀ Языки $a^*[b^nc^n]$ и $[a^nb^n]c^*$, как можно легко доказать, являются КС-языками, но их пересечение есть язык $[a^nb^nc^n]$, который — как это было доказано в примере 8.11 — не КС-язык. ▶

Следствие 8.4. Дополнение КС-языка в общем случае не является КС-языком.

◀ Действительно, если бы для любого КС-языка L его дополнение $\overline{L}$ было бы КС-языком, то для пересечения любых двух

КС-языков L_1 и L_2 мы имели бы: $L_1 \cap L_2 = \overline{\overline{L_1} \cup \overline{L_2}}$ — КС-язык, что противоречит теореме 8.10. ▶

Однако оказывается, что класс КС-языков замкнут относительно операции ***суженного пересечения языков*** — операции пересечения с регулярными языками.

Теорема 8.11. Если L — КС-язык, а R — регулярный язык, то $L \cap R$ — КС-язык.

◀ Пусть $G = (V, N, S, P)$ — КС-грамматика, порождающая язык L, а $M = (Q, V, q_0, F, \delta)$ — конечный автомат, допускающий язык R (см. **7.5**).

Будем считать, что грамматика G дана в приведенной форме, причем аксиома не содержится в правых частях правил вывода (см. **8.2**). Конечный же автомат M является детерминированным.

В основе конструкции грамматики для пересечения $L \cap R$ лежит следующая неформальная идея. Мы хотим построить такую КС-грамматику G', которая порождала бы все цепочки, одновременно порождаемые грамматикой G и допускаемые конечным автоматом M. Это значит, что любая цепочка x, порождаемая грамматикой G', должна удовлетворять двум требованиям: 1) $S \vdash_G^* x$ и 2) в M должен найтись единственный путь из начального состояния в одно из заключительных состояний, на котором читается цепочка x. Если цепочка x пустая, то она принадлежит пересечению $L \cap R$ тогда и только тогда, когда в P есть правило $S \to \lambda$, а начальное состояние автомата M является и заключительным. Для непустой цепочки $x = x(1)x(2)\ldots x(k)$ из пересечения $L \cap R$ тогда должна существовать единственная последовательность состояний $q_0, s_1, \ldots, s_{k-1}, q_f$ $(q_f \in F)$ множества Q, в которой $q_0 \to_{x(1)} s_1$,

$$s_1 \to_{x(2)} s_2, \ldots, s_{k-1} \to_{x(k)} q_f.$$

Будем тогда в качестве нетерминалов грамматики G' рассматривать всевозможные упорядоченные тройки вида $[qXr]$,

где q, r — состояния конечного автомата M, а X — символ из объединенного алфавита грамматики G (похожий прием мы применяли, строя КС-грамматику, эквивалентную заданному МП-автомату, см. **8.4**). Заставим грамматику G' выводить из аксиомы все непустые цепочки вида

$$[q_0x(1)s_1][s_1x(2)s_2]\dots[s_{k-1}x(k)q_f]$$

для всех $q_f \in F$ и всех последовательностей $s_1, s_2, \dots, s_{k-1}$ состояний из Q при условии, что $S \vdash_G^* x(1)x(2)\dots x(k)$. Нетерминал же $[qar]$ при $a \in V$ может быть, по определению, заменен терминалом a тогда и только тогда, когда в системе команд δ конечного автомата M есть команда $qa \to r$, т.е. из состояния q по символу a можно перейти в состояние r.

При таком определении грамматики G' она породит непустую терминальную цепочку x тогда и только тогда, когда x порождается грамматикой G и читается на некотором пути из начального состояния в одно из заключительных, — именно тогда (и только тогда) из цепочки $[q_0x(1)s_1][s_1x(2)s_2]\dots[s_{k-1}x(k)q_f]$ может быть выведена сама цепочка $x = x(1)x(2)\dots x(k)$. Образно говоря, грамматика G' каждый символ непустой цепочки x, порождаемой грамматикой G, помещает между двумя „стражами“ — состояниями конечного автомата, и символ может избавиться от этих „стражей“ тогда и только тогда, когда в конечном автомате M есть переход по нему из первого состояния во второе.

Дадим теперь формальное определение грамматики G':

$$G' = (V,\ N',\ S',\ P'),$$

где $N' = \{[qXr]\colon q, r \in Q \text{ и } X \in V \cup N\} \cup \{S'\}$, $S' \notin V \cup N$ (аксиома грамматики G'), а множество правил вывода P' строится следующим образом:

1) правило вывода $S' \to \lambda$ принадлежит P' тогда и только тогда, когда в множестве правил P грамматики G есть правило $S \to \lambda$, а для конечного автомата M имеет место $q_0 \in F$;

2) для любого правила $A \to \gamma$ в P $(\gamma \neq \lambda)$ в P' вводится множество всех правил вида

$$[s_1 A s_{k+1}] \to [s_1 \gamma(1) s_2][s_2 \gamma(2) s_3] \ldots [s_k \gamma(k) s_{k+1}]$$

для произвольной последовательности $s_1, s_2, \ldots, s_{k+1}$ состояний из множества Q $(k \geqslant 1$ — длина цепочки $\gamma)$;

3) для любого заключительного состояния $q_f \in F$ конечного автомата M в P' вводится правило вывода $S' \to [q_0 S q_f]$, где S — аксиома грамматики G;

4) правило вывода $[qar] \to a$ для $a \in V$ принадлежит P' тогда и только тогда, когда команда $qa \to r$ принадлежит системе команд δ конечного автомата M;

5) никаких других правил вывода, кроме указанных в пп. 1–4, множество P' не содержит.

Непосредственно из построения грамматики G' видно, что пустая цепочка λ порождается грамматикой G' тогда и только тогда, когда правило вывода $S \to \lambda$ содержится в множестве P, т.е. $\lambda \in L$, и конечный автомат M допускает пустую цепочку, т.е. $q_0 \in F$. Итак, $S' \vdash^* \lambda$ тогда и только тогда, когда $\lambda \in L \cap R$.

Для непустой цепочки $x = x(1)x(2)\ldots x(m)$ можно доказать (подробное доказательство мы опускаем*), что $S \vdash_G^* x$, т.е. $x \in L$, тогда и только тогда, когда

$$S' \vdash [q_0 S q_f] \vdash^* [q_0 x(1) s_1][s_1 x(2) s_2] \ldots [s_{m-1} x(m) q_f]$$

для любых $s_1, \ldots, s_{m-1} \in Q$.

Тогда из цепочки $[q_0 x(1) s_1][s_1 x(2) s_2] \ldots [s_{m-1} x(m) q_f]$ цепочка $x = x(1)x(2)\ldots x(m)$ может быть выведена в грамматике G' применением правила, приведенного выше в п. 4, в том и только в том случае, когда для конечного автомата M имеет место

$$q_0 \to_{x(1)} s_1 \to_{x(2)} s_2 \to \ldots \to s_{m-1} \to_{x(m)} q_f$$

*Это доказательство без особого труда может быть восстановлено с помощью метода индукции по длине вывода.

для некоторых состояний $s_1, \ldots, s_{m-1}$, т.е. x читается на пути из q_0 в q_f, проходящем через состояния $s_1, \ldots, s_{m-1}$, и тем самым $x \in R$, т.е. $x \in L \cap R$.

Итак, окончательно $x \in L(G') \Leftrightarrow x \in L \cap R$. ▶

Пример 8.21. Построим грамматику, порождающую пересечение языка всех *палиндромов* в алфавите $\{a, b\}$ с языком a^*bba^*.

Грамматику для языка палиндромов задаем в виде

$$S \to aTa \,|\, bTb \,|\, aa \,|\, bb \,|\, a \,|\, b \,|\, \lambda,$$
$$T \to aTa \,|\, bTb \,|\, aa \,|\, bb \,|\, a \,|\, b,$$

где S — аксиома. Граф конечного автомата, допускающего язык a^*bba^*, приведен на рис. 8.36.

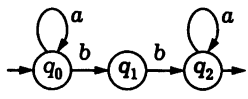


Рис. 8.36

Согласно правилам построения грамматики G', изложенным в доказательстве теоремы 8.11, имеем следующую грамматику для пересечения заданных языков:

$$S' \to [q_0 S q_2],$$

$$[r_1 S r_4] \to [r_1 a r_2][r_2 T r_3][r_3 a r_4] \,|\, [r_1 b r_2][r_2 T r_3][r_3 b r_4] \qquad (8.20)$$

(для любых последовательностей состояний r_1, r_2, r_3, r_4 конечного автомата);

$$[r_1 S r_3] \to [r_1 a r_2][r_1 a r_2] \,|\, [r_1 b r_2][r_2 b r_3]$$

(для любых последовательностей состояний r_1, r_2, r_3 конечного автомата);

$$[r_1 S r_2] \to [r_1 a r_2] \,|\, [r_1 b r_2]$$

(для любых последовательностей состояний r_1, r_2 конечного автомата);

$$[r_1 T r_4] \to [r_1 a r_2][r_2 T r_3][r_3 a r_4] \,|\, [r_1 b r_2][r_2 T r_3][r_3 b r_4]$$

(для любых последовательностей состояний r_1, r_2, r_3, r_4 конечного автомата);

$$[r_1 T r_3] \to [r_1 a r_2][r_1 a r_2] \,|\, [r_1 b r_2][r_2 b r_3] \tag{8.21}$$

(для любых последовательностей состояний r_1, r_2, r_3 конечного автомата);

$$[r_1 T r_2] \to [r_1 a r_2] \,|\, [r_1 b r_2]$$

(для любых последовательностей состояний r_1, r_2 конечного автомата);

$$[q_0 a q_0] \to a, \tag{8.22}$$

$$[q_0 b q_1] \to b, \tag{8.23}$$

$$[q_1 b q_2] \to b, \tag{8.24}$$

$$[q_2 a q_2] \to a. \tag{8.25}$$

Рассмотрим пример порождения какой-нибудь цепочки из определенного выше пересечения двух языков. Возьмем цепочку $abba$. Выведем сначала „двойника“ этой цепочки, в котором каждый символ „окружен“ состояниями конечного автомата. В процессе вывода мы стараемся „положить“ нашу цепочку на некоторый путь из начальной вершины автомата в заключительную:

$$S' \vdash [q_0 S q_2] \vdash [q_0 a q_0][q_0 T q_2][q_2 a q_2] \vdash [q_0 a q_0][q_0 b q_1][q_1 b q_2][q_2 a q_2].$$

На втором шаге написанного выше вывода мы применили первое из правил (8.20) при $r_1 = r_2 = q_0$, $r_3 = r_4 = q_2$, а на третьем шаге — второе правило (8.21) при $r_1 = q_0$, $r_2 = q_1$, $r_3 = q_2$.

Теперь мы видим, что все „скобки“ состояний можно „отряхнуть“: последовательно применяя правила (8.22)–(8.25), получаем цепочку $abba$.

Рассмотрим теперь „неправильную“ цепочку $aba \notin a^* b b a^*$. Вывод ее „двойника“ может быть таким:

$$S' \vdash [q_0 S q_2] \vdash [q_0 a q_0][q_0 T q_2][q_2 a q_2] \vdash [q_0 a q_0][q_0 b q_2][q_2 a q_2].$$

При выводе мы старались помещать каждый входной символ конечного автомата между такими состояниями, чтобы он входил в метку дуги из первого во второе состояние, но мы видим, что нетерминал $[q_0bq_2]$ не может быть заменен ни одним терминалом, и вывод зашел в тупик. Разбор других вариантов вывода „двойника“ цепочки aba мы опускаем. В данном случае оказывается, что любой вывод закончится тупиковой цепочкой. #

Заметим, что в рассмотренном примере конструкцию теоремы нельзя применять к грамматике языка палиндромов, если она задана в виде

$$S \to aSa \,|\, bSb \,|\, a \,|\, b \,|\, \lambda,$$

поскольку в этом случае нельзя обойтись без применения правила $S \to \lambda$ при порождении цепочки четной длины.

Так как при построении грамматики G' правила с пустой правой частью (из множества правил грамматики G) никак не преобразуются, грамматика G' в таком случае не породит „двойника“ ни одной цепочки четной длины языка палиндромов.

Следовательно, требование, чтобы грамматика КС-языка L была задана в приведенной форме и ее единственное разрешенное λ-правило $S \to \lambda$ применялось только при выводе пустой цепочки, является существенным при построении КС-грамматики для пересечения языка L с регулярным языком R.

Доказанное утверждение о „контекстной свободности“ пересечения КС-языка с любым регулярным языком в совокупности с леммой о разрастании для КС-языков полезно при доказательстве утверждений о том, что какой-либо язык не является контекстно-свободным.

Пример 8.22. Докажем, что язык $L = \{ww\colon w \in \{a, b\}^*\}$ — так называемый язык двойных слов в алфавите $\{a, b\}$ — не является контекстно-свободным.

Применить к решению этой задачи сразу лемму о разрастании довольно трудно. Поступим так. Рассмотрим пересечение языка L с регулярным языком $a^*b^*a^*b^*$. Легко понять, что это пересечение состоит из всех цепочек вида $a^m b^n a^m b^n$ $(m, n \geqslant 0)$. Предполагая, что язык L контекстно-свободный, получим в силу теоремы 8.11, что контекстно-свободным является и язык $\{a^m b^n a^m b^n : m, n \geqslant 0\}$. Однако этот язык не есть КС-язык (см. пример 8.12). Следовательно, не является КС-языком и исходный язык L.

Дополнение 8.1. О методах синтаксического анализа КС-языков

Проблема синтаксического анализа для *КС-языков* состоит в построении алгоритма, который по любой *КС-грамматике* $G = (V, N, S, P)$ и *цепочке* x в ее *терминальном алфавите* V распознает, принадлежит ли x *языку* $L(G)$, *порождаемому грамматикой* G. В случае положительного ответа на вопрос алгоритм должен строить *дерево вывода* x в грамматике G.

Существование такого алгоритма следует из факта разрешимости *проблемы принадлежности* для КС-языков. Мы рассмотрим некоторые алгоритмы синтаксического анализа для определенных классов КС-языков.

Прототипом синтаксического анализатора является *МП-автомат*, который строится по данной КС-грамматике (см. **8.4**), но такой МП-автомат, как мы видели, является в общем случае недетерминированным и может даже для *допустимой цепочки* построить *вывод*, который заканчивается *тупиковой конфигурацией*. Чтобы на основе такого *распознавателя* построить алгоритм синтаксического анализа (и тем самым превратить распознаватель в ***анализатор***), нужно разработать определенный ***механизм управления выводом*** на множестве *конфигураций МП-автомата*. Этот механизм должен обеспечить эффективный поиск *допускающей последовательности конфи-*

граций для допустимых цепочек. В частности, может быть поставлена задача разработки алгоритма ***беступикового***, или ***однопроходного***, ***анализа***. Беступиковый анализ имеет место, если получение тупиковой конфигурации в процессе анализа означает „неправильность" анализируемой цепочки, т.е. тот факт, что она не принадлежит языку, порождаемому заданной грамматикой. Беступиковый анализ, как можно показать, невозможен в случае произвольного КС-языка, но для некоторых (достаточно широких) классов КС-языков он может быть реализован. Некоторые алгоритмы беступикового синтаксического анализа мы очень коротко рассмотрим в этом дополнении.

Существуют две основные стратегии синтаксического анализа: 1) ***нисходящий анализ*** (называемый также анализом „***сверху вниз***", или анализом „***разверткой***") и 2) ***восходящий анализ*** (анализ „***снизу вверх***", „***сверткой***").

В нисходящем анализе дерево вывода цепочки строится от *корня* к *листьям*, т.е. дерево вывода „реконструируется" в прямом порядке, и *аксиома грамматики* „развертывается" в цепочку. В восходящем анализе дерево вывода строится от листьев к корню и анализируемая цепочка „свертывается" в аксиому.

Рассмотрим две стратегии анализа по очереди.

Нисходящий анализ. $LL(k)$-грамматики. Мы видели, что МП-автомат, который при доказательстве теоремы 8.8 мы строили по данной КС-грамматике, „воспроизводит" левый вывод в грамматике. Основная „задача" данного автомата как распознавателя состоит в том, чтобы на каждом *шаге* „угадать" очередное применяемое *правило вывода* и „правильно выбрать" соответствующую *команду* при построении допускающей последовательности конфигураций. Механизм управления выводом, которым мы должны снабдить классический МП-автомат, должен обеспечить выбор команды (единственной в случае беступикового анализа) по определенной информации о состоянии процесса поиска дерева вывода (или, что равно-

сильно, *левого вывода*) *входной цепочки*. Обычно механизм управления выводом реализуется в виде специальных таблиц, которые называют ***управляющими таблицами*** и которые можно рассматривать как дополнительное устройство памяти в МП-автомате.

Существует класс грамматик, называемых ***$LL(k)$-грамматиками***, в которых применяемая команда однозначно определяется:

1) *прочитанной частью* w входной цепочки; эту цепочку w называют ***левым контекстом***;

2) находящимся в данный момент в *верхней ячейке магазина нетерминальным символом* A;

3) *началом* (*префиксом*) v *непрочитанной части* цепочки, состоящим не более чем из k букв ($k \geqslant 1$); этот префикс называют ***аванцепочкой*** (рис. 8.37).

Эта информация, по которой беступиковый анализатор выбирает нужную команду на каждом шаге работы с входной цепочкой, организована в виде управляющей таблицы (конкретные формы таких таблиц будут рассмотрены ниже). При этом левый вывод цепочки $x = wvu$, если $x \in L(G)$, может быть представлен следующим образом: из аксиомы выводится цепочка $wA\alpha$, затем нетерминал A заменяется в соответствии с правилом $A \to \gamma$ и из цепочки $\gamma\alpha$ выводится терминальная цепочка vu, где

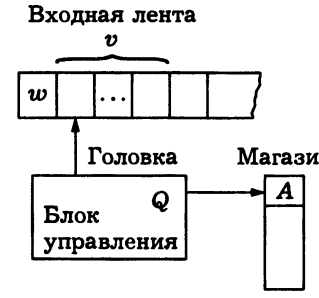


Рис. 8.37

$$|v| \leqslant k \quad \text{и} \quad S \overset{l}{\vdash}{}^{*} wA\alpha \overset{l}{\vdash} w\gamma\alpha \overset{l}{\vdash}{}^{*} wvu$$

(символ $\overset{l}{\vdash}$ означает левую выводимость).

Описанное выше представление левого вывода цепочки wvu предполагает, что мы произвольно в этом выводе фиксировали некий шаг, состоящий в замене нетерминала A посредством

применения правила вывода $A \to \gamma$. Так как в левом выводе на каждом шаге производится замена *самого левого вхождения* нетерминального символа, то слева от A в цепочке $wA\alpha$, полученной перед рассматриваемым шагом, должны быть только терминальные символы. Следовательно, определенный выше левый контекст w есть не что иное, как *левое крыло вхождения* нетерминала A в цепочку $wA\alpha$.

Моделируя этот левый вывод, т.е. читая записанную на входной ленте цепочку $x = wvu$, МП-автомат читает левый контекст w, а затем с помощью управляющей таблицы, „видя" в магазине символ A, учитывая левый контекст w и зная аванцепочку v, принимает единственно правильное решение, применяя команду, соответствующую правилу $A \to \gamma$. Так на интуитивном уровне можно определить ключевое свойство $LL(k)$-грамматики.

Переходим теперь к построению формального определения $LL(k)$-грамматики.

Пусть дана КС-грамматика G. Для цепочки α в *объединенном алфавите* грамматики G и положительного натурального k определим множество $F_k(\alpha)$, состоящее из всех терминальных цепочек, которые либо выводятся из цепочки α (если их длина строго меньше k), либо являются k-буквенными префиксами терминальных цепочек, выводимых из α (обратим внимание на то, что везде речь идет о левом выводе).

Таким образом,

$$F_k(\alpha) = \{v : (\alpha \overset{l}{\vdash}{}^* v)\&(|v| < k) \vee (\exists x \in V^*)(\alpha \overset{l}{\vdash}{}^* vx)\&(|v| = k)\}.$$

Нетрудно видеть, что для всякой терминальной цепочки x получим $F_k(x) = \{x\}$, если $|x| \leqslant k$, и $F_k(x) = \{x(1)x(2)\ldots x(k)\}$, если $|x| > k$. Множества $F_k(\alpha)$ (для разных цепочек α) иногда будем называть ***F_k-множествами***.

Пример 8.23. Зададим грамматику G с терминальным алфавитом $\{a, b, c, d\}$ и нетерминальным алфавитом, состоящим

из одной аксиомы S, следующим множеством правил вывода:

$$S \to aSbSc \mid aSb \mid bSc \mid d.$$

Вычислим множество $F_3(aSbSc)$. Первая буква всех терминальных цепочек, выводимых из заданной, уже фиксирована — это буква a. Нетерминал S может быть заменен буквой d, после чего возникнет трехбуквенный префикс adb. Но символ S можно заменить и цепочками $aSbSc$, aSb, bSc. В силу этого первые две буквы цепочки, выводимой из исходной, могут быть либо aa, либо ab. Продолжение вывода, как нетрудно понять, может дать третью букву — либо a, либо b, либо d. Окончательно получаем $F_3(aSbSc) = \{adb, aaa, aab, aad, aba, abb, abd\}$.

Определение 8.11. КС-грамматику $G = (V, N, S, P)$ называют ***LL(k)-грамматикой*** (для произвольно фиксированного $k \geqslant 1$), если для любых $w \in V^*$, $A \in N$, $\alpha \in (V \cup N)^*$, таких, что существуют левые выводы

$$S \overset{l}{\vdash}{}^* wA\alpha \overset{l}{\vdash} w\beta\alpha \overset{l}{\vdash}{}^* wy$$

и

$$S \overset{l}{\vdash}{}^* wA\alpha \overset{l}{\vdash} w\gamma\alpha \overset{l}{\vdash}{}^* wz,$$

из $F_k(y) = F_k(z)$ следует $\beta = \gamma$.

Таким образом, из этого определения следует, что для $LL(k)$-грамматики левый контекст w нетерминала A в цепочке $wA\alpha$ и не более чем k символов, с которых начинается терминальная цепочка y, выводимая из $A\alpha$, однозначно определяют то правило, которое нужно применить к цепочке $wA\alpha$ (заменяя в ней выделенное, т.е. первое, вхождение нетерминала A), чтобы сделать очередной шаг в левом выводе цепочки wy (и именно этой цепочки!) из аксиомы грамматики G. При совпадении множеств $F_k(y)$ и $F_k(z)$, как следует из сформулированного определения, указанные два вывода „сливаются“ в один, т.е. тогда $y = z$.

Определение $LL(k)$-грамматики является само по себе труднопроверяемым. Нужно какое-то условие, проверка которого позволила бы дать ответ на вопрос, является ли заданная КС-грамматика $LL(k)$-грамматикой. Доказывается, что дать ответ на этот вопрос можно, только фиксировав число k. Построить же алгоритм, отвечающий на вопрос, является ли данная КС-грамматика $LL(k)$-грамматикой для некоторого k (не задавая его заранее), невозможно.

Следующая теорема (формулируемая без доказательства) дает соответствующий критерий (необходимое и достаточное условие) при фиксированном k.

Теорема 8.12. КС-грамматика G является $LL(k)$-грамматикой (для фиксированного $k \geqslant 1$) тогда и только тогда, когда для любой цепочки $w \in V^*$ выполняется следующее условие: для любых $A \in N$, $\alpha \in (V \cup N)^*$, таких, что $S \overset{l}{\vdash}{}^* wA\alpha$, и любых двух различных правил $A \to \beta$ и $A \to \gamma$ грамматики G имеет место $F_k(\beta\alpha) \cap F_k(\gamma\alpha) = \varnothing$. #

Условие теоремы называют часто ***$LL(k)$-условием***. Его следует проверять, фиксируя по очереди левые контексты (т.е. различные цепочки w) для всех нетерминалов грамматики.

Рассмотрим пример, в котором, используя теорему 8.12, мы убедимся в том, что заданная КС-грамматика является $LL(k)$-грамматикой (для фиксированного k).

Пример 8.24. Грамматика G задается системой правил

$$S \to abA \,|\, \lambda, \qquad (1)\,|\,(2)$$
$$A \to Saa \,|\, b. \qquad (3)\,|\,(4).$$

Докажем, что данная грамматика является $LL(2)$-грамматикой. Чтобы проверить условие теоремы 8.12, достаточно для каждого нетерминала $B \in \{S, A\}$ нашей грамматики проделать следующее: 1) вычислить все такие цепочки $w \in \{a, b\}^*$ и $\alpha \in \in \{a, b, S, A\}$, что имеет место левая выводимость

$$S \overset{l}{\vdash}{}^* wB\alpha; \tag{8.26}$$

2) фиксировав „левый контекст“ w, для каждой пары различных правил вывода $B \to \gamma$ и $B \to \beta$ вычислить множества $F_2(\beta\alpha)$ и $F_2(\gamma\alpha)$ для всех таких α, что при фиксированном w выполняется (8.26), и убедиться в том, что их пересечение пусто.

Если выполнение п. 2 для всех возможных левых контекстов w и всех нетерминалов B подтвердит истинность условия теоремы 8.12, то тем самым и будет доказано, что перед нами $LL(2)$-грамматика.

Следует заметить, что в общем случае вычисление пар цепочек (w, α), удовлетворяющих условию (8.26) (для всех нетерминалов B КС-грамматики), требует применения специального алгоритма. Но для конкретной грамматики нашего примера эти пары цепочек достаточно просто вычисляются из анализа выводов грамматики. По очереди проанализуем оба нетерминала, т.е. S и A, грамматики G.

Нетерминал S. В этом случае имеем, во-первых, тривиальную выводимость $S \vdash^0 S$ за нуль шагов, т.е. в этом случае цепочки w и α обе являются пустыми: $w = \alpha = \lambda$. Нетерминал S может быть заменен согласно правилу (1) с правой частью abA или согласно правилу (2) с пустой правой частью. Вычислим тогда множества $F_2(abA)$ и $F_2(\lambda)$. Ясно, что $F_2(abA) = \{ab\}$, а $F_2(\lambda) = \{\lambda\}$ и эти множества не пересекаются.

Проанализируем теперь, каким образом в заданной грамматике может быть выведена из аксиомы S за число шагов, большее нуля, цепочка вида $wS\alpha$ для некоторых терминальной цепочки w и цепочки в объединенном алфавите α.

Так как вхождение S может возникнуть только после применения правила (3), а чтобы применить его, нужно сначала применить правило (1), то, чередуя применение этих правил $n \geqslant 1$ раз, получим вывод:

$$S \vdash abA \vdash abSaa \vdash ababAaa \vdash$$
$$\vdash ababSaaaa \vdash \ldots \vdash (ab)^n S (aa)^n. \quad (8.27)$$

Это значит, что все возможные пары (w, α), такие, что $S \overset{l}{\vdash}{}^* wS\alpha$ (с учетом уже ранее рассмотренного случая, когда обе цепочки пусты), есть $w = (ab)^n$ и $\alpha = (aa)^n$, $n \geqslant 0$.

Вычисляем множества $F_2(abA(aa)^n)$ и $F_2(\lambda(aa)^n)$ для различных $n \geqslant 0$. Первое множество, как нетрудно видеть, для всех n будет равно $\{ab\}$, а второе при $n = 0$ будет состоять из одной пустой цепочки, а при $n > 0$ — из цепочки aa, т.е. можно утверждать, что для всех $n \geqslant 0$ пересечение

$$F_2(abA(aa)^n) \cap F_2(\lambda(aa)^n) = \varnothing.$$

Итак, для нетерминала S (аксиомы грамматики G) условие теоремы 8.12 выполнено.

Нетерминал A. Рассматривая вывод (8.27), легко убедиться в том, что все пары (w, α), для которых имеет место левая выводимость $S \overset{l}{\vdash}{}^* wA\alpha$, есть $w = (ab)^n$, $\alpha = (aa)^{n-1}$ для всех $n \geqslant 1$.

Нетерминал A является левой частью двух правил вывода (3) и (4) с правыми частями Saa и b соответственно. Вычисляем соответствующие F_2-множества: $F_2(Saa(aa)^{n-1}) = \{ab, aa\}$, а $F_2(b(aa)^{n-1}) = \{b\}$, если $n = 1$, и $F_2(b(aa)^{n-1}) = \{ba\}$, если $n > 1$. Как видно, при всех n пересечение указанных множеств пусто. Тем самым доказано, что и для нетерминала A критерий теоремы 8.12 выполнен и заданная грамматика является $LL(2)$-грамматикой.

Полученный результат показывает также, что эта грамматика является $LL(2)$-грамматикой, не являясь $LL(1)$-грамматикой. Действительно, для нетерминала S получим

$$F_1(abA(aa)^n) = \{a\}$$

при всех $n \geqslant 0$, а $F_1(\lambda(aa)^n) = \{a\}$ при всех $n \geqslant 1$. Следовательно, при всех $n \geqslant 1$ указанные множества совпадут, что означает

невыполнение $LL(1)$-условия. Заметим, что для нетерминала A $LL(1)$-условие выполняется.

Результаты анализа представлены в табл. 8.1.

Таблица 8.1

Нетерминал	Аванцепочка						
	aa	ab	ba	bb	a	b	λ
S	2	1	–	–	–	–	2
A	3	3	4	–	–	4	–

Внутри таблицы указаны номера применяемых правил. Прочерк означает, что данная пара (нетерминал, аванцепочка) не определяет никакого правила, и возникновение такой пары в процессе анализа сигнализирует о синтаксической ошибке.

Мы видим, что в данном случае номер применяемого на каждом шаге правила вывода однозначно определяется только двумя факторами: нетерминалом в верхней ячейке магазина и аванцепочкой. $LL(k)$-грамматики, в которых выполняется такое условие, называют ***сильными $LL(k)$-грамматиками***.

Таблица подобного вида для сильной $LL(k)$-грамматики и является примером управляющей таблицы. Именно она обеспечивает в данном случае тот механизм управления выводом, который позволяет МП-автомату, построенному по заданной сильной $LL(k)$-грамматике, на каждом шаге однозначно выбирать правило вывода и для „правильной“ цепочки строить ее левый вывод, а для „неправильной“ — сигнализировать об ошибке. #

Следующий пример показывает, что существуют $LL(k)$-грамматики, не являющиеся сильными.

Пример 8.25. Зададим грамматику G_1 системой правил

$$S \to aAaa \,|\, bAba, \qquad (1)\,|\,(2)$$
$$A \to b \,|\, \lambda. \qquad (3)\,|\,(4)$$

Для этой грамматики легко построить все ее деревья вывода (рис. 8.38) и проверить $LL(k)$-условие.

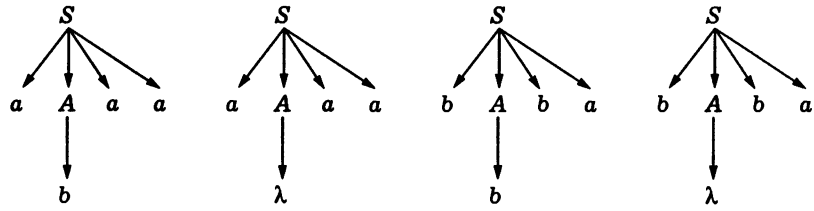


Рис. 8.38

В данном случае $S \overset{l}{\vdash}{}^* wS\alpha$ влечет $w = \lambda$, $\alpha = \lambda$ и $F_2(aAaa) = \{ab, aa\}$, а $F_2(bAba) = \{bb\}$, т.е. $F_2(aAaa) \cap F_2(bAba) = \varnothing$.

Из $S \overset{l}{\vdash}{}^* wA\alpha$ следует, что $w = a$ или $w = b$. Если $w = a$, то $\alpha = aa$ и $F_2(baa) = \{ba\}$, а $F_2(\lambda aa) = \{aa\}$, т.е. $F_2(baa) \cap F_2(\lambda a) = \varnothing$.

Если же $w = b$, то $\alpha = ba$ и $F_2(bba) = \{bb\}$, $F_2(\lambda ba) = \{ba\}$, и в этом случае также $F_2(bba) \cap F_2(\lambda ba) = \varnothing$. Таким образом, рассматриваемая грамматика есть $LL(2)$-грамматика.

Сведем полученные результаты в табл. 8.2.

Таблица 8.2

Левый контекст	Нетерминал	Аванцепочка	Применямое правило
λ	S	ab	(1)
		aa	(1)
λ	S	bb	(2)
a	A	ba	(3)
		aa	(4)
b	A	bb	(3)
		ba	(4)

Из табл. 8.2 следует, что одна и та же пара (A, ba) не определяет однозначно применяемого правила и требуется информация о левом контексте.

Из этой же таблицы видно, что данная грамматика не является $LL(1)$-грамматикой, поскольку, например, однобуквенная аванцепочка b вместе с нетерминалом A и левым контекстом b не определяет однозначно применямого правила. #

Таким образом, для сильных $LL(k)$-грамматик МП-автомат, который строится по КС-грамматике, может быть преобразован в синтаксический анализатор путем добавления к нему управляющей таблицы и выходной ленты (первоначально пустой), на которой автомат пишет номера правил в порядке их применения, а также сигнал ошибки. В результате анализа для правильной цепочки, записанной на входной ленте, на выходной ленте появится ее левый вывод (точнее, „протокол“ левого вывода в виде последовательности номеров применямых правил), а для неправильной цепочки на выходной ленте появится сообщение об ошибке (в виде какого-нибудь специального символа).

Представим теперь протоколы анализа некоторых цепочек в грамматике G примера 8.24.

Пусть $x = ababbaa$; имеем последовательность конфигураций (четвертая компонента конфигурации — содержимое выходной ленты):

$$(q,\ ababbaa,\ S,\ \lambda) \vdash (q,\ ababbaa,\ abA,\ 1) \vdash^2$$
$$\vdash^2 (q,\ abbaa,\ A,\ 1) \vdash (q,\ abbaa,\ Saa,\ 13) \vdash$$
$$\vdash (q,\ abbaa,\ abAaa,\ 131) \vdash^2 (q,\ baa,\ Aaa,\ 131) \vdash$$
$$\vdash (q,\ baa,\ baa,\ 1314) \vdash^3 (q,\ \lambda,\ \lambda,\ 1314).$$

Обратим внимание на то, что на каждом шаге вывода наш автомат-анализатор в существенном отличии от обычного распознавателя, который может легко „заблудиться“ (особенно при выборе альтернативы для нетерминала A), использует информацию управляющей таблицы.

Посмотрим, как отработает такой анализатор синтаксическую ошибку. Берем цепочку $y = abbb$, имеем

$$(q,\ abbb,\ S,\ \lambda) \vdash (q,\ abbb,\ abA,\ 1) \vdash^2$$
$$\vdash^2 (q,\ bb,\ A,\ 1) \vdash (q,\ bb,\ A,\ 1_{\text{„}}ERROR\text{“}),$$

так как пара $(A,\ bb)$ не определяет никакого правила.

Для не сильной $LL(k)$-грамматики учет левых контекстов в общем случае весьма сложен, и мы его здесь не рассматриваем*.

Приведем, наконец, простой пример грамматики с терминальным алфавитом $\{a, b, 0, 1\}$, не являющейся $LL(k)$-грамматикой ни для какого k:

$$S \to A \mid B,$$
$$A \to aAb \mid 0,$$
$$B \to aBbb \mid 1.$$

Для любого $k \geqslant 1$ имеем

$$S \overset{l}{\vdash} A \overset{l}{\vdash}{}^* a^k 0 b^k,$$
$$S \overset{l}{\vdash} B \overset{l}{\vdash}{}^* a^k 1 b^{2k}.$$

Одинаковый префикс a^k не дает возможности отождествить цепочки $\beta = A$ и $\gamma = B$, т.е. нет возможности обеспечить выбор альтернативы для нетерминала S.

Восходящий анализ. $LR(k)$-грамматики. Ключевым понятием при рассмотрении восходящего синтаксического анализа является понятие основы.

Определение 8.12. Пусть $G = (V,\ N,\ S,\ P)$ — КС-грамматика, ξ — цепочка в объединенном алфавите, $A \to \alpha \in P$. При $\alpha \sqsubseteq \xi$ вхождение α в ξ $(\beta,\ \alpha,\ \gamma)$ называют ***основой***, если цепочка $\beta A \gamma$ выводима из аксиомы S при условии, что цепочка ξ выводима из аксиомы S.

*См.: *Ахо А., Ульман Дж.*

Таким образом, по определению, основа — это такое вхождение правой части некоторого правила КС-грамматики в некоторую выводимую из аксиомы цепочку, что после замены этой правой части соответствующим нетерминалом полученная цепочка снова будет выводимой из аксиомы. Заметим, что для не выводимой из аксиомы цепочки любое вхождение правой части некоторого правила в нее можно считать основой.

Замечание 8.13. Ни в коем случае нельзя путать понятие основы в смысле определения 8.12 с понятием основы вхождения (см. **7.1**)!

Пример. Рассмотрим грамматику, задаваемую системой правил:

$$S \to Ac \,|\, Bd,$$
$$A \to aAb \,|\, ab,$$
$$B \to aBbb \,|\, abb.$$

Возьмем цепочку $x = aabbbbd$. Мы видим здесь вхождения двух правых частей правил — ab и abb. Вхождение первой цепочки не является основой. Действительно, заменяя ab нетерминалом A, получим цепочку $aAbbbd$. Для этой цепочки существует единственное правило $A \to aAb$, правая часть которого входит в нее. „Свертывая“ цепочку aAb в нетерминал A, получим $Abbd$ — слово, в которое не входит ни одна правая часть какого-либо правила данной грамматики. Следовательно, ни полученная цепочка, ни предшествующая ей цепочка не являются выводимыми из аксиомы.

Возвращаемся к цепочке x и, заменяя подцепочку abb нетерминалом B, получаем $aBbbd$, где единственное вхождение правой части правила — $aBbb$. Производя замену этого правила нетерминалом B, будем иметь цепочку Bd, которая „сворачивается“ в аксиому S. Итак, мы провели в обратном порядке вывод цепочки x из аксиомы:

$$S \vdash Bd \vdash aBbbd \vdash aabbbbd. \quad \#$$

Задача восходящего анализа и состоит в поиске „редукции“ исходной терминальной цепочки в аксиому, где на каждом шаге производится поиск основы в текущую цепочку с последующей заменой ее соответствующим нетерминалом, т.е. применяется „инверсия“ некоторого правила данной КС-грамматики: правая часть правила заменяется левой. Нетрудно понять, что если мы заменяем на каждом шаге такой „редукции“ самую левую основу, то восстанавливаем с конца к началу правый вывод исходной цепочки, а если самую правую основу, то „реконструируем“ левый вывод.

Как и в случае нисходящего анализа, здесь возникает проблема тупиков и связанная с ней проблема управления выводом. Мы рассмотрим некоторые простейшие методы построения „беступиковой редукции“, а именно такие методы определения основы, при применении которых попадание в тупик на некотором шаге редукции влечет синтаксическую неправильность входной цепочки.

Один из методов состоит в сопоставлении каждому правилу КС-грамматики некоторого бинарного отношения на множестве $(V \cup N)^*$. Вхождение правой части некоторого правила является основой тогда и только тогда, когда его крылья принадлежат сопоставленному данному правилу бинарному отношению. Проблему единственности определяемой таким образом основы, равно как и проблему характеристики класса КС-грамматик, допускающих такую „параметризацию“ бинарными отношениями, мы здесь не рассматриваем*.

Ограничимся простым примером.

Рассмотрим грамматику, порождающую множество непустых палиндромов в заданном алфавите V:

$$S \to \alpha S \alpha \,|\, \alpha\alpha \,|\, \alpha, \quad \alpha \in V.$$

Каждому правилу сопоставим одно и то же отношение

$$\{(u, v): |u| = |v|\},$$

*См.: *Глушков В.М., Цейтлин Г.Е., Ющенко Е.Л.*

так, что этому отношению принадлежат только пары с одинаковой длиной компонент.

Тогда осуществима беступиковая редукция любого палиндрома. Например:

$$abbccbba \dashv abbSbba \dashv abSba \dashv aSa \dashv S.$$

В данном случае на каждом шаге принадлежность крыльев вхождения правой части правила введенному отношению однозначно определяет основу, и получение тупиковой цепочки означает ее неправильность. Заметим, что без введенного таким образом управления выводом (т.е. выбором очередного правила) даже при „свертывании" правильной цепочки можно попасть в тупик:

$$abbccbba \dashv aSccbba \dashv aSSbba \dashv aSSSa.$$

Неверное определение основы на первом шаге редукции привело к тупиковой (т.е. не содержащей ни одного вхождения правой части какого-либо правила) цепочке, хотя входная цепочка является правильной.

Несколько более подробно рассмотрим класс КС-языков, допускающих беступиковый восходящий анализ и порождаемых так называемыми $LR(k)$-грамматиками.

Неформально ***$LR(k)$-грамматика***, $k \geqslant 0$, — это КС-грамматика, в которой основа однозначно определяется по левому крылу вхождения правой части некоторого правила вывода („левому контексту") и по не более чем k-буквенному префиксу правого крыла („правого контекста"). Строго понятие $LR(k)$-грамматики вводится весьма сложно, и мы его опускаем.

Оказывается, что класс языков, порождаемых $LR(k)$-грамматиками, может быть охарактеризован в терминах *детерминированных МП-автоматов* (*ДМП-автоматов*).

Прежде чем определять ДМП-автоматы, введем в рассмотрение модификацию МП-автомата, называемую ***расширенным МП-автоматом***, которая как раз ориентирована на представление восходящего синтаксического анализа.

Говоря коротко, отличие расширенного МП-автомата от обычного состоит в том, что в магазине можно заменять не только верхний символ, но и непустую цепочку символов, расположенную в верхних ячейках магазина, так, что первый ее символ есть верхний символ магазина, второй символ — символ, расположенный под верхним, и т.д.

Формально расширенный МП-автомат определяется совершенно так же, как обычный (см. определение 8.7), но функция переходов задается системой команд вида $qa\alpha \to r\beta$, где $a \in V \cup \{\lambda\}$; α и β — магазинные цепочки, причем α не пуста.

При записи системы команд расширенного МП-автомата условимся записывать магазинные цепочки от „дна“ магазина к его „верху“.

Можно доказать, что расширенные МП-автоматы допускают в точности тот же класс языков, что и обычные, т.е. можно доказать, что по любому расширенному МП-автомату можно построить эквивалентный ему МП-автомат в смысле определения 8.7*. Далее будем говорить „МП-автомат“, имея в виду расширенный МП-автомат.

Определение 8.13. МП-автомат называют ***детерминированным*** (***ДМП-автоматом***), если из любой его конфигурации непосредственно выводится не более чем одна конфигурация.

Нетрудно доказать следующее утверждение.

Теорема 8.13. МП-автомат является ДМП-автоматом тогда и только тогда, когда для любой упорядоченной пары $(q, \alpha) \in Q \times \Gamma^+$ верно одно из двух:

1) для любого $a \in V$ существует не более одной команды с левой частью $qa\alpha$, причем если для некоторого $a \in V$ такая команда существует, то не существует команды с левой частью $q\lambda\alpha$ и не существует ни одной команды с левой частью $qa'\alpha'$, где $a' \in \{a, \lambda\}$ и α' является *концом цепочки* α;

*См.: *Ахо А., Ульман Дж.*

2) существует не более одной команды с левой частью $q\lambda\alpha$, причем если такая команда существует, то не существует ни одной команды с левой частью $qa\alpha'$, где $a \in V \cup \{\lambda\}$ и α' — конец цепочки α, и не существует ни одной команды с левой частью $qa\alpha$, $a \in V$. #

Связь между ДМП-автоматами и $LR(k)$-грамматиками устанавливается следующей теоремой.

Теорема 8.14. Язык допускается ДМП-автоматом тогда и только тогда, когда он порождается некоторой $LR(k)$-грамматикой.

Определение 8.14. Язык называют ***детерминированным КС-языком***, если он допускается некоторым ДМП-автоматом.

Заметим, что поскольку для произвольного МП-автомата (в отличие от конечных автоматов), вообще говоря, не может быть построен эквивалентный ему ДМП-автомат, то класс детерминированных КС-языков строго содержится в классе всех КС-языков.

Для детерминированных КС-языков может быть предложена стратегия восходящего анализа, называемая ***стратегией „перенос — свертка“***. Суть ее состоит в следующем: входная цепочка посимвольно переписывается в магазин до тех пор, пока в магазине не сформируется основа (однозначно определяемая прочитанной частью цепочки и не более чем k буквами непрочитанной части для некоторого фиксированного $k \geqslant 0$).

Сформированная в магазине основа заменяется соответствующим нетерминалом. Если после этой замены в верхних ячейках магазина опять получилась основа, то она вновь „сворачивается“ в нетерминал, и так до тех пор, пока не окажется, что либо вся входная цепочка прочитана, а в магазине кроме начального символа в верхней ячейке осталась аксиома грамматики, либо цепочка еще не прочитана, а в верху магазина основы нет, либо, наконец, цепочка прочитана, а в верху магазина ни аксиомы, ни основы нет.

В первом случае мы имеем „допуск“ цепочки (успешный результат синтаксического анализа), во втором необходимо возобновить процесс „переноса“, т.е. продолжить переписывание входной цепочки в магазин, а в третьем констатировать „недопуск“ (синтаксическую ошибку).

Согласно теореме 8.14, подобный алгоритм может быть запрограммирован в системе команд некоторого ДМП-автомата (точнее, ДМП-автомата с выходной лентой, на которой записываются номера применяемых правил грамматики и/или сигнал ошибки).

Строгая теория $LR(k)$-грамматик весьма сложна и никак не может быть даже фрагментарно изложена в рамках настоящего учебника.

Мы рассмотрим в заключение один пример.

Пример 8.26. Приведем ДМП-автомат для анализа языка правильных скобочных структур, порождаемых грамматикой $S \to ()\,|\,(S)\,|\,SS$. Точнее, мы рассмотрим язык правильных скобочных структур с „концевым маркером“ $\star$: записывая на ленту правильную скобочную структуру, в конце ставим символ $\star$.

Система команд анализирующего ДМП-автомата имеет следующий вид:

$$q_0 a \square \to q_1 \square a, \qquad (1)$$
$$q_0 a a' \to q_1 a' a, \qquad (2)$$
$$q_1 \lambda() \to q_1 S, \qquad (3)$$
$$q_1 \lambda(S) \to q_1 S, \qquad (4)$$
$$q_1 \lambda S S \to q_1 S, \qquad (5)$$
$$q_1 \lambda \gamma \to q_0 \gamma, \qquad (6)$$
$$q_1 \lambda \square a \to q_0 \square a, \qquad (7)$$
$$q_1 a \square S \to q_1 \square S a, \qquad (8)$$
$$q_1 \star \square S \to q_2 \square. \qquad (9)$$

В этой системе команд q_0 и q_2 — соответственно начальное заключительное состояния; $a \in \{(,)\}$; $a' \in \{(,),S\}$; $\square$ — на-

чальный символ магазина (называемый иногда „маркером дна магазина“); γ — произвольная цепочка, длина которой равна 3 и которая не равна (S) и не оканчивается цепочкой $()$ или SS.

Легко убедиться в том, что записанная система команд действительно определяет ДМП-автомат (см. теорему 8.13).

Команды (1) и (2) обеспечивают перенос в магазин входной цепочки, команды (3)–(5) — команды свертки, команды (6)–(8) обеспечивают переход из фазы свертки в фазу переноса, если в магазине нет основы, а команда (9) переводит автомат в заключительное („допускающее“) состояние в случае правильной входной цепочки.

Проанализируем на данном ДМП-автомате цепочку

$$x = (())()\star,$$

которая является правильной скобочной структурой.

Имеем последовательность конфигураций*:

$$\begin{aligned}
&\langle q_0,(())()\star,\square\rangle \vdash \langle q_1,())()\star,\square(\rangle \vdash \\
&\quad\vdash \langle q_0,())()\star,\square(\rangle \vdash \langle q_1,))()\star,\square((\rangle \vdash \langle q_0,))()\star,\square((\rangle \vdash \\
&\quad\vdash \langle q_1,)()\star,\square(()\rangle \vdash \langle q_1,)()\star,\square(S\rangle \vdash \langle q_0,)()\star,\square(S\rangle \vdash \\
&\quad\vdash \langle q_1,()\star,\square(S)\rangle \vdash \langle q_1,()\star,\square S\rangle \vdash \langle q_1,)\star,\square S(\rangle \vdash \\
&\quad\vdash \langle q_0,)\star,\square S(\rangle \vdash \langle q_1,\star,\square S(()\rangle \vdash \langle q_1,\star,\square SS\rangle \vdash \\
&\quad\vdash \langle q_1,\star,\square S\rangle \vdash \langle q_2,\lambda,\square\rangle. \quad \#
\end{aligned}$$

Рассмотрение перехода от $LR(k)$-грамматик к ДМП-автоматам и наоборот выходит за рамки нашего изложения. На простейшем примере мы продемонстрировали только работу детерминированного восходящего анализатора.

*Здесь мы использовали угловые скобки $\langle\,\rangle$ для обозначения конфигураций, чтобы не путать с круглыми скобками $(\,)$, являющимися терминалами грамматики.

Дополнение 8.2. Семантика формальных языков

Классическая теория формальных языков, как уже отмечалось, занималась исключительно *синтаксисом языков*, изучая методы порождения и распознавания множеств *слов*. *Семантика* формальных языков, сравнительно молодая ветвь теории, занимается способами сопоставления некоего „смысла“ словам (цепочкам) языка.

Необходимость в построении точного математического понятия „смысла“ диктуется развитием информационных технологий, прежде всего технологии проектирования компиляторов.

Рассмотренные выше языковые модели определенным образом связаны с этапами этой технологии.

Текст входной программы, как известно, анализируется в несколько проходов. На первом проходе производится лексический анализ, а именно проверяется правильность простейших элементов текста, называемых лексемами. Примерами лексем могут служить идентификаторы и константы, разрешенные синтаксисом входного языка программирования. В процессе лексического анализа не проверяется синтаксическая правильность всей программы в целом, а проверяется только синтаксическая правильность лексем (в частности, правильность написания идентификаторов и констант). Так как лексемы обычно являются элементами некоторого *регулярного языка*, то базовой моделью для лексического анализатора является модель *конечного автомата*.

Если текст программы успешно прошел этап лексического анализа, то тогда проверяется его глобальная синтаксическая правильность. При этом каждая лексема рассматривается как буква. Здесь применяются методы синтаксического анализа, в частности рассмотренные выше (см. **Д.8.1**). В предположении, что синтаксис языка программирования описан *КС-граммати-*

кой, основой для построения синтаксических анализаторов, как мы уже видели, выбирается модель *МП-автомата*.

По завершении синтаксического анализа строится *дерево вывода* входной программы. После этого переходят к этапу генерации объектного кода, т.е. внутреннего машинного представления входного текста. Это значит, что выполняется перевод с некоторого языка программирования на язык машинных кодов. Но чтобы выполнить перевод текста на другой язык, необходимо каким-то образом понять его „смысл“. Следовательно, анализ уже синтаксически проверенной программы с точки зрения ее „смысла“ (семантический анализ) необходимо предшествует самой генерации объектного кода. И прежде всего необходимо уточнить математически, что такое „смысл“ (как раньше мы математически определяли синтаксис в терминах грамматик и автоматов).

В этом дополнении мы рассмотрим некоторые методы формального (математического) определения семантики для *КС-языков*. Тем самым мы всюду в дальнейшем предполагаем, что язык, семантика которого определяется, может быть задан некоторой КС-грамматикой.

Сразу же необходимо сделать уточнение. Как мы уже заметили ранее, исследуя явление *неоднозначности* в КС-языках, „смысл“ следует сопоставлять не самим словам языка, а деревьям их вывода: меняя дерево вывода данной цепочки, мы меняем и ее „смысл“, понимаем ее по-другому (см. **8.1**). Далее, можно сопоставлять „смысл“ не только словам языка, точнее, деревьям вывода этих слов из аксиомы грамматики, но и так называемым фразам языка — терминальным цепочкам, выводимым из разных нетерминалов грамматики.

Например, фраза „...а так как мне бумаги не хватило“ не является законченным предложением русского языка, но имеет, очевидно, какой-то „смысл“. Точно так же оператор присваивания, „вынутый“ из какой-то программы на каком-то языке программирования, не является „программой“, не может рассматриваться как элемент данного языка, но мы

в состоянии сопоставить ему тот или иной „смысл“. Тем самым возникает идея определить „смысл“ через отображение множества „синтаксических объектов“ — деревьев выводов фраз языка в некоторое „предметное множество“, множество „семантических объектов“.

Здесь мы ограничимся крайне элементарным введением в формальное описание семантики КС-языков и опишем некоторый простейший „кирпичик“ формальной семантики языков программирования.

Начнем с формального определения фразы КС-языка.

Определение 8.15. Пусть $G = (V, N, S, P)$ — КС-грамматика. Терминальную цепочку x называют ***фразой языка*** $L(G)$, если $A \vdash^* x$ для некоторого нетерминала $A \in N$.

Пусть $T(x)$ — дерево вывода фразы x с корневым нетерминалом A. Возьмем в $T(x)$ некоторое поддерево с корневым нетерминалом B. Выводимую из этого вхождения B терминальную цепочку называют ***подфразой фразы*** x. Если вершина дерева $T(x)$, соответствующая данному вхождению B, имеет глубину 1 (или, что равносильно, уровень, на единицу меньший уровня корня A), то данная подфраза называется ***подфразой первого уровня*** фразы x.

Рассмотрим в качестве примера следующую грамматику арифметических выражений:

$$Expr \to Atom \,|\, (Expr + Expr) \,|\, (Expr * Expr),$$
$$Atom \to a_1 \,|\, \ldots \,|\, a_n.$$

Предполагается, что вместо вхождения нетерминала $Atom$ может быть подставлен любой символ некоторого алфавита $V = \{a_1, \ldots, a_n\}$ (атом в арифметическом выражении может быть либо переменным, либо константой).

Нарисуем дерево вывода выражения

$$(a + (b * (c + (d * (e + g))))),$$

где a, b, c, d, e, g — некоторые атомы из V (рис. 8.39).

Это выражение в качестве подфраз первого уровня имеет цепочки a и

$$(b*(c+(d*(e+g)))).$$

Вторая фраза имеет подфразы первого уровня b и

$$(c+(d*(e+g)))$$

и т.д. Заметим, что фраза a, выводимая из самого левого узла *Expr* глубины 1 имеет в качестве подфразы первого уровня ту же цепочку a, т.е. подфраза первого уровня может совпадать с самой исходной фразой.

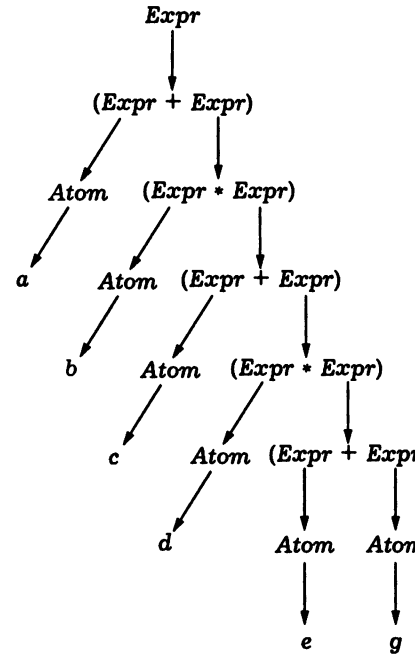


Рис. 8.39

Множество всех фраз языка L обозначим через $PH(L)$, а множество всех деревьев вывода фраз L — через $TPH(L)$. Для однозначного языка L эти множества эквивалентны.

Определение 8.16. ***Семантическая функция языка*** L есть отображение

$$SEM(L)\colon TPH(L) \to \mathbb{U}(L)$$

множества всех деревьев вывода фраз L в некоторое множество $\mathbb{U}(L)$, называемое универсумом (предметной областью, семантической областью, областью интерпретации) языка L.

Замечание 8.14. Ради общности следовало бы определить семантическую функцию как частичное отображение, но мы не будем этого делать, вводя для „бессмысленных“ фраз специальный „неопределенный“ элемент в универсум („нуль“ предметной области, „неопределенный смысл“). Заметим также, что

для однозначного языка семантическая функция может быть определена как отображение из множества фраз языка. #

Способы определения семантической функции могут быть различными. Нас будет интересовать важный частный случай, когда значение семантической функции на некоторой фразе (под фразой здесь и далее понимается дерево ее вывода, причем разные деревья для одной и той же цепочки рассматриваются как разные фразы) определяется однозначно через значение этой функции на подфразах первого уровня. Более формально: представим фразу x синтаксически как „соединение“ своих подфраз первого уровня:

$$x = \varphi(y_1, \ldots, y_m) = u_1 y_1 \ldots u_m y_m u_{m+1}, \tag{8.28}$$

где u_1, ..., u_{m+1} — некоторые терминальные цепочки. Например, для определенного выше языка арифметических выражений фраза, дерево вывода которой изображено на рис. 8.39 раскладывается следующим образом:

$$(a+(b*(c+(d*(e+g))))) = u_1 a u_2 (b*(c+(d*(e+g))))u_3,$$

где u_1 есть цепочка (, u_2 — цепочка +, а u_3 — цепочка).

Таким образом, любая фраза представляется, вообще говоря, не как простое соединение своих подфраз первого уровня, а как соединение с „прослойками“ в виде определенных терминальных цепочек. Операцию φ в (8.28), задающую такие прослойки, называют часто ***конкатенарной операцией***. Эта операция определяется грамматикой языка.

Тогда предполагается, что если фраза x представлена в виде (8.28), то определено отображение ψ: $\mathbb{U}(L)^m \to \mathbb{U}(L)$ (т.е. некоторая операция на предметной области), такое, что

$$SEM(L)(x) = \psi\big(SEM(L)(y_1), \ldots, SEM(L)(y_m)\big). \tag{8.29}$$

Это ограничение, накладываемое на семантическую функцию, назовем ***принципом гомоморфной интерпретации***.

Примем теперь некоторые соглашения об обозначениях. Семантическую функцию языка L будем обозначать $[\![\cdot]\!]_L$, опуская индекс, указывающий на язык, если это не ведет к недоразумению. Тогда равенство (8.29) можно переписать в виде

$$[\![x]\!] = [\![\varphi(y_1,\ldots,y_m)]\!] = \psi([\![y_1]\!],\ldots,[\![y_m]\!]).$$

Подобные определения семантики фразы через семантику ее подфраз первого уровня называют семантическими правилами языка. Семантические правила соответствуют синтаксическим правилам — правилам исходной КС-грамматики. Так, для приведенной выше грамматики арифметических выражений, полагая $\mathbb{U} = \mathbb{R}$, можно записать следующие семантические правила:

$$[\![Expr]\!] = [\![Expr]\!] + [\![Expr]\!]$$

(для синтаксического правила $Expr \to (Expr + Expr)$), т.е. здесь конкатенарной операции φ, такой, что $\varphi(u,v) = (u+v)$ для любых двух фраз u и v, сопоставляется обычное арифметическое сложение;

$$[\![Expr]\!] = [\![Expr]\!] * [\![Expr]\!]$$

(для синтаксического правила $Expr \to (Expr * Expr)$) операция на универсуме — арифметическое умножение;

$$[\![Expr]\!] = [\![Atom]\!]$$

(для синтаксического правила $Expr \to Atom$);

$$[\![Atom]\!] = [\![a_i]\!]$$

(для синтаксического правила $Atom \to a_i$, $i = \overline{1,n}$).

Строго говоря, в первых двух правилах мы должны различать три разных вхождения одного и того же нетерминала $Expr$, так как они соответствуют разным деревьям.

Семантическая функция языка арифметических выражений будет определена, если мы зададим значение этой функции на атомах: это естественно ассоциируется с хорошо знакомой процедурой вычисления значения арифметического выражения при подстановке вместо входящих в него переменных конкретных числовых значений (при этом не исключается, что атом может быть константой — обозначением конкретного числа; в таком случае само синтаксическое правило задает сразу значение семантической функции на данном атоме).

Так, для приведенного выше выражения, полагая $[\![a]\!] = 1$, $[\![b]\!] = 2$, $[\![c]\!] = 3$, $[\![d]\!] = 4$, $[\![e]\!] = 5$, $[\![g]\!] = 6$, получим значение всего выражения, равное 95.

Таким образом, в этом конкретном случае семантика фразы есть числовое значение представленного данной фразой арифметического выражения. Мы рассмотрели в качестве универсума множество вещественных чисел и получили одну семантику. Задав универсум как-нибудь иначе (например, как множество комплексных чисел или множество функций некоторого класса), получим совсем другую семантику. Для одного и того же синтаксиса, следовательно, могут быть определены различные семантики (семантические функции).

Разобранный выше на примере языка арифметических выражений подход к определению семантики называют иногда ***экстенсиональным подходом***. Суть его состоит в том, что явно определяется универсум как множество „внеязыковых“ объектов (экстенсионалов) и каждой языковой фразе сопоставляется некоторый экстенсионал. Для разобранного выше примера экстенсионал — это вещественное число. Слово „внеязыковых“ взято в кавычки потому, что универсум сам может быть некоторым языком (выступающим тогда по отношению к исходному языку как метаязык). Не исключено даже, что метаязык совпадает с самим определяемым языком — примером могут служить всевозможные толковые словари.

Существует и другой подход к определению семантики, называемый ***интенсиональным***, одной из разновидностей кото-

рого является ***аксиоматический метод определения семантики языка.***

Аксиоматический метод предполагает рассмотрение исходного, „объектного“ языка, семантика которого определяется, как ***формальной теории*** (или ***формальной системы***). Не давая строгого определения формальной теории в его общности, поясним его суть и рассмотрим пример.

Формальная теория задается как некоторый язык, цепочки которого называются в этом случае ***утверждениями*** (или ***предложениями***). В этом языке определяется подъязык так называемых доказуемых утверждений: задается некоторое начальное множество утверждений, которые считаются априори доказанными (множество ***аксиом теории***), и задается некоторое множество ***правил вывода***, применяя которые к некоторым утверждениям (в частности, уже доказанным), можно получать новые утверждения. Если мы применяем правило вывода к доказанному утверждению, то получаем новое доказанное утверждение. Утверждение, которое таким образом может быть выведено из аксиом, называют ***теоремой*** данной теории. Утверждение считается имеющим смысл, если оно есть либо аксиома, либо теорема данной теории. В отличие от экстенсионального подхода при интенсиональном подходе априори не определяется никакой универсум, и при таком подходе к определению семантики „иметь смысл“ означает „быть теоремой или аксиомой данной теории“.

Вернемся к рассмотренному выше языку арифметических выражений. Зададим его в виде формальной теории следующего вида:

1) аксиома — любой атом $a_1, \ldots, a_n$;

2) правила вывода:

$$e_1, e_2 \Rightarrow (e_1 + e_2), \qquad (1)$$

$$e_1, e_2 \Rightarrow (e_1 * e_2), \qquad (2)$$

т.е. если e_1 и e_2 доказаны, то доказано утверждение $(e_1 + e_2)$ и утверждение $(e_1 * e_2)$.

Построим, например, доказательство записанного выше выражения:

$a,\ b,\ c,\ d,\ e,\ g$	— аксиомы;
$(e+g)$	— правило вывода (1);
$(d*(e+g))$	— (2);
$(c+(d*(e+g)))$	— (1);
$(b*(c+(d*(e+g))))$	— (2);
$(a+(b*(c+(d*(e+g)))))$	— (1).

Нетрудно видеть, что мы определили тот же язык другим способом. Множество доказуемых утверждений совпадает здесь с множеством всех утверждений, и, таким образом, семантика в данном случае совпала с синтаксисом: утверждение имеет смысл тогда и только тогда, когда может быть доказано, т.е. тогда и только тогда, когда является арифметическим выражением, порождаемым приведенной выше грамматикой.

Замечание 8.15. В этой связи полезно заметить, что и правила вывода формальной теории следует трактовать как просто „правила замены“ (в полной аналогии с порождающими грамматиками), согласно которым разрешается от определенных цепочек (в левой части правила) переходить к новым цепочкам (в правой его части).

Тогда по правилам вывода можно получать из цепочек, не обязательно теорем или аксиом, какие-то другие цепочки, т.е. строить выводы, не являющиеся доказательствами. Это опять-таки аналогично грамматикам, в которых применение продукций, вообще говоря, не обязано быть выводом из аксиомы.

Такая широкая и чисто синтаксическая трактовка правил вывода позволяет в теории формальных систем и в связанной с нею теории доказательств строить выводы из „гипотез“ — цепочек, которые предполагаются доказанными. Если затем удается действительно доказать их, то построенное первоначально „относительное“ доказательство превратится в „абсолютное“ (т.е. начинающееся с аксиом). #

Построив аксиоматическую семантику языка, мы можем на этой базе определить семантику уже экстенсионально, положив, что экстенсионал утверждения есть множество всех его доказательств (если утверждение не имеет доказательства, то его экстенсионал считается не определенным — формально мы включаем тогда в предметную область специальный „неопределенный элемент“ с тем, как уже отмечалось, чтобы можно было вводимую семантическую функцию считать отображением).

В заключение рассмотрим экстенсиональное определение семантики простейшего языка программирования, который будем называть MILAN (MIniLANguage). Синтаксис языка определим посредством форм Бэкуса — Наура:

(программа) ::= (последовательность_операторов)
(последовательность_операторов) ::= (оператор) |
(оператор) (последовательность_операторов)
(оператор) ::= (присваивание) | (условный_переход) | (цикл)
(присваивание) ::= (переменное) := (терм)
(условный_переход) ::= *if* (условие) *then*
(последовательность_операторов)
else (последовательность_операторов) |
if (условие) *then* (последовательность_операторов)
(цикл) ::= *while* (условие) *do*
(последовательность_операторов) *end*
(переменное) ::= $x_1|\ldots|x_n$
(терм) ::= (функциональный_символ)
((последовательность_термов)) |
(переменное) | (константа)
(последовательность_термов) ::= (терм) |
(терм) (последовательность_термов)
(функциональный_символ) ::= $f_1|\ldots|f_m$
(константа) ::= $c_1|\ldots|c_k$
(условие) ::= (предикатный_символ)
((последовательность_термов))
(предикатный_символ) ::= $p_1|\ldots p_s$

Заданная таким образом КС-грамматика определяет так называемый абстрактный синтаксис MILANa: мы игнорируем некоторые синтаксические детали, такие, как слова *begin*, *end*, обрамляющие последовательность операторов, а также то, что операторы разделяются точкой с запятой, что между термами в последовательности термов ставится запятая и т.п. Мы также не уточняем структуру переменных (идентификаторов), констант, функциональных и предикатных символов, считая, что эти нетерминалы „пробегают“ каждый свой алфавит.

Чтобы формально описать экстенсиональную семантику введенного посредством написанных выше синтаксических правил языка, нужно определить универсум. В зависимости от того, какой универсум рассматривается, в рамках экстенсионального подхода различают 1) ***денотационную семантику***, 2) ***операциональную семантику*** и 3) ***трансформационную семантику*** (***семантику смешанных вычислений***). Здесь мы рассмотрим только денотационную семантику.

В денотационной семантике языка, подобного MILANy, универсум определяется как множество преобразователей состояний памяти. К построению этого множества и переходим.

Определение 8.17. ***Состояние памяти*** — это произвольное *отображение* множества переменных I языка программирования в множество D данных (значений).

В силу сформулированного определения мы отождествляем память с множеством переменных (идентификаторов, „ячеек“, имен) данного языка программирования (конкретно — языка MILAN), а состояние памяти σ есть отображение вида

$$\sigma\colon I \to D,$$

сопоставляющее каждой переменной значение, принадлежащее некоторому множеству значений (данных) D. Последнее можно рассматривать как объединение некоторого *семейства множеств*, служащих *носителями многосортной алгебры* данных

языка. Их можно отождествить с типами данных, используемыми в рассматриваемом языке: числами, массивами, строками, стуктурами и т.п. В множество D включен также ***неопределенный элемент*** $\mathbb{O}$ (***неопределенное значение***).

Множество всех состояний памяти обозначим Σ.

Определение 8.18. ***Преобразователь состояний памяти*** — это произвольное отображение множества Σ состояний памяти в себя.

Множество всех преобразователей состояний памяти обозначим $(\Sigma \to \Sigma)$.

На множестве Σ определяется структура *индуктивного упорядоченного множества* (см. **1.8**).

Область определенности состояния памяти σ — это множество, обозначаемое $D(\sigma)$, всех переменных x, таких, что $\sigma(x) \neq \mathbb{O}$.

Тогда положим для двух произвольных состояний σ и ρ

$$\sigma \leqslant \rho \Leftrightarrow (D(\sigma) \subseteq D(\rho))\&((\forall x \in D(\sigma))(\sigma(x) = \rho(x))).$$

Очевидно, что отношение $\leqslant$ есть *отношение порядка*. Далее, множество Σ имеет по отношению $\leqslant$ *наименьший элемент*, а именно такое состояние $\mathbb{O}$, что $(\forall x \in I)(\mathbb{O}(x) = \mathbb{O})$. Его называют ***всюду неопределенным состоянием памяти***; ясно, что $D(\mathbb{O}) = \varnothing$. Кроме того, для любой неубывающей последовательности состояний памяти

$$\sigma_1 \leqslant \sigma_2 \leqslant \ldots \leqslant \sigma_n \leqslant \ldots$$

состояние σ_0, такое, что

$$\sigma_0(x) = \sigma_n(x) \Leftrightarrow x \in D(\sigma_n),$$

есть *точная верхняя грань* этой *последовательности*, причем

$$D(\sigma_0) = \bigcup_{n \geqslant 1} D(\sigma_n).$$

Таким образом, множество состояний памяти Σ, снабженное отношением порядка $\leqslant$, является индуктивным упорядоченным множеством.

С учетом результатов, полученных в **4.5** (см. теорему 4.11), отсюда следует, что и множество преобразователей состояний $(\Sigma \to \Sigma)$ есть индуктивное упорядоченное множество. В этом множестве отношение порядка $\leqslant$ определяется условием

$$f \leqslant g \Leftrightarrow (\forall \sigma \in \Sigma)(f(\sigma) \leqslant g(\sigma)),$$

наименьшим элементом является ***стирающий преобразователь*** $\mathbf{0}$, такой, что $\mathbf{0}(\sigma) = \mathbb{0}$ для всякого $\sigma \in \Sigma$, а точной верхней гранью неубывающей последовательности преобразователей состояний $f_1 \leqslant f_2 \leqslant \ldots f_n \leqslant \ldots$ является преобразователь f_0, определяемый следующим образом:

$$(\forall \sigma \in \Sigma)(\forall x \in I)(f_0(\sigma)(x) = f_n(\sigma)(x) \Leftrightarrow x \in D(f_n(\sigma))),$$

причем

$$D(f_0(\sigma)) = \bigcup_{n \geqslant 1} D(f_n(\sigma)).$$

Более того, оказывается, что ***композиция*** преобразователей состояний (как *отображений*) ***непрерывна*** в смысле сохранения точных верхних граней (см. **1.8**), точнее, для любой неубывающей последовательности преобразователей состояний

$$f_1 \leqslant f_2 \leqslant \ldots f_n \leqslant \ldots$$

и произвольного преобразователя состояний g

$$g \circ \sup f_n = \sup(g \circ f_n),$$
$$\sup f_n \circ g = \sup(f_n \circ g).$$

Докажем первое из этих равенств. Для произвольного $\sigma \in \Sigma$ имеем

$$g \circ \sup f_n(\sigma) = \sup f_n(g(\sigma)).$$

Тогда для всякого $x \in D(g \circ f_n(\sigma))$

$$g \circ \sup f_n(\sigma)(x) = \sup f_n(g(\sigma))(x) = f_n(g(\sigma))(x) = g \circ f_n(\sigma)(x),$$

т.е. поскольку, как можно показать, композиция *монотонна* в смысле определения, данного в **1.8**, то

$$D(g \circ \sup f_n(\sigma)) = \bigcup_{n \geqslant 1} D(g \circ f_n)(\sigma)$$

и

$$g \circ \sup f_n = \sup(g \circ f_n).$$

Имея в виду все рассмотренные выше свойства множества $(\Sigma \to \Sigma)$, определим денотационную семантику языка MILAN.

Семантическая функция есть отображение $[\![\cdot]\!]$ множества фраз в множество $(\Sigma \to \Sigma)$ преобразователей состояний памяти.

1. Семантика оператора присваивания:

$$[\![x := t]\!](\sigma)(y) \begin{cases} \sigma(y), & y \neq x; \\ t^\sigma, & y = x, \end{cases}$$

где через t^σ обозначено значение терма t в состоянии σ, равное $\mathbb{O}$, если хотя бы для одного переменного, входящего в терм, его значение не определено, и равное значению терма при подстановке на место каждого его переменного x его значения $\sigma(x)$.

Суть записанного выше семантического правила очень проста: оператору присваивания $x := t$ сопоставляется преобразователь состояний, меняющий состояние таким образом, что все переменные, кроме x, сохраняют свои значения, а новое значение переменного x есть значение правой части оператора присваивания (терма t) в состоянии σ.

2. Семантика оператора условного перехода:

$$[\![if\ p(t_1,\ldots,t_n)\ then\ S_1\ else\ S_2]\!](\sigma) =$$

$$= \begin{cases} [\![S_1]\!](\sigma), & p(t_1{}^\sigma,\ldots,t_n{}^\sigma) = 1; \\ [\![S_2]\!](\sigma), & p(t_1{}^\sigma,\ldots,t_n{}^\sigma) = 0, \end{cases}$$

Это правило написано в предположении, что значения всех термов в состоянии σ определены. В противном случае оператору условного перехода сопоставляется стирающий преобразователь $\mathbf{0}$. Подобное же соглашение принимается и далее в аналогичных ситуациях.

Семантика условного перехода без *else*-альтернативы записывается аналогично, но при условии ложности предиката p в состоянии σ берется тождественный преобразователь состояний ID.

3. Семантика цикла:

$$[\![while\, p(t_1,\ldots,t_n)\, do\, S\, end]\!](\sigma) =$$

$$= \begin{cases} [\![S]\!] \circ [\![while\, p(t_1,\ldots,t_n)\, do\, S\, end]\!](\sigma), & p(t_1{}^\sigma,\ldots,t_n{}^\sigma) = 1; \\ ID(\sigma), & p(t_1{}^\sigma,\ldots,t_n{}^\sigma) = 0. \end{cases}$$

Таким образом, семантика цикла определяется как решение уравнения. Это определение корректно, так как правая часть уравнения есть непрерывное отображение множества $(\Sigma \to \Sigma)$ в себя (что следует из доказанной выше непрерывности композиции преобразователей состояний и очевидной непрерывности тождественного отображения).

4. Семантика последовательности операторов:

$$[\![S_1; S_2]\!] = [\![S_1]\!] \circ [\![S_2]\!].$$

Определенная таким образом формальная семантика языка MILAN становится базой для семантического анализа программ, т.е. для строгого математического доказательства утверждений о программах. Этот анализ, проводимый технологически после синтаксического анализа программы и получения ее дерева вывода, основан на результате, известном под

названием принципа индукции по неподвижной точке*. Приведем формулировку этого принципа без доказательства.

Принцип индукции по неподвижной точке. Пусть $\mathcal{P}$ есть некоторое утверждение о программах, $\mathcal{F}$ — непрерывное отображение множества $(\Sigma \to \Sigma)$ в себя, f — наименьшая *неподвижная точка отображения* $\mathcal{F}$ или, что то же самое, *наименьшее решение уравнения* $X = \mathcal{F}(X)$.

Тогда, если $\mathcal{P}(f^{(0)})$ истинно и для каждого i из истинности $\mathcal{P}(f^{(i)})$ следует истинность $\mathcal{P}(f^{(i+1)})$, то истинно $\mathcal{P}(f)$. (Под $f^{(i)}$ понимается i-е приближение наименьшей неподвижной точки: $f^{(0)} = \mathbf{0}$, $f^{(i)} = \mathcal{F}(f^{(i-1)})$ (см. **1.8**).)

Проиллюстрируем применение этого принципа на примере анализа простейшей программы на языке MILAN.

Рассмотрим программу вычисления факториала:

```
begin
  x := n;
  y := 0;
  z := 1;
  while (y ≠ x) do
    y := y + 1;
    z := z * y;
  end
end
```

Здесь предполагается, что все переменные принимают целые значения. Букву n следует понимать не как переменную программы, а как условное обозначение произвольного натурального числа, „засылаемого“ в „ячейку“ x. Мы вынуждены прибегать к столь неэстетичному (с программистской точки зрения) вводу исходного значения, так как наш простенький язык не имеет средств ввода/вывода.

Докажем следующее утверждение $\mathcal{P}$ об этой программе: если после выполнения программы состояние памяти σ таково, что $\sigma(x) \neq \mathbb{0}$ и $\sigma(z) \neq \mathbb{0}$, то $\sigma(z) = \sigma(x)!$

*См.: *Донохоу П.*

Заметим, что мы не определяли в языке операторов ввода и вывода и, следовательно, начальное состояние памяти задается априори (некоторым „оракулом“), а вопрос, как это состояние сформировано, остается за пределами формализации. В нашем примере речь идет о задании начального значения переменной n.

Проанализируем теперь, что означает в данном случае индукция по неподвижной точке.

Ясно, что перед выполнением цикла состояние памяти σ_0 таково, что всегда $\sigma_0(z) = \sigma_0(y)!$. Нам нужно доказать, что если выполнение цикла заканчивается после конечного числа итераций, то в результирующем состоянии σ будем иметь $\sigma(z) = \sigma(x)! = \sigma(y)!$.

Нам бу́дет удобно определить операцию p-дизъюнкции преобразователей состояний памяти, где $p(t_1, \ldots, t_n)$ — некоторое условие языка MILAN.

Определение 8.19. Назовем ***p-дизъюнкцией*** преобразователей состояний памяти f и g преобразователь, обозначаемый ${}_p(f \vee g)$ и определяемый следующим образом:

$$(\forall \nu \in \Sigma)\, {}_p(f \vee g)(\nu) = \begin{cases} \mathbb{O}, & \text{значение предиката } p \\ & \text{в состоянии } \nu \text{ не определено;} \\ f(\nu), & p_\nu = 1; \\ g(\nu), & p_\nu = 0, \end{cases}$$

(через p_ν обозначено значение условия, или предиката, p в состоянии ν: $p_\nu = p(t_1^\nu, \ldots, t_n^\nu)$).

В соответствии с тем, как мы определили выше, Преобразователь состояний f, соответствующий циклу, есть наименьшая неподвижная точка уравнения $f = F(f)$, где функция F может быть представлена в виде

$$F(g)(\nu) = {}_{\neg p}\big(ID \vee [\![S]\!] \circ g\big)(\nu)$$

(для условия p и тела S цикла *while*).

Согласно алгоритму вычисления наименьшей неподвижной точки (см. **1.8**), $f = \sup\{f^{(i)}\colon i \geqslant 0\}$, где $f^{(0)} = \mathbf{0}$ — стирающий преобразователь.

Вычисляя первое приближение наименьшей неподвижной точки, получаем

$$f^{(1)} = {\neg_p}(ID \vee [\![S]\!] \circ \mathbf{0}),$$

где S — тело цикла.

Очевидно, что для произвольного состояния $\nu \neq \mathbb{0}$ неравенство $f^{(1)}(\nu) \neq \mathbb{0}$ имеет место тогда и только тогда, когда p_ν определено и равно 0. Тогда $f^{(1)}(\nu) = \nu$.

Таким образом, первое приближение соответствует нулевому числу повторений тела цикла.

Для второго приближения имеем

$$f^{(2)} = {\neg_p}\bigl(ID \vee [\![S]\!] \circ {\neg_p}(ID \vee [\![S]\!] \circ \mathbf{0})\bigr).$$

Точно так же легко видеть, что $f^{(2)}(\nu) \neq \mathbb{0}$ тогда и только тогда, когда p_ν определено и равно 0 ($f^{(2)}(\nu) = \nu$) или когда $p_\nu = 1$, но $p_{[\![S]\!](\nu)} = 0$ ($f^{(2)}(\nu) = [\![S]\!](\nu)$). Таким образом, второе приближение соответствует не более чем однократному числу повторений тела цикла. Методом математической индукции несложно доказать, что i-е приближение наименьшей неподвижной точки для семантики цикла соответствует не более чем $(i-1)$-кратному повторению тела цикла, если $f^{(i)}(\nu) \neq \mathbb{0}$ для произвольного состояния $\nu \neq \mathbb{0}$.

Следовательно, индукция по неподвижной точке сводится в данном случае к индукции по числу повторений цикла, и нам для данного конкретного примера нужно доказать, что выполнение тела цикла сохраняет соотношение значений переменных y и z: для любого ν, такого, что $\nu(z) = \nu(y)!$, $[\![S]\!](\nu)(z) = [\![S]\!](\nu)(y)!$, где S — тело цикла в программе вычисления факториала.

В самом деле,

$$[\![S]\!](\nu)(z) = [\![z := z * y]\!]\big([\![y := y+1]\!](\nu)\big)(z) =$$
$$= \nu(y)! * (\nu(y)+1) = (\nu(y)+1)! = [\![S]\!](\nu)(y)!,$$

что и требовалось доказать.

Теперь остается заметить, что поскольку условие выхода из цикла имеет вид $y \neq x$, то, задав переменной n некоторой значение из $\mathbb{N}$, после выполнения нашей программы достигнем требуемого результата.

Мы рассмотрели простейший случай денотационного определения для языка, не содержащего даже блоков и процедур. При появлении последних возникают локальные определения переменных и необходимо модифицировать понятие состояния памяти. Это может быть реализовано с помощью понятия среды, которая формально определяется как декартово произведение $I \times Loc$ множества переменных на множество „адресов“, а состояние памяти при этом понимается как отображение вида $\sigma\colon I \times Loc \to D$.

Денотационное определение, как уже упоминалось, не является единственным в рамках экстенсионального подхода. Кроме него наиболее употребительными являются операциональное и трансформационное определения. При операциональном определении в качестве экстенсионала программы рассматривается множество последовательностей состояний памяти, генерируемых при выполнении программы. Трансформационное определение сопоставляет программе преобразователь вида „состояние памяти $\to$ пара (программа, состояние памяти)“. А именно если программа начинает работать над некоторым исходным состоянием памяти σ, то все операторы, которые могут быть выполнены, выполняются, а все операторы, которые не могут быть выполнены (из-за неопределенности значений переменных) „складываются“ в новую программу (называемую остаточной): таким образом возникает пара (остаточная программа, частичный результат = новое состояние σ'). Подобное

вычисление, называемое смешанным, позволяет определять различные преобразования (трансформации) программ.

Семантика языков программирования в настоящее время является бурно развивающейся областью теории формальных языков, используемой при разработке и реализации языков программирования и программных продуктов (в частности, при разработке надежного программного обеспечения).

Дополнение 8.3. Графовое представление МП-автоматов

Как и *конечные автоматы*, *МП-автоматы* также допускают графовое представление, которое, конечно, является более сложным, чем графовое представление конечных автоматов. Так, МП-автомату

$$M = (Q, V, \Gamma, q_0, F, Z_0, \delta)$$

сопоставляется ***ориентированный мультиграф*** (в отличие от обычного *ориентированного графа* в мультиграфе допускается несколько *дуг* для одной и той же *упорядоченной пары вершин*, т.е. несколько дуг, начало и конец которых совпадают), множество вершин которого есть множество *состояний* автомата, а каждая дуга которого имеет метку в виде *упорядоченной тройки* (Z, a, γ), где Z — *магазинный символ*, a — *входной символ* или *пустая цепочка*, γ — *магазинная цепочка*. По определению, дуга e, ведущая из вершины (состояния) q в вершину (состояние) r имеет метку (Z, a, γ), если и только если в *системе команд* δ находится *команда* $qaZ \to r\gamma$.

Замечание 8.16. Более формально ориентированный мультиграф может быть определен как упорядоченная тройка $G = (V, E, \varphi)$, где V — множество вершин, E — множество дуг, а φ — *отображение* множества E в множество $V \times V$, называемое ***функцией инцидентности***. Таким образом, если для различных дуг $e_1, \ldots, e_k$ все значения $\varphi(e_1), \ldots,$

$\varphi(e_k)$ равны, то возникает „множественная“, или „кратная“, дуга, т.е. множество дуг, отображаемых в одну и ту же упорядоченную пару вершин (u, v). Графически это можно изобразить так, как показано на рис. 8.40. В том случае, когда функция инцидентности есть *инъекция*, мультиграф превращается в обычный ориентированный граф.

В мультиграфе, представляющем МП-автомат, множество дуг для заданной упорядоченной пары состояний (q, r) находится во *взаимно однозначном соответствии* с множеством всех команд $qaZ \to r\gamma$ для различных $a \in V \cup \{\lambda\}$, $Z \in \Gamma$, $\gamma \in \Gamma^*$.

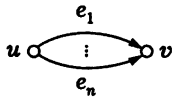


Рис. 8.40

Можно было бы задавать МП-автомат и обычным ориентированным графом, но тогда пришлось бы по-иному вводить метки дуг. Технически удобнее каждой команде сопоставлять отдельную дугу и строить таким образом ориентированный мультиграф. #

Введем понятие ***магазинной метки пути*** в мультиграфе МП-автомата. Для этого заметим, что понятие ***пути в мультиграфе*** слегка отличается от такового в обычном ориентированном графе. Под путем в мультиграфе понимается произвольная последовательность дуг e_1, e_2, ..., e_n, ..., где $n \geqslant 1$, такая, что любая пара соседних дуг e_i, e_{i+1} этой последовательности смежна, т.е. существует вершина q_{i+1}, в которую дуга e_i заходит и из которой дуга e_{i+1} выходит. Число дуг в пути (для конечного пути) назовем его длиной. Пути в мультиграфе будем записывать как цепочки в алфавите E: так, запись $e_1 e_2^m e_3^n$ обозначает путь, являющийся последовательностью дуг

$$e_1, \underbrace{e_2, \ldots, e_2}_{m \text{ раз}}, \underbrace{e_3, \ldots, e_3}_{n \text{ раз}}.$$

Тогда магазинная метка дуги (т.е. пути длины 1) с меткой (Z, a, γ) есть цепочка γ; магазинная метка пути e_1, e_2, ..., e_n длины n при условии, что магазинная метка пути e_1, e_2, ..., e_{n-1}, служащего началом исходного пути, определена и равна

γ, а дуга e_n имеет метку (Y, b, α), равна $\alpha(Y^{-1}\gamma)$. Напомним, что для произвольных алфавита W, его буквы a и цепочки x в алфавите W выражение $a^{-1}x$ означает цепочку y, такую, что $ay = x$, и не определено, если первая буква цепочки x отлична от a (см. задачу 7.23). Таким образом, магазинная метка пути в мультиграфе, представляющем МП-автомат, может быть не определена.

Введем операцию ***Y-сцепления магазинных меток*** двух смежных дуг e_1 и e_2 с метками (Z, a, γ) и (Y, b, α) соответственно, полагая $\gamma \otimes_Y \alpha = \alpha Y^{-1}\gamma$ (рис. 8.41).

Средние компоненты меток дуг МП-автомата будем называть их ***входными метками***. Входная метка пути в мультиграфе образуется стандартно из входных меток дуг, как в конечном автомате.

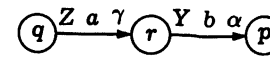


Рис. 8.41

Рассмотрим в качестве примера МП-автомат для языка $L_1 = \{a^n b^n : n \geqslant 0\}$. Его система команд имеет следующий вид:

$$q_0 aZ \to q_0 aZ,$$
$$q_0 aa \to q_0 aa,$$
$$q_0 ba \to q_1 \lambda,$$
$$q_1 ba \to q_1 \lambda,$$
$$q_1 \lambda Z \to q_2 \lambda.$$

По этой системе команд строим мультиграф (рис. 8.42).

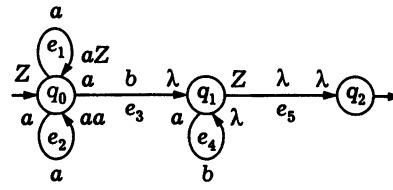


Рис. 8.42

Ясно, что дуга с меткой (Z, a, aZ) не может быть сцеплена сама с собой, так как выражение $aZ \otimes_Z aZ = aZ(Z^{-1}aZ)$ не

определено. Но верхняя петля в начальной вершине может сцепиться с нижней, а последняя сцепляется сама с собой. Например,

$$aZ \otimes_a aa = aa(a^{-1}aZ) = aaZ,$$
$$aaZ \times_a aa = aa(a^{-1}aaZ) = aaaZ.$$

Неформально любой путь $e_1e_2^{n-1}$ соответствует переписыванию с ленты в магазин n символов a. Тогда путь $e_1e_2^{n-1}e_3e_4^{n-1}e_5$ имеет входную метку a^nb^n и магазинную метку λ.

Так, для цепочки *aaabbb* будем иметь последовательность сцеплений:

$$((((((aZ \otimes_a aa) \otimes_a aa) \otimes_a \lambda) \otimes_a \lambda) \otimes_a \lambda) \otimes_Z \lambda) =$$
$$= ((((aaaZ \otimes_a \lambda) \otimes_a \lambda) \otimes_a \lambda) \otimes_Z \lambda) = Z \otimes_Z \lambda = \lambda.$$

Если в цепочке больше символов a, чем b, то мы, вычисляя магазинную метку соответствующего пути, придем к выражению $a^k \otimes_Z \lambda$, которое не определено. Если же в цепочке больше символов b, чем a, то мы не попадем в заключительную вершину, так как в нашем мультиграфе нет дуги с меткой, первая компонента которой равна Z, а вторая — b.

Обратим внимание на то, что если мы забудем о магазинных метках и будем рассматривать наш мультиграф как граф обычного конечного автомата, то получим *регулярное выражение* a^*b^+. Таким образом, вычисление магазинных меток вдоль путей мультиграфа фильтрует множество входных цепочек, допускаемых МП-автоматом, т.е. в *регулярном языке* выделяется подмножество более сложной структуры — некоторый *КС-язык*.

Любая дуга, исходящая из начальной вершины и имеющая метку (Z_0, a, γ) для произвольных $a \in V \cup \{\lambda\}$ и $\gamma \in \Gamma^*$, называется ***стартовой***. Тогда нетрудно дать графовую интерпретацию языка МП-автомата.

Теорема 8.15. Язык МП-автомата — это множество всех входных цепочек, являющихся в представляющем МП-автомат

мультиграфе входными метками путей, первая дуга которых стартовая, последняя заходит в заключительное состояние, а магазинная метка равна пустой цепочке.

Вопросы и задачи

8.1. Однозначна ли грамматика с системой правил

$$S \to aSa|bSb|aa|bb|a|b|\lambda?$$

Какой язык она порождает? Преобразовать ее к эквивалентной однозначной грамматике в приведенной форме.

8.2. Какой язык порождает грамматика с системой правил

$$S \to bSS|a?$$

8.3. Доказать, что каждый КС-язык может порождаться грамматикой $G = (V, N, S, P)$, в которой каждое правило имеет вид $A \to \alpha$ или $A \to w$, где $A \in N$, $\alpha \in N^+$, $w \in V^*$.

8.4. Построить КС-грамматику, порождающую язык

$$\{a^n b^m\colon\ n \neq m,\ n, m > 0\}.$$

8.5. Построить КС-грамматику с терминальным алфавитом V, порождающую все цепочки, содержащие вхождения некоторых слов из заданного конечного множества непустых слов в V.

8.6. Найти приведенную форму КС-грамматик:

а) $S \to a|A$, $B \to b$,
$A \to AB$, $C \to Sa|\lambda$;

б) $S \to A|B$, $C \to S|a|\lambda$,
$A \to C|D$, $D \to S|b$,
$B \to D|E$, $E \to S|c|\lambda$;

в) $S \to A|B$, $A \to aB|bS|b|\lambda$,
$B \to AB|Ba$, $C \to AS|b|\lambda$;

г) $S \to aB|bA|cC$, $A \to cBS|bA|C|b|\lambda$,
$B \to bSA|cCb|S$, $C \to Cd|aCa$;

д) $S \to ABC|aA|bB|\lambda$, $A \to B|\lambda|aB|b$,
$B \to C|bA|a$, $C \to B|bS|\lambda$;

е) $S \to AB|a|b$,
$A \to B|\lambda|aS$,
$B \to S|b|aA$.

8.7. Доказать, что любой КС-язык в однобуквенном алфавите регулярен.

8.8. Можно ли множество всех КС-грамматик с заданными алфавитами V и N описать посредством некоторой КС-грамматики? Какими будут алфавиты этой грамматики?

8.9. Построить грамматику, порождающую язык

$$\{a^{k^n}: n > 0\}$$

для фиксированной натуральной константы k. Является ли этот язык контекстно-свободным?

8.10. Доказать, что каждый линейный язык, не содержащий пустой цепочки, порождается грамматикой, все правила которой имеют вид $A \to aB$, $A \to Ba$, $A \to a$.

8.11. Доказать, что любая КС-грамматика может быть преобразована к эквивалентной КС-грамматике $G = (V, N, S, P)$, каждое правило которой имеет вид $A \to BC$ или $A \to a$, где A, B, $C \in N$, $a \in V \cup \{\lambda\}$. (КС-грамматику с такими ограничениями на правила вывода называют КС-грамматикой, заданной в нормальной форме Хомского.)

8.12. Доказать следующую лемму о разрастании для линейных языков.

Если L — линейный язык, то найдется такая константа p, что если $z \in L$ и $|z| > p$, то $z = uvwxy$, где $|uvxy| < p$, $vx \neq \lambda$ и $(\forall n > 0)$ $uv^n wx^n y \in L$.

Используя полученный результат, доказать, что язык правильных скобочных структур не является линейным.

8.13. Найти КС-грамматику, порождающую языки:

$$\text{а) } \{a^n b^n c^k \colon k, n > 0\}; \quad \text{б) } \{a^k b^n c^n \colon k, n > 0\}.$$

8.14. Доказать, что множество всех цепочек, которые могут появляться в магазине МП-автомата, регулярно.

Указание: использовать графовое представление МП-автомата (см. **Д.8.3**).

8.15. Построить МП-автомат, допускающий язык

$$\{a^n b^k \colon k \neq n\}.$$

8.16. Построить МП-автомат и КС-грамматику для языка в алфавите $\{a, b\}$, состоящего из всех таких цепочек, что в них число символов a совпадает с числом символов b.

8.17. Показать, что грамматика, заданная системой правил

$$S \to A \mid B, \quad A \to aAb \mid ab, \quad B \to a^3Bb \mid aBb^3 \mid a^3b \mid ab^3,$$

неоднозначна. Описать язык, который она порождает.

8.18. Построить КС-грамматику, порождающую все цепочки нулей и единиц с одинаковым числом тех и других.

8.19. Будет ли контекстно-свободным язык, являющийся дополнением языка предыдущей задачи?

8.20. Пусть L — КС-язык. Доказать, что:
а) $INIT(L) = \{w \colon (\exists x)\,(wx \in L)\}$ есть КС-язык;
б) $FIN(L) = \{w \colon (\exists x)\,(xw \in L)\}$ есть КС-язык;
в) $SUB(L) = \{w \colon (\exists x)(\exists y)\,(xwy \in L)\}$ есть КС-язык.

8.21. Доказать, что если L — КС-язык, а R — регулярный язык, то $L/R = \{w \colon wx \in L$ для некоторого $x \in R\}$ есть КС-язык.

8.22. Доказать, что следующие языки не являются контекстно-свободными:

а) $\{a^n b^m a^n b^m\colon n, m \geqslant 0\}$;

б) $\{a^n b^m c^n\colon m > n, m, n > 0\}$;

в) $\{0^n 10^n 10^n 1\colon n \geqslant 0\}$;

г) $\{x : x \in \{a, b, c, d\}^*$ и число вхождений всех четырех букв в x одинаково$\}$;

д) $\{w^n\colon w \in \{a, b\}^*\}$, где n — фиксированное число, не меньшее 2;

е) $\{(a^n b^n c^n)^m\colon n, m \geqslant 0\}$;

ж) $\{a^n b^n c^n\colon n \geqslant 0\}^*$.

8.23. Является ли контекстно-свободным язык, состоящий их всех цепочек в алфавите $\{a, b, c\}$, таких, что число вхождений в них всех трех символов попарно различно?

8.24. Используя конструкцию из доказательства теоремы 8.11, построить КС-грамматику для языка всех таких палиндромов в алфавите $\{0, 1\}$, которые одновременно содержатся в регулярном языке: а) $0^*1^*0^*$; б) $0^*1^+0^+1^*0^*$; в) состоящем из всех цепочек с четным числом нулей и четным числом единиц.

8.25. Пусть V и W — произвольные алфавиты, а $h\colon V^* \to W^*$ — морфизм. Доказать, что: а) если L — КС-язык в алфавите V, то $h(L)$ — КС-язык в алфавите W; б) если K — КС-язык в алфавите W, то $h^{-1}(K)$ — КС-язык в алфавите V.

Указание: для доказательства первого утверждения достаточно воспользоваться теоремой 8.9; чтобы доказать второе утверждение, нужно построить МП-автомат с буфером по аналогии с конструкцией конечного автомата с буфером (см. **Д.7.3**).

8.26 (задача о КС-языках как решениях алгебраических систем уравнений). Пусть $\mathcal{L}(V)$ — полукольцо всех языков в алфавите V. Мультипликативным термом в $\mathcal{L}(V)$ от переменных $X_1, \ldots, X_n$ называют выражение вида $\alpha_1 X_1^{k_1} \ldots \alpha_n X_n^{k_n} \alpha_{n+1}$, где $\alpha_i \in V^*$, $i = \overline{1, n+1}$, $k_j \geqslant 0$, $j = \overline{1, n}$,

и $n \geqslant 1$. Полиномом от переменных $X_1, \ldots, X_n$ называют произвольную (конечную) сумму мультипликативных термов.

Алгебраической системой уравнений в полукольце $\mathcal{L}(V)$ называют систему уравнений относительно неизвестных $X_1, \ldots, X_n$ вида

$$X_i = p_i(X_1, \ldots, X_n), \quad i = \overline{1, n},$$

где $p_i(X_1, \ldots, X_n)$ — полином от переменных $X_1, \ldots, X_n$.

Доказать, что алгебраическая система уравнений всегда имеет наименьшее решение и что язык в алфавите V является контекстно-свободным тогда и только тогда, когда он является компонентой наименьшего решения некоторой алгебраической системы уравнений.

У к а з а н и е: установите взаимно однозначное соответствие между алгебраическими системами уравнений и КС-грамматиками, отождествив нетерминалы грамматики с неизвестными системы уравнений. Далее, используя формулу наименьшей неподвижной точки, покажите, что компонента $X_i^{(s)}$ s-го приближения решения алгебраической системы есть язык, состоящий из всех таких цепочек в алфавите V, что они выводятся из нетерминала X_i, и высота дерева вывода каждой из них не превышает s.

СПИСОК РЕКОМЕНДУЕМОЙ ЛИТЕРАТУРЫ

Учебники и учебные пособия

Архангельский А.А. Канторовская теория множеств. М.: Изд-во Моск. ун-та, 1988. 112 с.

Ахо А., Ульман Дж. Теория синтаксического анализа, перевода и компиляции: Пер. с англ: В 2 т. Т. 1. М.: Мир, 1978. 612 с.

Ахо А., Хопкрофт Дж., Ульман Дж. Построение и анализ вычислительных алгоритмов: Пер. с англ. М.: Мир, 1979. 536 с.

Барти Т., Биркгоф Г. Современная прикладная алгебра: Пер. с англ. М.: Мир, 1976. 400 с.

Белоусов А.И., Мартынов Б.В., Щетинин А.Н. Лекции по дискретной математике. М.: Изд-во МГТУ им. Н.Э.Баумана, 1994. 96 с.

Богомолов А.М., Салий В.Н. Алгебраические основы теории дискретных систем. М.: Наука, 1997. 368 с.

Гладкий А.В. Формальные грамматики и языки. М.: Наука, 1973. 386 с.

Гретцер Г. Общая теория решеток: Пер. с англ. М.: Мир, 1982. 456 с.

Евстигнеев В.А. Применения теории графов в программировании. М.: Наука, 1985. 352 с.

Каргополов М.И., Мерзляков Ю.И. Основы теории групп. М.: Наука, 1972. 240 с.

Колмогоров А.Н., Драгалин А.Г. Введение в математическую логику. М.: Изд-во Моск. ун-та, 1982. 120 с.

Кон П. Универсальная алгебра: Пер. с англ. М.: Мир, 1968. 352 с.

Кристофидес Н. Теория графов. Алгоритмический подход: Пер. с англ. М.: Мир, 1978. 432 с.

Кук Д., Бейз Г. Компьютерная математика: Пер. с англ. М.: Наука, 1990. 384 с.

Куратовский К., Мостовский А. Введение в теорию множеств: Пер. с англ. М.: Мир, 1970. 416 с.

Лекции по теории графов / *В.А. Емеличев, О.И. Мельников, В.И. Сарванов, Р.И. Тышкевич* М.: Наука, 1990. 384 с.

Нефедов В.Н., Осипова В.А. Курс дискретной математики. М.: Изд-во МАИ, 1992. 264 с.

Нигматуллин Р. Сложность булевых функций. М.: Наука, 1991. 240 с.

Плоткин Б.И., Гринглаз Л.Я., Гварамия А.А. Элементы алгебраической теории автоматов. М.: Высш. шк., 1994. 191 с.

Саломаа А. Жемчужины теории формальных языков: Пер. с англ. М.: Мир, 1986.

Сикорски Р. Булевы алгебры: Пер. с англ. М.: Мир, 1969. 375 с.

Шенфилд Дж. Математическая логика: Пер. с англ. М.: Наука, 1975. 528 с.

Шиханович Ю. А. Введение в современную математику: (Начальные понятия). М.: Наука, 1965. 376 с.

Яблонский С.В. Введение в дискретную математику. 3-е изд. М.: Высш. шк., 2001. 384 с.

Задачники

Гаврилов Г.П., Сапоженко А.А. Сборник задач по дискретной математике. 2-е изд. М.: Наука, 1992. 368 с.

Лавров И.А., Максимова Л.Л. Задачи по теории множеств, математической логике и теории алгоритмов. 3-е изд. М.: Физматлит, 1995. 255 с.

Сборник задач по алгебре / Под ред. *А.И. Кострикина.* М.: Наука, 1987. 352 с.

Дополнительная литература

Акритас А. Основы компьютерной алгебры с приложениями: Пер. с англ. М.: Мир, 1994. 544 с.

Булос Дж., Джеффри Р. Вычислимость и логика: Пер. с англ. М.: Мир, 1994. 396 с.

Вирт Н. Алгоритмы и структуры данных: Пер. с англ. М.: Мир, 1989. 360 с.

Глушков В.М., Цейтлин Г.Е., Ющенко Е.Л. Методы символьной мультиобработки. Киев: Наук. думка, 1980. 252 с.

Донохоу П. О взаимодополняющих определениях // Семантика языков программирования: Сб. статей. М.: Мир, 1980. С. 222–394.

Ершов Ю.Л., Палютин Е.А. Математическая логика. М.: Наука, 1979. 320 с.

Кантор Г. Труды по теории множеств. М.: Наука, 1985. 430 с.

Касьянов В.Н. Оптимизирующие преобразования программ. М.: Наука, 1988. 336 с.

Катленд Н. Вычислимость. Введение в теорию рекурсивных функций: Пер. с англ. М.: Мир, 1983. 256 с.

Кнут Д. Искусство программирования для ЭВМ: В 3 т. Т. 1: Основные алгоритмы. М.: Наука, 1976. 736 с.

Кофман А. Введение в прикладную комбинаторику. М.: Наука, 1975. 480 с.

Кушнер Б.А. Лекции по конструктивному математическому анализу. М.: Наука, 1973. 448 с.

Мендельсон Э. Введение в математическую логику. М.: Наука, 1975. 320 с.

Матросов В.Л., Стеценко В.А. Лекции по дискретной математике. М.: МГПУ, 1997. 220 с.

Успенский В.А. Теорема Геделя о неполноте. М.: Наука, 1982. 111 с.

Фоменко А.Т. Наглядная геометрия и топология. М.: Изд-во Моск. ун-та, 1992. 432 с.

Handbook of Theoretical Computer Science. Vol. 1: Algorithms and Complexity. Vol. 2: Formal Models and Semantics. Elsevier (North Holland), 1990.

Kuich W., Salomaa A. Semirings, Automata, Languages. Springer, Berlin, 1986.

Периодическая литература

Белоусов А.И. Алгоритм обобщенной сортировки и его применение в диалоговых обучающих программах // Вестник МГТУ. Сер. Приборостроение. 1993. № 3. С. 118–126.

Белоусов А.И., Пастуховский А.В. Ориентированные гиперграфы и системы подстановок // Фундаментальная и прикладная математика. 1996. № 4. С. 1163–1186.

Гоген Дж.А., Мезегер Ж. Модели и равенство в логическом программировании // Математическая логика в программировании: Сб. статей. М.: Мир, 1991. С. 274–310.

Ершов А.П. О сущности трансляции // Программирование. 1977. № 5. С. 21–38.

Кораблин Ю.П., Налитов С.Д. Событийная семантика схем программ языка взаимодействия последовательных процессов // Программирование. 1993. № 3. С. 48–61.

Нгиян С.А. Функциональные языки программирования // Программирование. 1991. № 5. С. 77–86.

Скотт Д. Набросок математической теории вычислений // Кибернетический сборник. № 14. М.: Мир, 1977. С. 107–121.

Скотт Д. Теория решеток, типы данных и семантика // Данные в языках программирования: Сб. статей. М.: Мир, 1982. С. 25–53.

Тарасюк И.В. Понятия эквивалентностей для разработки параллельных систем с использованием сетей Петри // Программирование. 1998. № 4. С. 19–39.

Хоар Ч. Непротиворечивые взаимодополняющие теории семантики языков программирования // Семантика языков программирования: Сб. статей. М.: Мир, 1980. С. 196–221.

De Bakker J.W., Zucker J.J. Processes and denotational semantics of concurrency. Information and Control, 1982. № 54. P. 70–120.

De Boer F.S., Rutten J.J., Kok J.N., Palamidessi C. Semantic Models for concurrent logic languages // Theoretical Computer Science. 1991. № 86. P. 3–33.

Jensen K. An Introduction to the Theoretical Aspects of Coloured Petri Nets // Lecture Notes in Comput. Sci. 1994. № 803. P. 230–272.

Kuich W. Automata and languages generalized to ω-continuous semirings // Theoretical Comput. Sci. 1991. № 79. P. 137–150.

Kurka P. A Comparison of Finite and Cellular Automata // Lect. Notes in Comput. Sci. 1994. N 841. P. 484–493.

Peeva K. Equivalence, redution and minimization of finite automata over semirings // Theoretical Computer Science. 1991. N 88, P. 269–285.

Schmeck H. Algebraic Characterization of Reducible Flowcharts // J. Comput. System Sci. 1983. V. 27. P. 165–199.

Thomas W. Logical Specifications of Infinite Computations // Lecture Notes in Comput. Sci. 1994. N 803. P. 583–621.

ПРЕДМЕТНЫЙ УКАЗАТЕЛЬ

ОГЛАВЛЕНИЕ

Учебное издание

Математика в техническом университете
Выпуск XIX

Белоусов Алексей Иванович
Ткачев Сергей Борисович

ДИСКРЕТНАЯ МАТЕМАТИКА

Редактор *Е.В. Авалова*
Художник *С.С. Водчиц*
Компьютерная верстка *С.Б. Ткачева*

Оригинал-макет подготовлен
в Издательстве МГТУ им. Н.Э. Баумана
под руководством *А.Н. Канатникова*

Подписано в печать 19.04.2004. Формат 60×88/16.
Печать офсетная. Бумага офсетная.
Усл. печ. л. 46,5. Уч.-изд. л. 47,84.
Тираж 3000 экз. Заказ №10061

Издательство МГТУ им. Н.Э. Баумана.
105005, Москва, 2-я Бауманская, 5.

Отпечатано в ГУП ППП «Типография «Наука».
121099, г. Москва, Шубинский пер., 6.

ISBN 5-7038-1769-2