ОГЛАВЛЕНИЕ

ОГЛАВЛЕНИЕ

Глава 5. Формальные языки и грамматики

Глава 6. Параметрические системы программирования

В связи с бурным ростом вычислительной техники и постоянным расширением сферы применения ЭВМ в современной кибернетике возникают и интенсивно развиваются все новые теоретические направления. В частности, оформляется в науку такая, вначале главным образом эмпирическая область знаний, как программирование, в задачу которого входит разработка средств общения человека с ЭВМ. Теория программирования основывается на таких важных разделах дискретной математики, как теория алгоритмов, схематология, теория автоматов и теория формальных языков. Практические приложения ее состоят в решении прежде всего задач, связанных с разработкой систем математического обеспечения ЭВМ и проектированием структур самих машин. При решении задач автоматизации программирования и проектирования ЭВМ, в частности, возникает необходимость в применении идей и методов общей алгебры — одного из основополагающих и глубоко разработанных направлений математики. Этим обусловливается необходимость ознакомления специалистов по программированию и конструированию систем математического обеспечения ЭВМ с основными алгебраическими понятиями и результатами.

Монография представляет собой введение в теорию программирования с уклоном применения в ней понятий и методов универсальных алгебр. Первая часть посвящена основным понятиям общей алгебры. Рассматриваются интуитивная теория множеств, основные понятия теории отношений и универсальных алгебр. Значительное внимание уделено структурам, в частности структурам подалгебр универсальных алгебр, с целью установления критериев полноты — необходимых и достаточных условий, при которых произвольная система элементов порождает данную алгебру.

Понятия и результаты, изложенные в первой части, подбирались с учетом применения их во второй части книги. Так, аппарат теории множеств и отношений применяется в связи с изучением проблемы тождественных преобразований в системах алгоритмических алгебр (гл. 4, § 4.1, 4.2), классификацией формальных языков (гл. 5), разработкой методов синтаксического анализа для параметрических систем программирования (гл. 6). Свойства структуры подалгебр универсальных алгебр, рассмотренные в гл. 3, применяются при изучении проблемы полноты для модифицированных алгебр Поста (гл. 4, § 4.3, 4.4) и алгебр контекстно-свободных языков (гл. 5, § 5.5). С аппаратом многоосновных алгебр (гл. 3, § 3.6) связаны концепция алгоритмических алгебр (гл. 4, § 4.1) и методы формализации синтаксиса и семантики языков программирования (гл. 6, § 6.7). В первой части использованы результаты, изложенные в работах [1, 6, 8—10, 24, 25, 36, 49, 52, 54, 60, 61, 64, 66, 67, 70, 71, 73, 74, 77, 78, 86, 103, 108, 111—115, 118, 119, 127, 128, 133, 134, 139, 155, 156].

Вторая часть посвящена элементам теории программирования. В гл. 4 рассматривается аппарат алгоритмических алгебр, предложенный В. М. Глушковым [26] для решения проблем прикладной теории алгоритмов. Особое внимание уделено задачам, возникающим при автоматизации проектирования ЭВМ и программирования. Установлена связь аппарата алгоритмических алгебр с концепцией структурного программирования [41], которое представляет собой комплекс технологических средств, ориентированных на конструирование больших алгоритмов и программ. Показано, что алгоритмические алгебры можно рассматривать в качестве теории схем структурированных алгоритмов и программ и применять при разработке схемного и программного математического обеспечения для современных вычислительных систем, в частности при решении проблемы формализации семантики языков программирования (гл. 6, § 6.7). В гл. 4 использованы результаты, изложенные в работах [8, 18, 23—27, 31—37, 41, 43—50, 53—63, 65, 69, 72, 75—78, 80—82, 84, 91, 92, 113—117, 120—122, 128, 131, 133, 134, 136, 139, 145, 146, 148, 155, 156, 158].

Пятая глава является введением в теорию формальных языков. Рассматриваются вопросы классификации порождающих грамматик, формы их представлений, алгебраические свойства операций над языками. Изложенные результаты применяются в гл. 6 при решении проблем анализа и синтеза автоматных моделей языков (§ 6.1, 6.2), создании формализмов, ориентированных на эффективное решение задач конструирования многих важных системных процессов (§ 6.3—6.6), развитии алгебраических методов описания синтаксиса и семантики языков программирования (§ 6.7). В гл. 5 использованы результаты, изложенные в работах [3, 7, 17—25, 36, 37, 48, 70, 80, 85, 94, 95, 99, 100, 109, 110, 118, 128, 137, 138, 146].

Методологии построения современных языков и систем программирования посвящена гл. 6. В настоящем издании эта глава существенно переработана. В ней отражены некоторые новые результаты. Для этого, в связи с ограниченностью объема книги, пришлось опустить часть материала, изложенного в первом издании. Так, в § 6.1 рассмотрена концепция автоматов над внутренней памятью [37], обобщающая известные автоматные структуры (магазинные и БС-автоматы, счетчиковые машины и др.), применяемые в системном программировании (в первом издании этот параграф посвящен историческому очерку развития идей программирования в СССР и за рубежом). В связи с введением автоматов над внутренней памятью в определенной мере изменены и последующие параграфы главы, в частности рассмотрены стратегия двустороннего синтаксического анализа и ее обобщение в связи с конструированием параллельных систем программирования для многопроцессорных вычислительных комплексов, методы формализации синтаксиса и семантики языков программирования с привлечением аппарата многоосновных алгебр. Методы конструирования параметрических систем программирования применялись при разработке программного обеспечения некоторых современных вычислительных систем как специализированного, так и универсального назначения. Дальнейшее развитие этих методов будет способствовать созданию высокопроизводительных комплексов ЭВМ последующих поколений. В гл. 6 использованы результаты, изложенные в работах [2—5,11—16, 18—21, 28—30, 36—42, 48, 51, 52, 62, 68, 70, 71, 79, 83, 85, 87—90, 93, 95—98, 101—107, 123—126, 128—130, 132, 135, 140—147, 149—154, 157, 159, 160].

В настоящее издание монографии внесены поправки, вызванные допущенными в первом издании неточностями, опечатками или недосмотрами.

Авторы

§ 1.1. Понятие множества. Подмножества. Включения.

Потребности интенсивно развивающегося анализа, особенно теории функций действительного переменного, обусловили в конце XIX в. создание теории множеств — фундаментального раздела современной математики.

В окружающей нас реальной действительности существуют как отдельные объекты, так и их совокупности, или множества. Например, можно говорить об отдельной почтовой марке и об их коллекции, о произведении Л. Н. Толстого «Война и мир» и о полном собрании сочинений этого писателя, о числе 5 и о совокупности всех натуральных чисел, о некоторой ячейке памяти и о массиве ячеек запоминающего устройства ЭВМ, о программе, написанной в некотором алгоритмическом языке, и о самом языке как множестве таких программ.

Основоположник теории множеств известный немецкий математик Г. Кантор понимал множество как объединение в одно целое объектов, хорошо различаемых нашей интуицией или мыслью. Термины «множество», «совокупность», «коллекция», «собрание», «массив», «объединение» вполне равноправны. Далее в качестве основного используется лишь один из них — множество, причем в отличие от аксиоматических теоретико-множественных построений (см., например, [9]) рассматривается интуитивная теория множеств. Понятию множества нельзя дать строгого математического определения, так как оно настолько широко и общо́, что не входит как часть ни в какое другое более общее понятие. Поэтому множество относится к неопределяемым понятиям науки.

Элементы множества могут быть самого разнообразного вида, любой природы. Однако нас далее будут интересовать множества, образованные из абстрактных математических объектов, таких, как числа, функции, векторы, цепочки символов (слова) в некотором алфавите. Объекты или элементы, составляющие множество,

будем обозначать строчными латинскими буквами: $a, b, c, \ldots, x, y, \ldots$, а множества — прописными: $A, B, C, \ldots, M, N, \ldots$ Как обычно, обозначения элементов или множеств могут быть с индексами или без них. Следует различать общий элемент x множества A, т. е. произвольный элемент, принадлежащий данному множеству, и конкретные элементы $a, b, c, \ldots$, каждый из которых отличен от остальных. Символ $\in$ обозначает принадлежность элемента x множеству A. Так, выражение $x \in A$ читается «элемент x принадлежит множеству A». Если x не принадлежит A, будем писать $x \notin A$. Элементы, принадлежащие некоторому множеству A, условимся заключать в фигурные скобки: $A = \{a, b, c, \ldots\}$ означает, что $a, b, c, \ldots$ — суть элементы множества A. Множество, состоящее из конечного числа элементов, называется **конечным**; в противном случае говорят о **бесконечном** множестве. Например, конечны множество всех кресел в зрительной зале кинотеатра, множество всех абонентов городской телефонной станции, множество всех четных положительных чисел, меньших 9 ($A = \{2, 4, 6, 8\}$), множество всевозможных пар из элементов a и b ($B = \{(a, a), (b,b), (a, b), (b, a)\}$). Примерами бесконечных множеств являются множество всех натуральных чисел ($N = \{1, 2, 3, \ldots\}$), множество всех четных натуральных чисел ($N_0 = \{2, 4, 6, \ldots\}$), множество точек на окружности, множество слов произвольной длины в алфавите из двух символов: 0, 1 (под длиной слова в некотором алфавите понимается число символов из этого алфавита, входящих в данное слово).

Каковы же способы задания множеств?

Приведенные примеры показывают, что множество может быть задано перечислением всех его элементов. Этот способ пригоден для задания конечных множеств, состоящих из сравнительно небольшого числа легко различимых элементов. Например, множество всех книг в библиотеке задается библиотечным каталогом, множество номеров телефонных абонентов — телефонным справочником, множество ячеек оперативной памяти ЭВМ — совокупностью адресов этих ячеек. Однако невозможно перечислить множество всех птиц в лесу или множество всех песчинок на пляже. Гораздо чаще множества задаются посредством указания характеристического свойства — признака, которому удовлетворяют элементы этого множества и только они. Это наиболее общий способ задания множеств. Таким способом задаются бесконечные множества: множество натуральных чисел, множество всех действительных чисел в интервале $[0, 1]$, множество всевозможных последовательностей конечной длины, состоящих из букв некоторого алфавита, множество всех программ, которые можно записать на определенном языке программирования, и др.

Множества могут быть заданы также с помощью элементов уже известных множеств в результате применения к ним некоторого правила (алгоритма) или с помощью некоторых операций над известными множествами. Например, множество целых чисел, заданных в десятичной системе счисления, сумма цифр которых делится на 5, множество всех прямых вида $y = x + c$, где c — произвольный элемент из множества всех действительных чисел, множество всех слов произвольной длины, состоящих из элементов (букв) множества (алфавита) $\mathfrak{A} = \{a_1, a_2, \ldots, a_k\}$, множество $C = \{2, 4, 6\}$ всех элементов, принадлежащих одновременно множеству $A = \{1, 2, \ldots, 7\}$ и множеству $B = \{2, 4, 6, 8, 10\}$. При этом описания вспомогательных множеств всегда должны содержаться в описании задаваемого множества. Как правило, используется следующая форма записи: $N_0 = \{2n \mid n = 1, 2, \ldots\}$ — множество всех четных натуральных чисел, $N_1 = \{2n - 1 \mid n = 1, 2, \ldots\}$ — множество всех нечетных натуральных чисел, $p = \left\{(x, y) \,\middle|\, \begin{matrix} x \in \{a_1, \ldots, a_k\} \\ y \in \{b_1, \ldots, b_k\} \end{matrix}\right\}$ — множество всех пар, первая компонента которых принадлежит множеству $A = \{a_1, \ldots, a_k\}$, а вторая — множеству $B = \{b_1, \ldots, b_k\}$. В приведенных примерах описания вспомогательных множеств содержатся после черты.

В математических рассуждениях часто приходится рассматривать так называемые **одноэлементные** множества, т. е. множества, состоящие лишь из одного элемента (например, $\{a\}$), а также **пустое** множество, не содержащее элементов вообще, $\varnothing$. Необходимость в этих понятиях вызвана тем, что, говоря о множестве, мы иногда заранее не знаем, содержит ли оно какие-нибудь элементы, содержит ли оно один или более чем один элемент. Например, неизвестно, существуют ли целые не равные нулю числа x, y, z такие, что $x^n + y^n = z^n$ при $n > 2$, или множество таких чисел пусто (великая теорема Ферма). В то же время в евклидовой геометрии пусто множество треугольников, сумма углов которых не равна 180°, а множество действительных корней уравнения $x^3 - x^2 + x - 1 = 0$ одноэлементно.

Множества A и B называются **равными**, $A = B$, если они состоят из одних и тех же элементов. Если все элементы множества A являются также элементами множества B, говорят, что A — **подмножество** B, или A **включается** в B. Включение множества A в множество B обозначается следующим образом: $\subset$, $\subseteq$, $\supset$, $\supseteq$. Так, выражение $A \subseteq B$ читается «множество A включается в множество B». Такой же смысл имеет запись $B \supseteq A$. Если $A \subseteq B$, причем $A \neq B$, то употребляется запись $A \subset B$. Знак $\subset$ ($\supset$), в отличие от знака $\subseteq$ ($\supseteq$), называется знаком **строгого включения**. Очевидно, для любого множества A выполняются включения $\varnothing \subseteq A$, $A \subseteq A$ (**рефлексивность**).

Множества A и $\varnothing$ называются **несобственными подмножествами** множества A, а все остальные подмножества A — его **собственными подмножествами**. Например, множество пешек

является собственным подмножеством множества всех шахматных фигур, множество натуральных чисел — подмножеством множества всех целых чисел, а последнее, в свою очередь, — подмножеством множества всех рациональных чисел.

Если $A \subseteq B$ и $B \subseteq A$, то $A = B$ **(антисимметричность)**. Кроме того, если $A \subseteq B$ и $B \subseteq C$, то $A \subseteq C$ **(транзитивность)**.

Для удобства изложения далее используются следующие обозначения: $q_1, q_2, \ldots, q_n \in A$ — каждый элемент q_i принадлежит множеству A; $A_1, A_2, \ldots, A_n \subseteq A$ — каждое множество A_i является подмножеством множества A $(i = 1, 2, \ldots, n)$.

Обычно множества, встречающиеся в рассуждениях той или иной математической теории, являются подмножествами некоторого фиксированного множества E, называемого **универсальным** для данной теории. Как правило, множество E представляется здесь в неявной форме. Например, в некоторых разделах элементарной математики универсальными являются множество всех точек пространства (в геометрии), множество всех неотрицательных чисел (в арифметике), множество всех комплексных чисел и алгебраических функций (в алгебре). Каждому подмножеству $A \subseteq E$ можно сопоставить **характеристическую функцию** φ_A такую, что $\varphi_A(x) = 0$ для всякого $x \in A$ и $\varphi_A(x) = 1$, если $x \overline{\in} A$. Характеристической функции соответствует определенное свойство элементов, принадлежащих множеству A, которое также называется **характеристическим**. Если характеристические свойства выделяют в универсальном множестве E одно и то же подмножество, то они называются эквивалентными относительно E. Выберем, например, в качестве универсального множество всех треугольников евклидовой плоскости. Тогда характеристическое свойство «быть треугольником, у которого углы при основании равны», определяет подмножество всех равнобедренных треугольников. Эквивалентным данному свойству является свойство «быть треугольником, у которого совпадают высота и медиана, проведенные из одной вершины». В то же время свойство «быть равносторонним треугольником с углом, отличным от 60°», очевидно, определяет множество $\varnothing$.

В качестве элементов множества могут использоваться другие множества, тогда говорят о совокупности **(семействе)** множеств $\{A_\alpha \mid \alpha \in I\}$, где A_α — некоторые множества, а I — множество индексов, которое может быть конечным или бесконечным. Так, можно рассматривать множество $\{A_n \mid n \in N\}$, где A_n — множество всех слов в алфавите $\mathfrak{A}$ длины, не превышающей n, или множество $\{P_n \mid n \in N\}$, где P_n — множество всех многочленов $a_0 x^k + a_1 x^{k-1} + \ldots + a_{k-1} x + a_k$ с рациональными коэффициентами a_i $(i = 0, 1, 2, \ldots, k)$ при $k \leqslant n$. В обоих случаях множеством индексов является множество N всех натуральных чисел $(I = N)$.

Рассмотрим множество $\mathfrak{M}(A)$ всех подмножеств некоторого множества A, включая и его несобственные подмножества. Например, если $A = \{a, b, c\}$, то $\mathfrak{M}(A) = \{\{\varnothing\}, \{a\}, \{b\}, \{c\}, \{a, b\}, \{a, c\}, \{b, c\}, \{a, b, c\}\}$. Пусть $B = \{a_1, a_2, \ldots, a_n\}$ — некоторое конечное множество. Каждому подмножеству $B' \subseteq B$ сопоставим двоичный набор $\tilde{\alpha}(B') = (\alpha_1, \alpha_2, \ldots, \alpha_n)$ такой, что

$$\alpha_i = \begin{cases} 0, & \text{если } a_i \in B', \\ 1, & \text{если } a_i \overline{\in} B' \end{cases} \quad (i = 1, 2, \ldots, n).$$

Так, несобственным подмножествам множества B соответствуют наборы $\tilde{\alpha}(\varnothing) = (1, 1, \ldots, 1)$ и $\tilde{\alpha}(B) = (0, 0, \ldots, 0)$, а подмножеству $B' = \{a_2, a_3, a_5\}$ — набор $\tilde{\alpha}(B) = (1, 0, 0, 1, 0, 1, \ldots, 1)$. Таким образом, для того чтобы подсчитать число элементов, входящих в множество $\mathfrak{M}(B)$, достаточно выяснить число $l(n)$ двоичных n-мерных наборов. Рассуждая по индукции, замечаем, что $l(1) = 2^1 = 2$. Далее, допустим, что $l(n-1) = 2^{n-1}$. Но каждому $(n-1)$-мерному набору $(\alpha_1, \alpha_2, \ldots, \alpha_{n-1})$ соответствуют n-мерные двоичные наборы $(\alpha_1, \alpha_2, \ldots, \alpha_{n-1}, 0)$ и $(\alpha_1, \alpha_2, \ldots, \alpha_{n-1}, 1)$. Следовательно, число элементов $l(n)$, входящих в множество $\mathfrak{M}(B)$, равно $l(n-1) \cdot 2 = 2^{n-1} \cdot 2 = 2^n$.

§ 1.2. Основные операции над множествами

Как отмечалось, множества могут порождаться в результате применения некоторых операций к уже известным множествам. Рассмотрим так называемые **булевы операции** над множествами: объединение, пересечение и дополнение.

Объединением множеств A и B $(A \cup B)$ называется множество всех элементов, каждый из которых принадлежит хотя бы одному из множеств A или B. При этом элементы, принадлежащие как множеству A, так и множеству B, входят в объединение $A \cup B$ только один раз. Например, $\{a, b, c, d, e\} = \{a, b, c\} \cup \{b, c, d, e\}$; $N = N_0 \cup N_1$ (N_0, N_1 — множество четных и нечетных натуральных чисел соответственно).

Пересечением множеств A и B $(A \cap B)$ называется множество, состоящее из всех элементов, одновременно принадлежащих как множеству A, так и множеству B. Например, $\{b, c\} = \{a, b, c\} \cap \{b, c, d, e\}$; $K = P \cap R$ (K, P, R — множества квадратов, прямоугольников и ромбов соответственно).

Множества A и B называются **непересекающимися**, если $A \cap B = \varnothing$; в противном случае множества A и B пересекаются. Не пересекаются, например, множества всех четных и нечетных чисел, множества всех прямоугольных и равносторонних треугольников в евклидовом пространстве.

Операции объединения и пересечения множеств могут быть распространены на случай совокупности множеств $\{A_\alpha \mid \alpha \in I\}$. Тогда объединение $\bigcup\limits_{\alpha \in I} A_\alpha$ состоит из всех элементов, которые принадлежат хотя бы одному из множеств A_α, а пересечение $\bigcap\limits_{\alpha \in I} A_\alpha$ — из всех элементов, одновременно принадлежащих каждому из множеств A_α.

Дополнением множества $A \subseteq E$ к универсальному множеству E называется множество $\bar{A}$ всех элементов $x \in E$ таких, что $x \bar{\in} A$. Дополнение часто также обозначают[1] $C_E(A)$ или $C(A)$. Например, $N_1 = C_N(N_0)$ (N, N_0, N_1 — определенные выше множества натуральных чисел), $T_r^0 = C_{T_r}(T_r^1)$ (T_r, T_r^0, T_r^1 — множества соответственно всех равнобедренных треугольников, всех равносторонних треугольников, всех равнобедренных треугольников, у которых углы при основании отличны от угла при вершине).

Пусть φ_A, φ_B — характеристические свойства множеств A и B соответственно. Тогда $\varphi_{A \cup B} = \varphi_A \vee \varphi_B$, $\varphi_{A \cap B} = \varphi_A \wedge \varphi_B$, $\varphi_{\bar{A}} = \neg \varphi_A$ — характеристические свойства соответственно объединения $A \cup B$, пересечения $A \cap B$, дополнения $\bar{A}$, где $\varphi_A \vee \varphi_B$ — **логическое сложение (дизъюнкция)** свойств φ_A, φ_B (т. е. характеристическое свойство истинно тогда и только тогда, когда хотя бы одно из свойств φ_A, φ_B истинно), $\varphi_A \wedge \varphi_B$ — **логическое произведение (конъюнкция)** свойств φ_A, φ_B (т. е. характеристическое свойство истинно тогда и только тогда, когда оба свойства φ_A, φ_B истинны), $\neg \varphi_A$ — **отрицание** свойства φ_A (т. е. характеристическое свойство истинно, когда φ_A ложно, и ложно, когда φ_A истинно)[2].

Для иллюстрации булевых операций над множествами часто используются диаграммы Венна (рис. 1, квадрат означает универсальное множество E, а заштрихованные области — результат применения операций к множествам A и B). Назовем наиболее важные свойства булевых операций $\cup$, $\cap$, $-$:

коммутативность —

$$\begin{cases} A \cup B = B \cup A, \\ A \cap B = B \cap A; \end{cases}$$

ассоциативность —

$$\begin{cases} (A \cup B) \cup C = A \cup (B \cup C), \\ (A \cap B) \cap C = A \cap (B \cap C) \end{cases}$$

[1] C — начальная буква французского слова complement — дополнение.

[2] Операции над высказываниями $\vee$, $\wedge$, $\neg$ впервые были рассмотрены известным английским математиком Дж. Булем. Более подробно о логических операциях алгебры высказываний см. в работе [86].

(свойство ассоциативности позволяет использовать бесскобочную запись $A_1 \cup A_2 \cup \ldots \cup A_n$, $A_1 \cap A_2 \cap \ldots \cap A_n$);

дистрибутивность пересечения относительно объединения —

$$A \cap (B \cup C) = (A \cap B) \cup (A \cap C);$$

дистрибутивность объединения относительно пересечения —

$$A \cup (B \cap C) = (A \cup B) \cap (A \cup C);$$

идемпотентность —

$$\begin{cases} A \cup A = A, \\ A \cap A = A \end{cases}$$

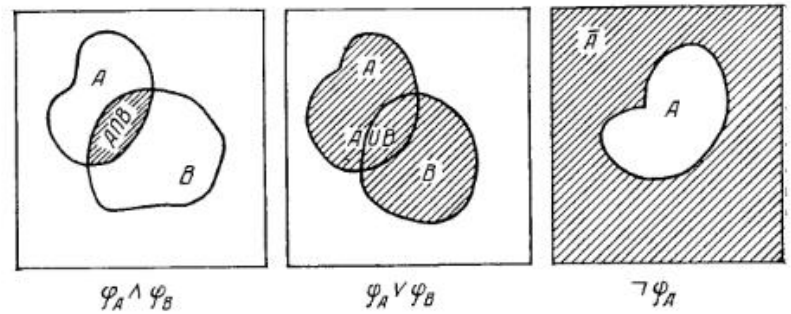


Рис. 1.

(свойство идемпотентности позволяет записывать формулы, содержащие знаки $\cup$, $\cap$, без коэффициентов и показателей);

правила поглощения —

$$\begin{cases} A \cup (A \cap B) = A, \\ A \cap (A \cup B) = A; \end{cases}$$

инволюция —

$$\bar{\bar{A}} = A;$$

правила де Моргана

$$\begin{cases} \overline{A \cup B} = \bar{A} \cap \bar{B}, \\ \overline{A \cap B} = \bar{A} \cup \bar{B}; \end{cases}$$

обобщение правил де Моргана для совокупности множеств —

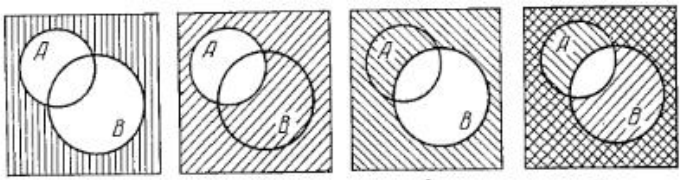
$$\begin{cases} \overline{\bigcup\limits_{\alpha \in I} A_\alpha} = \bigcap\limits_{\alpha \in I} \bar{A}_\alpha, \\ \overline{\bigcap\limits_{\alpha \in I} A_\alpha} = \bigcup\limits_{\alpha \in I} \bar{A}_\alpha; \end{cases}$$

свойства универсального и пустого множеств —

$$\begin{cases} A \cup \overline{A} = E, \\ A \cap \overline{A} = \varnothing, \end{cases}$$

$$\begin{cases} \overline{E} = \varnothing, \ A \cup \varnothing = A, \ A \cup E = E, \\ \overline{\varnothing} = E, \ A \cap E = A, \ A \cap \varnothing = \varnothing. \end{cases}$$

Заключенные в скобки пары свойств двойственны друг другу в том смысле, что если в одном из них заменить $\cup$ знаком $\cap$, E — знаком $\varnothing$ и, наоборот, $\cap$ — знаком $\cup$, а $\varnothing$ — знаком E, то получается парное свойство.



а б в г

Рис. 2.

С целью упрощения записи формул скобки могут опускаться. Тогда предполагается, что их роль играет знак дополнения и порядок выполнения операций следующий: дополнение, пересечение, объединение. Согласно этому условию формулу $\overline{((\overline{A}) \cup (A \cap B))} \cup (A \cap B)$ можно кратко записать так: $\overline{\overline{A} \cup A \cap B} \cup A \cap B$.

В справедливости перечисленных свойств нетрудно убедиться. В качестве примера докажем первое правило де Моргана

$$\overline{A \cup B} = \overline{A} \cap \overline{B}. \tag{1.1}$$

Покажем вначале, что левая часть этого равенства включается в правую, т. е.

$$\overline{A \cup B} \subseteq \overline{A} \cap \overline{B}. \tag{1.2}$$

Действительно, пусть $x \in \overline{A \cup B}$. Это означает, что $x \notin A$ и $x \notin B$, но тогда $x \in \overline{A}$ и $x \in \overline{B}$, следовательно, $x \in \overline{A} \cap \overline{B}$. Таким образом, выполняется включение (1.2).

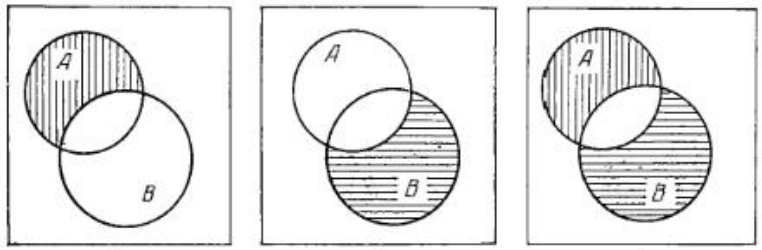
Справедливо также и обратное включение

$$\overline{A} \cap \overline{B} \subseteq \overline{A \cup B}. \tag{1.3}$$

В самом деле, пусть $x \in \overline{A} \cap \overline{B}$, это означает, что $x \notin A$ и $x \notin B$.

Но тогда $x \notin A \cup B$ и, следовательно, $x \in \overline{A \cup B}$. Ввиду справедливости включений (1.2), (1.3) выполняется равенство (1.1). Проведенное доказательство интерпретируется на диаграммах Венна (рис. 2). Заштрихованная область рис. 2, *а* совпадает с областью, отмеченной двойной штриховкой на рис. 2, *г*.

К основным теоретико-множественным операциям относится разность множеств. **Разностью множеств** A и B $(A \setminus B)$ называется множество всех элементов из A, не принадлежащих множеству B. При этом не обязательно выполнение включения $B \subseteq A$, так что $A \setminus B = A \setminus (A \cap B)$. Например, $\{a, b, c\} \setminus \{b, c, d\} = \{a\}$. Пусть P, R, K — множества прямоугольников, ромбов и квадратов соответственно. Тогда разность $P \setminus R$ — множество прямоугольников, не являющихся квадратами, т. е. $P \setminus R = P \setminus K$, где $K = P \cap R$. Разность множеств иллюстрируется на рис. 3, *а*. Заметим, что разность множеств можно выразить с помощью введенных ранее булевых операций $A \setminus B = A \cap \overline{B}$. С операцией разности тесно связана операция симметрической разности множеств. **Симметрической разностью** множеств A и B $(A \ominus B)$ называется множество, содержащее все элементы из A, не принадлежащие множеству B, а также все элементы из B, не принадлежащие множеству A. Например, $\{a, b, c\} \ominus \{b, c, d\} = \{a, d\}$; $P \ominus R$ — множество всех прямоугольников и ромбов, не являющихся квадратами (P, R — соответственно множества прямоугольников и ромбов). Симметрическая разность иллюстрируется на рис. 3, *в*. По определению $A \ominus B = (A \setminus B) \cup (B \setminus A) = (A \cap \overline{B}) \cup (B \cap \overline{A})$. Назовем некоторые свойства симметрической разности:



а б в

Рис. 3.

коммутативность —

$$A \ominus B = B \ominus A,$$

ассоциативность —

$$A \ominus (B \ominus C) = (A \ominus B) \ominus C,$$

дистрибутивность относительно пересечения —

$$A \cap (B \oplus C) = (A \cap B) \oplus (A \cap C),$$

свойства универсального и пустого множеств —

$$A \oplus A = \varnothing,\ A \oplus E = \overline{A},\ A \oplus \varnothing = A.$$

При классификации элементов и в ряде других случаев используется **разбиение** $R(A)$ множества A на непересекающиеся подмножества — классы этого разбиения; иными словами, множества $R(A) = \{A_1, A_2, \ldots, A_k\}$ образуют разбиение множества A, если $A = \bigcup_{i=1}^{k} A_i$, причем $A_i \cap A_j = \varnothing$, где $i \neq j$ $(i, j = 1, 2, \ldots, k)$. В этом случае множество A часто называют **прямой суммой** подмножеств $A_1, A_2, \ldots, A_k$. Например, множество Z всех целых чисел является прямой суммой множеств Z_0 — всех четных чисел и Z_1 — всех нечетных чисел, а множество $\mathfrak{M}(B)$ всех подмножеств множества $B = \{a_1, a_2, \ldots, a_n\}$ — прямой суммой множеств $\mathfrak{M}_0, \mathfrak{M}_1, \ldots, \mathfrak{M}_n$, где $\mathfrak{M}_0 = \{\varnothing\}$, $\mathfrak{M}_i$ — множество всех подмножеств множества B, каждое из которых содержит точно i элементов $(i = 1, 2, \ldots, n)$.

Разбиение $R_1(A) = \{A_1, A_2, \ldots, A_h\}$ называется **подразбиением** разбиения $R_2(A)$, если для каждого класса $B_i \in R_2(A)$ существует разбиение $R(B_i) = \{A_{i_1}, A_{i_2}, \ldots, A_{i_s}\} \subseteq R_1(A)$. Например, подразбиением любого разбиения множества $A = \{a_1, a_2, \ldots, a_n\}$ является тривиальное разбиение $R(A) = \{\{a_1\}, \{a_2\}, \ldots, \{a_n\}\}$.

Рассмотрим также операцию декартова (прямого) произведения множеств. **Декартовым произведением** $A \times B$ множеств A и B называется множество всех пар вида (a_i, b_j), в которых первая компонента принадлежит множеству A $(a_i \in A)$, а вторая — множеству B $(b_j \in B)$. Аналогично декартовым произведением $A_1 \times A_2 \times \cdots \times A_n$ множеств $A_1, A_2, \ldots, A_n$ называется множество всех n-мерных последовательностей $(a_{i_1}, a_{i_2}, \ldots, a_{i_n})$, в которых k-я компонента принадлежит множеству A_k $(a_{i_k} \in A_k)$ при любом $k = 1, 2, \ldots, n$. В частности, если $A_1 = A_2 = \cdots = A_n = A$, то декартово произведение $\underbrace{A \times A \times \cdots \times A}_{n \text{ раз}}$ называют n-й **степенью** множества A. Причем при $n = 0$ принимается $A^0 = \varnothing$, а при $n = 1$ $A^1 = A$.

С помощью понятия степени множества определим операцию $A^* = \bigcup_{k=0}^{\infty} A^k$, которая порождает множество всех последовательностей (произвольной длины), состоящих из элементов множества A. Декартово произведение, вообще говоря, некоммутативно $(A \times B \neq B \times A)$ и ассоциативно $((A \times B) \times C = A \times (B \times C))$. Очевидно, $(A_1 \times A_2 \times \cdots \times A_n) = \varnothing$ тогда и только тогда, когда по крайней мере одно из множеств A_i пусто, $A_i = \varnothing$ $(i \in \{1, 2, \ldots$

$\ldots, n\}$). Пусть для множеств $A_1, A_2, \ldots, A_n$ и $B_1, B_2, \ldots, B_n$ выполняются включения $B_i \subseteq A_i$ $(i = 1, 2, \ldots, n)$. Тогда $(B_1 \times B_2 \times \cdots \times B_n) \subseteq A_1 \times A_2 \times \cdots \times A_n$. Декартово произведение множеств связано с числовым умножением. Пусть $A_1, A_2, \ldots, A_n$ — конечные множества, причем множество A_i содержит m_i элементов $(i = 1, 2, \ldots, n)$. Тогда декартово произведение $A_1 \times A_2 \times \cdots \times A_n$ содержит $m_1 m_2 \ldots m_n$ элементов. Приведем примеры декартовых произведений различных множеств.

Пример 1. Пусть даны множества $A = \{a, b\}$ и $B = \{b, c\}$. Тогда $A \times B = \{a, b\} \times \{b, c\} = \{(a, b), (a, c), (b, b), (b, c)\}$.

Пример 2. Пусть V — множество всех действительных чисел. Тогда $V^2 = V \times V$ представляет собой множество всех пар (x, y),

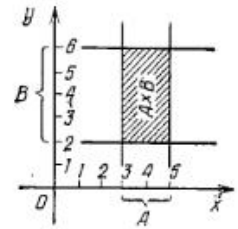


Рис. 4.

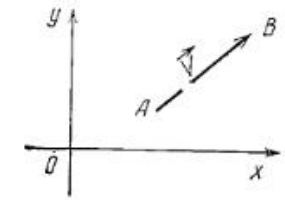


Рис. 5.

где $x, y \in V$. Каждой такой паре $(x, y) \in V^2$ можно сопоставить точку координатной плоскости, абсцисса которой равна первой компоненте пары, а ордината — второй. Если теперь в качестве множества A выбрать все точки x оси абсцисс такие, что $3 \leqslant x \leqslant 5$, а в качестве множества B — все точки y оси ординат такие, что $2 \leqslant y \leqslant 6$, то декартову произведению $A \times B$ соответствует множество точек плоскости с координатами из множеств A и B (рис. 4, заштрихованный прямоугольник).

Пример 3. Рассмотрим множество $\vec{V}$ направленных отрезков — векторов координатной плоскости. Для задания некоторого вектора $\overrightarrow{AB}$ достаточно указать координаты его начальной точки A, длину вектора и направление $s \in S$ (рис. 5), S — множество всех направленных лучей, которые могут исходить из произвольной точки плоскости. Тогда множество $\vec{V}$ векторов плоскости можно представить как декартово произведение $\vec{V} = V^2 \times V \times S$, где V^2, V и S — множества соответственно координат всех точек плоскости, длин всех отрезков плоскости и всех направленных лучей из произвольной точки.

Пример 4. Декартова степень $E_2^n = \underbrace{E_2 \times E_2 \times \cdots \times E_2}_{n \text{ раз}}$, где

$E_2 = \{0, 1\}$, представляет собой множество всех n-мерных двоичных наборов, состоящих из нулей и единиц.

Пример 5. Операция $\mathfrak{A}^*$ порождает множество всех наборов любой конечной длины в алфавите $\mathfrak{A}$, причем здесь $\varnothing$ — пустой набор.

§ 1.3. Мощность множеств. Счетные множества

Изученные в предыдущем параграфе операции и их свойства справедливы как для конечных, так и для бесконечных множеств. Принципиальное различие между конечными и бесконечными множествами проявляется при сравнении их по числу входящих элементов. Пусть, например, даны два конечных множества: A с числом элементов n и B с числом элементов m. Тогда выполняется лишь одно из трех возможных соотношений $n = m$, $n > m$, $n < m$. Если множества A и B состоят из сравнительно небольшого числа элементов, то вопрос о том, какое из этих соотношений справедливо, может быть решен в результате прямого подсчета элементов в сравниваемых множествах. Однако такой подсчет даже конечных множеств не всегда осуществим. Например, как подсчитать число элементов в множестве песчинок на городском пляже или число атомов на земном шаре?

Существует и другой способ сравнения множеств. Так, для того чтобы выяснить, какое из следующих множеств содержит больше элементов: множество кресел в зрительном зале или множество всех желающих посмотреть фильм, достаточно усадить зрителей на соответствующие места. Результат сравнения этих множеств зависит от того, будут ли в зале свободные места, или, наоборот, у кинотеатра окажутся «охотники» за лишними билетами. Точно так же можно сказать, что песчинок на пляже меньше, чем атомов на земном шаре, так как каждая песчинка содержит по крайней мере один атом.

Сравнение множеств в приведенных примерах связано с установлением взаимно-однозначного соответствия. Элементы множеств A и B находятся во **взаимно-однозначном соответствии**, если каждому элементу a множества A по некоторому закону сопоставлен единственный элемент b множества B, причем каждый элемент $b \in B$ оказывается сопоставленным одному и только одному элементу $a \in A$. Взаимно-однозначное соответствие элементов множеств A, B обозначается $a \leftrightarrow b$.

Множества A и B **называются эквивалентными** или **равномощными** ($A \sim B$), если можно установить взаимно-однозначное соответствие их элементов.

Из определения эквивалентности множеств следует, что:

1) $A \sim A$ **(рефлексивность)**,

2) если $A \sim B$, то $B \sim A$ **(симметричность)**,

3) если $A \sim B$ и $B \sim C$, то $A \sim C$ **(транзитивность)**.

Приведенное определение эквивалентности справедливо как для конечных, так и для бесконечных множеств. Очевидно, конечные множества A и B эквивалентны лишь тогда, когда они содержат одинаковое число элементов; если $A \sim B'$, где $B' \subset B$, то множество A содержит меньше элементов, чем множество B. Таким образом, понятие мощности является обобщением понятия числа элементов. Приведем примеры эквивалентности бесконечных множеств.

Пример 1. Множество $N = \{1, 2, 3, \ldots\}$ натуральных чисел эквивалентно множеству $N^2 = \{1, 4, 9, \ldots\}$ квадратов этих чисел. Взаимно-однозначное соответствие множеств N и N^2 устанавливается по следующему закону: каждому числу $n \in N$ сопоставляется число $n^2 \in N^2$ $(n \leftrightarrow n^2)$.

Пример 2. Множество Z всех целых чисел эквивалентно множеству Z_0 всех четных чисел; взаимно-однозначное соответствие устанавливается следующим образом: $z \leftrightarrow 2z$, где $z \in Z$, $2z \in Z_0$.

Приведенные примеры показывают, что бесконечное множество может быть эквивалентно своему собственному подмножеству. В этом проявляется принципиальное различие между конечными и бесконечными множествами, так как из основного закона конечных множеств следует: часть всегда меньше целого.

Пример 3. Рассмотрим множество L слов произвольной длины в бесконечном алфавите $\mathfrak{L} = \{b_1, b_2, \ldots, b_n, \ldots\}$. Каждому символу $b_n \in \mathfrak{L}$ сопоставим его код — слово $(1^n 0)$ в двухбуквенном алфавите $\mathfrak{A} = \{0, 1\}$, где 1^n — краткая запись последовательности 111...1 длины n. Тогда каждому слову $(b_{i_1} b_{i_2} \ldots b_{i_k}) \in L$ можно поставить в соответствие слово $(1^{i_1} 0\, 1^{i_2} 0 \ldots 1^{i_k} 0)$. Нетрудно видеть, что указанное соответствие взаимно-однозначно и поэтому множество L эквивалентно множеству L^c кодирующих слов в алфавите $\mathfrak{A}$ (т. е. множеству всех слов, начинающихся с единицы, заканчивающихся нулем и не содержащих подслов вида 00). Описанная конструкция реализуется также при конечном алфавите $\mathfrak{L}$ (частный случай так называемого **взаимно-однозначного кодирования**).

Пример 4. Множество всех действительных чисел эквивалентно множеству точек прямой. Взаимно-однозначное соответствие этих множеств устанавливается, например, с помощью числовой прямой, принимаемой за ось абсцисс некоторой координатной системы.

Пример 5. Пусть X — множество точек оси абсцисс, а G — множество точек единичной полуокружности с центром в точке $O_1(0, 1)$, за исключением ее концов $M_1(-1, 1)$, $M_2(1, 1)$ (рис. 6). Полуокружность касается оси абсцисс в начале координат. Множества X и G эквивалентны: их взаимно-однозначное соответствие

можно установить, сопоставляя произвольной точке $x \in X$ точку $q \in G$, в которой луч O_1x пересекает данную окружность.

Пример 6. Множество всех точек x оси абсцисс, для которых выполняются неравенства $a < x < b$, где a, b — некоторые действительные числа, называется **интервалом** (a, b). Если к интервалу (a, b) присоединить его концы, то получится **сегмент** $[a, b]$, т. е. множество точек, удовлетворяющих неравенству $a \leqslant x \leqslant b$. Спроектировав множество G точек полуокружности на интервал $(-1, 1)$ (рис. 7), убедимся в эквивалентности этих множеств. Однако $X \sim G$ (см. предыдущий пример), и, следовательно, множество X всех точек числовой оси эквивалентно ее интервалу $(-1, 1)$.

Очевидно, таким образом можно установить эквивалентность числовой оси и любого ее интервала, а следовательно, и эквивалентность любых двух интервалов.

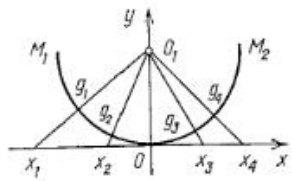


Рис. 6.

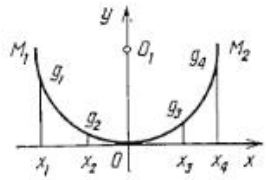


Рис. 7.

Пример 7. Сегмент $[a, b]$ эквивалентен интервалу (a, b). Действительно, рассмотрим произвольную последовательность $S = \{s_1, s_2, \ldots, s_n, \ldots\}$ различных точек сегмента $[a, b]$ такую, что $s_1 = a$, $s_2 = b$. Тогда точки $s_3, \ldots, s_n, \ldots$, как и остальные точки сегмента $[a, b]$, не попавшие в выбранную последовательность S, принадлежат интервалу (a, b). Выполнив сопоставление $s_i \in S \leftrightarrow s_{i+2} \in (a, b)$ и $y \in [a, b] \leftrightarrow y \in (a, b)$, где $y \overline{\in} S$, установим взаимно-однозначное соответствие сегмента $[a, b]$ и интервала (a, b).

Часто рассматриваются множества, эквивалентные множеству $N = \{1, 2, \ldots, n, \ldots\}$ всех натуральных чисел. Такие множества называются **счетными**. Следовательно, некоторое бесконечное множество A счетно, если можно установить взаимно-однозначное соответствие его элементов и натуральных чисел, т. е. указать способ задания элементов множества A (без повторений) в виде бесконечного перечня (пересчета) так, что первый элемент в этом перечне соответствует числу 1, второй — числу 2 и т. д. Таким образом, несмотря на бесконечность пересчета, каждый элемент множества A занимает в нем некоторое конечное положение и число, соответствующее данному элементу, служит его индексом, $A = \{a_1, a_2, \ldots, a_n, \ldots\}$.

Перейдем к изучению некоторых важных свойств счетных множеств. Всякое бесконечное подмножество B счетного множества A также счетно. Действительно, вследствие бесконечности подмножества B можно осуществить пересчет его элементов по порядку их следования в множестве A.

Теорема 1.1. *Объединение конечной или счетной совокупности счетных множеств также является счетным множеством.*

Доказательство. Рассмотрим сначала конечную совокупность счетных множеств $\{A_1, A_2, \ldots, A_k\}$, где $A_i = \{a_1^{(i)}, a_2^{(i)}, \ldots, a_n^{(i)}, \ldots\}$ $(i = 1, 2, \ldots, k)$. Выпишем в строку все элементы множеств $A_1, A_2, \ldots, A_k$:

$$a_1^{(1)}, a_1^{(2)}, \ldots, a_1^{(k)}; \; a_2^{(1)}, a_2^{(2)}, \ldots, a_2^{(k)}; \; \ldots; \; a_n^{(1)}, a_n^{(2)}, \ldots, a_n^{(k)}; \; \ldots$$

Пересчет выписанных элементов можно осуществить по порядку их следования в строке, причем элемент, встречающийся в строке более одного раза, приобретает номер при первой встрече, а затем пропускается. В результате каждый элемент объединения $\bigcup_{i=1}^{k} A_i$ получит свой номер, что и требовалось.

В случае счетной совокупности счетных множеств $\{A_i \mid i = 1, 2, \ldots\}$,

$$A_1 = \{a_1^{(1)}, a_2^{(1)}, \ldots, a_n^{(1)}, \ldots\},$$
$$A_2 = \{a_1^{(2)}, a_2^{(2)}, \ldots, a_n^{(2)}, \ldots\},$$
$$\ldots \ldots \ldots \ldots \ldots$$
$$A_m = \{a_1^{(m)}, a_2^{(m)}, \ldots, a_n^{(m)}, \ldots\},$$
$$\ldots \ldots \ldots \ldots \ldots$$

все их элементы записываются в одну строку по группам элементов с равной суммой верхнего и нижнего индексов в порядке возрастания этой суммы:

$$a_1^{(1)}; \; a_1^{(2)}, a_2^{(1)}; \; a_1^{(3)}, a_2^{(2)}, a_3^{(1)}; \; a_1^{(4)}, a_2^{(3)}, a_3^{(2)}, a_4^{(1)}; \; \ldots$$

В остальном рассуждения аналогичны рассуждениям в случае, рассмотренном выше. Теорема доказана.

Следствие 1. Множество Z всех целых чисел счетно.

Справедливость следствия вытекает из соотношения $Z = N \cup \overline{N}$, где $\overline{N} = \{0, -1, -2, \ldots\}$.

Числовое множество $\mathfrak{M}$ называется **плотным**, если для каждой пары чисел $q_1, q_2 \in \mathfrak{M}$ $(q_1 < q_2)$ всегда существует некоторое число $q \in \mathfrak{M}$ такое, что $q_1 < q < q_2$. Из плотности множества $\mathfrak{M}$ следует его бесконечность, более того, для каждой пары чисел $q_1, q_2 \in \mathfrak{M}$ существует бесконечное множество чисел $q \in \mathfrak{M}$, для которых выполняется неравенство $q_1 < q < q_2$.

Всякое подмножество $Z' \subseteq Z$ множества всех целых чисел не плотно. В отличие от множества Z множество R всех рациональных чисел $\left(\text{т. е. чисел } \frac{p}{q}, \text{ где } p \text{ и } q \text{ целые}\right)$ плотно. Однако, несмотря на плотность множества R, для него справедливо такое следствие.

Следствие 2. Множество R всех рациональных чисел счетно.

Действительно, множество R является объединением счетных множеств

A_1 — всех целых чисел $n = 0, \pm 1, \pm 2, \ldots$,

A_2 — всех дробей вида $\frac{n}{2}$, $n = 0, \pm 1, \pm 2, \ldots$,

A_3 — всех дробей вида $\frac{n}{3}$, $n = 0, \pm 1, \pm 2, \ldots$,

. . . . . .

A_m — всех дробей вида $\frac{n}{m}$, $n = 0, \pm 1, \pm 2, \ldots$,

. . . . . .

Следствие 3. Декартово произведение $A \times B$ счетных множеств $A = \{a_1, a_2, \ldots, a_n, \ldots\}$ и $B = \{b_1, b_2, \ldots, b_n, \ldots\}$ счетно.

Действительно, множество всех пар (a_i, b_j) $(i, j = 1, 2, \ldots)$ можно представить как объединение счетной совокупности счетных множеств

$$A_1 = \{(a_1, b_1), (a_1, b_2), \ldots, (a_1, b_n), \ldots\},$$
$$A_2 = \{(a_2, b_1), (a_2, b_2), \ldots, (a_2, b_n), \ldots\},$$
. . . . . .
$$A_m = \{(a_m, b_1), (a_m, b_2), \ldots, (a_m, b_n), \ldots\},$$
. . . . . .

В частности, множество всех точек плоскости, обе координаты которых являются рациональными числами, счетно.

Используя индукцию, нетрудно показать, что декартово произведение $A_1 \times A_2 \times \cdots \times A_n$, где $A_1, A_2, \ldots, A_n$ — счетные множества, также счетно при любом n. Действительно, при $n = 1$ A_1 счетно по условию. Пусть $A_1 \times A_2 \times \cdots \times A_{n-1}$ счетно. Тогда по доказанному ранее декартово произведение $A_1 \times A_2 \times \cdots \times A_{n-1} \times A_n$ также счетно.

Следствие 4. Множество P всех многочленов $p(x) = a_0 x^n + a_1 x^{n-1} + \cdots + a_{n-1} x + a_n$ произвольной степени с рациональными коэффициентами a_i $(i = 0, 1, \ldots, n)$ счетно.

Множество P представляет собой объединение счетной совокупности множеств $\{P_n \mid n = 0, 1, 2, \ldots\}$, где P_n означает множество многочленов степени, не превышающей n. Каждому многочлену $p(x) = a_0 x^n + a_1 x^{n-1} + \cdots + a_{n-1} x + a_n$ можно сопоставить последовательность рациональных чисел $(a_0, a_1, \ldots, a_n)$, и поэтому $P_n \sim \underbrace{(R \times R \times \cdots \times R)}_{n \text{ раз}}$. Тогда в силу следствия 3 множество P_n счетно, а поэтому счетно и множество P.

Следствие 5. Множество всех алгебраических чисел (т. е. корней многочленов с рациональными коэффициентами) счетно.

Согласно следствию 4 множество P всех многочленов с рациональными коэффициентами счетно. Известно, что каждый такой многочлен имеет некоторое конечное число корней. Таким образом, множество всех алгебраических чисел можно представить как объединение счетной совокупности конечных множеств, которое, очевидно, счетно.

Следствие 6. Множество $F(\mathfrak{A})$ всех слов конечной длины в алфавите $\mathfrak{A} = \{a_1, a_2, \ldots, a_m\}$ счетно.

Действительно, множество $F(\mathfrak{A})$ можно представить как объединение счетной совокупности $\{A_n \mid n = 0, 1, 2, \ldots\}$, где A_n — множество всех слов в алфавите $\mathfrak{A}$ длины, не превышающей n. Очевидно, каждое множество A_n конечно, откуда следует счетность множества $F(\mathfrak{A})$.

§ 1.4. Континуальные множества

Среди бесконечных множеств существуют множества, элементы которых не поддаются пересчету. Такие множества принято называть **несчетными**. Существование несчетных множеств впервые было доказано Г. Кантором с помощью предложенного им диагонального метода.

Теорема 1.2 (Кантора). *Множество всех действительных чисел интервала* $(0, 1)$ *числовой оси несчетно.*

Доказательство. Как известно, каждому действительному числу из интервала $(0, 1)$ можно сопоставить правильную бесконечную десятичную дробь $0{,}a_1a_2a_3 \ldots$, которая содержит бесконечно много цифр, отличных от нуля. Так, если число представимо в виде конечной десятичной дроби (т. е. дроби с повторяющимися нулями), то ему сопоставляется бесконечная десятичная дробь с повторяющимися девятками. Например, числу 0,576 (или 0,576000...) соответствует дробь 0,575999... Такое соответствие действительных чисел интервала $(0, 1)$ и правильных бесконечных десятичных дробей взаимно-однозначно.

Предположим теперь, что теорема неверна и множество всех действительных чисел интервала $(0, 1)$ счетно, т. е. для этого множества существует пересчет $x_1, x_2, x_3, x_4, \ldots, x_n, \ldots$ Рас-

положим одну под другой правильные десятичные дроби, соответствующие элементам данного пересчета, так, что

$$
\begin{aligned}
&x_1 \leftrightarrow 0,\ a_{11}a_{12}a_{13}a_{14}\ \ldots\ a_{1n}\ \ldots,\\
&x_2 \leftrightarrow 0,\ a_{21}a_{22}a_{23}a_{24}\ \ldots\ a_{2n}\ \ldots,\\
&x_3 \leftrightarrow 0,\ a_{31}a_{32}a_{33}a_{34}\ \ldots\ a_{3n}\ \ldots,\\
&x_4 \leftrightarrow 0,\ a_{41}a_{42}a_{43}a_{44}\ \ldots\ a_{4n}\ \ldots,\\
&\ldots\ldots\ldots\ldots\ldots\ldots\ldots,\\
&x_n \leftrightarrow 0,\ a_{n1}a_{n2}a_{n3}a_{n4}\ \ldots\ a_{nn}\ \ldots,\\
&\ldots\ldots\ldots\ldots\ldots\ldots\ldots.
\end{aligned}
$$

Следуя по диагонали (указанной стрелками), образуем новую дробь $0,\ a'_{11}a'_{22}a'_{33}a'_{44}\ \ldots\ a'_{nn}\ \ldots$ такую, что при $n = 1, 2, \ldots$

$$a'_{nn} = \begin{cases} 1, & \text{если } a_{nn} \neq 1, \\ 2, & \text{если } a_{nn} = 1, \end{cases}$$

причем вновь полученная дробь должна остаться бесконечной (для этого достаточно предположить, что $a'_{nn} \neq 0$ при любом n). Данная дробь $0,\ a'_{11}a'_{22}a'_{33}\ \ldots\ a'_{nn}$ соответствует некоторому действительному числу x, которое принадлежит интервалу $(0, 1)$, но не входит в рассматриваемый пересчет. Действительно, построенная дробь отличается от каждой из дробей в указанном пересчете цифрой, расположенной по диагонали. Так, от дроби, соответствующей числу x_1, она отличается своей первой цифрой после запятой, от дроби, соответствующей числу x_2, — второй цифрой после запятой и т. д. Следовательно, для множества всех действительных чисел интервала $(0, 1)$ пересчета не существует. Таким образом, наше предположение относительно счетности этого множества неверно. Теорема доказана.

Всякое множество, эквивалентное множеству всех действительных чисел интервала $(0, 1)$, называется **континуальным или множеством мощности континуума**. В § 1.3 показано, что множества действительных чисел (точек) любого интервала (a, b), любого сегмента $[a, b]$ и, наконец, всей числовой оси $-\infty < x < \infty$ эквивалентны множеству точек интервала $(0, 1)$. Таким образом, каждое из этих множеств континуально.

Справедливо следующее утверждение: если A — некоторое несчетное множество и $B \subset A$ — счетное подмножество множества A, то множество $C = A \setminus B$, полученное в результате исключения из A всех элементов множества B, также несчетно. Действительно, из предположения о счетности множества C следует счетность множества $A = B \cup C$ (см. § 1.3, теорема 1.1).

Следствие 1. Множество $\bar{R}$ всех иррациональных чисел является множеством несчетной мощности.

Числа, не являющиеся корнями многочленов с рациональными коэффициентами, называются **трансцендентными.**

Следствие 2. Множество T всех трансцендентных чисел является множеством несчетной мощности.

Следствия 1 и 2 справедливы в силу того, что множества всех рациональных и всех алгебраических чисел счетны, а множество V всех действительных чисел является множеством мощности континуума.

Приведенные результаты доказывают само существование трансцендентных чисел. Первые трансцендентные числа построены Ж. Лиувиллем (1844 г.), причем их построение было сопряжено со значительными трудностями. Канторовский диагональный метод не только доказывает существование трансцендентных чисел, но и позволяет строить их по некоторому пересчету множества алгебраических чисел.

Теорема 1.3. *Множество $\mathfrak{M}(A)$ всех подмножеств некоторого счетного множества A является множеством мощности континуума.*

Доказательство. По определению счетного множества A существует взаимно-однозначное соответствие элементов этого множества и натуральных чисел, заданное в виде пересчета $a_1, a_2, \ldots, a_n, \ldots \in A$. Следовательно, для доказательства теоремы достаточно рассмотреть случай, когда в качестве множества A выбрано множество N всех натуральных чисел. Каждому подмножеству $N' \subseteq N$ соответствует его характеристическое свойство (характеристическая функция) $\varphi_{N'}(n)$ такое, что

$$\varphi_{N'}(n) = \begin{cases} 0, & \text{если } n \in N', \\ 1, & \text{если } n \overline{\in} N' \text{ при любом } n = 1, 2, \ldots \end{cases}$$

Это означает, что каждому подмножеству $N' \subseteq N$ можно сопоставить последовательность значений его характеристической функции $\varphi_{N'}(n)$ (т. е. бесконечную цепочку, состоящую из нулей и единиц) так, что $N' \leftrightarrow \varphi_{N'}(1), \varphi_{N'}(2), \ldots, \varphi_{N'}(k), \ldots$, причем указанное соответствие взаимно-однозначно.

Предположим теперь, что множество $\mathfrak{M}(N)$ счетно и существует пересчет $N'_1, N'_2, N'_3, \ldots, N'_k, \ldots$ его элементов. Расположим одну под другой бесконечные последовательности значений характеристических функций, соответствующих элементам данного пересчета, так, что

$$
\begin{aligned}
&N'_1 \leftrightarrow \varphi_{N'_1}(1),\ \varphi_{N'_1}(2),\ \varphi_{N'_1}(3),\ \ldots,\ \varphi_{N'_1}(k),\ \ldots,\\
&N'_2 \leftrightarrow \varphi_{N'_2}(1),\ \varphi_{N'_2}(2),\ \varphi_{N'_2}(3),\ \ldots,\ \varphi_{N'_2}(k),\ \ldots,
\end{aligned}
$$

$$N'_1 \leftrightarrow \varphi_{N'_1}(1),\ \varphi_{N'_1}(2),\ \varphi_{N'_1}(3),\ \ldots,\ \varphi_{N'_1}(k),\ \ldots;$$
$$\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots$$
$$N'_k \leftrightarrow \varphi_{N'_k}(1),\ \varphi_{N'_k}(2),\ \varphi_{N'_k}(3),\ \ldots,\ \varphi_{N'_k}(k),\ \ldots,$$
$$\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots$$

По диагонали (следуя стрелкам) $\varphi_{N'_1}(1)$, $\varphi_{N'_2}(2)$, $\ldots$, $\varphi_{N'_k}(k)$, $\ldots$, $\varphi_{N'_n}(n)$, $\ldots$ построим новую последовательность $\tilde{\varphi}_{N'}(1)$, $\tilde{\varphi}_{N'}(2)$, $\ldots$, $\tilde{\varphi}_{N'}(n)$, $\ldots$ такую, что $\tilde{\varphi}_{N'}(n) = \varphi_{N'_n}(n) \oplus 1$ при любом $n = 1, 2, \ldots$ ($x \oplus y$ — сложение по mod 2, удовлетворяющее правилам $0 \oplus 1 = 1 \oplus 0 = 1$, $1 \oplus 1 = 0 \oplus 0 = 0$). Эта последовательность является последовательностью значений характеристической функции $\tilde{\varphi}_{N'}(n)$, соответствующей некоторому подмножеству $N' \subset N$. Очевидно, подмножество N' не входит в рассматриваемый пересчет, так как по построению $\tilde{\varphi}_{N'}(n) \neq \varphi_{N'_n}(n)$ $(n = 1, 2, \ldots)$, и, следовательно, функция $\tilde{\varphi}_{N'}(n)$ отлична от любой из характеристических функций, соответствующих элементам данного пересчета. Это означает, что для множества $\mathfrak{M}(N)$ не существует пересчета всех подмножеств множества N. Таким образом, множество $\mathfrak{M}(N)$, а следовательно, и множество $\mathfrak{M}(A)$ несчетны для любого счетного множества A.

В заключение заметим, что каждой последовательности значений $\varphi_{N'}(1)$, $\varphi_{N'}(2)$, $\ldots$, $\varphi_{N'}(k)$, $\ldots$ некоторой характеристической функции $\varphi_{N'}(n)$ можно поставить в соответствие правильную бесконечную двоичную дробь $0, \varphi_{N'}(1)\varphi_{N'}(2)\ldots\varphi_{N'}(k)\ldots$, которая представляет некоторое действительное число из интервала $(0, 1)$ в двоичной системе счисления, причем каждое число из интервала $(0, 1)$ однозначно (с точностью до представления $0{,}0100\ldots = 0{,}00111\ldots$) представимо в виде правильной бесконечной двоичной дроби (точно так же, как оно представимо в виде правильной бесконечной десятичной дроби).

Таким образом, множество $\mathfrak{M}(A)$ эквивалентно множеству всех действительных чисел интервала $(0, 1)$ и, следовательно, является множеством мощности континуума. Теорема доказана.

Рассмотрим множество $F(\mathfrak{A})$ всех слов в некотором конечном алфавите $\mathfrak{A}$. Любое подмножество $L \subseteq F(\mathfrak{A})$ называется **языком над алфавитом $\mathfrak{A}$**.

Следствие. Множество всех языков над конечным алфавитом $\mathfrak{A}$ является множеством мощности континуума. Справедливость следствия вытекает из счетности множества $F(\mathfrak{A})$ (см. § 1.3).

§ 1.5. Кардинальные числа

Рассмотрим совокупность $\{A\}$ всех множеств, эквивалентных некоторому произвольно выбранному множеству A. Очевидно, множества, входящие в $\{A\}$, эквивалентны. Под **кардинальным числом** $\overline{\overline{A}}$ понимается некоторый объект, определяющий мощность любого из множеств рассматриваемой совокупности. В случае конечности множества A кардинальным числом $\overline{\overline{A}}$ любого из множеств, входящих в $\{A\}$, является натуральное число, определяющее число элементов в каждом из множеств данной совокупности[3].

Обратимся к важному вопросу сравнения кардинальных чисел. Кардинальные числа $\overline{\overline{A}}$ и $\overline{\overline{B}}$ конечных множеств A и B, как подчеркивалось выше, являются натуральными числами, и поэтому возможно одно из следующих трех соотношений: $\overline{\overline{A}} = \overline{\overline{B}}$, $\overline{\overline{A}} < \overline{\overline{B}}$ и $\overline{\overline{A}} > \overline{\overline{B}}$.

При сравнении бесконечных множеств A и B логически возможны четыре случая.

1. Множества A и B эквивалентны, $A \sim B$. В этом случае $\overline{\overline{A}} = \overline{\overline{B}}$.

2. Множества A и B неэквивалентны, но одно из них, например A, эквивалентно подмножеству другого, $A \sim B' \subset B$. Тогда говорят, что мощность множества A меньше мощности множества B, $\overline{\overline{A}} < \overline{\overline{B}}$.

3. Множество A эквивалентно некоторому подмножеству множества B, и, наоборот, множество B эквивалентно некоторому подмножеству множества A, $A \sim B' \subset B$ и $B \sim A' \subset A$. Этот случай, как будет показано далее, сводится к первому.

4. Множество A не эквивалентно никакому подмножеству множества B и множество B не эквивалентно никакому подмножеству множества A (множества A и B несравнимы).

Четвертый случай невозможен и множество всех кардинальных чисел вполне упорядочено (см. § 2.5)[4]. Рассмотрение третьего случая исчерпывается следующей важной теоремой.

Теорема 1.4 (Кантора — Бернштейна). *Если множество A эквивалентно некоторому подмножеству множества B, $A \sim B_1 \subset B$, и одновременно множество B эквивалентно некоторому подмножеству множества A, $B \sim A_1 \subset A$, то множества A и B эквивалентны.*

---

[3] Канторовский символ $\overline{\overline{A}}$ означает, что при определении кардинального числа множества A происходит двойное абстрагирование от природы элементов множества A и их порядка.

[4] См. также работу Ф. Хаусдорфа [108] или П. С. Александрова [1].

Доказательство. Пусть даны множества A_1 и B_1, причем $A \sim B_1 \subset B$ и $B \sim A_1 \subset A$. Предполагается, что подмножества A_1 и B_1 являются собственными подмножествами множеств A и B, так как в противном случае справедливость теоремы очевидна. Ввиду $B \sim A_1$ достаточно показать, что $A_1 \sim A$.

Из $B \sim A_1$, поскольку $B_1 \subset B$, следует существование подмножества $A_2 \subset A_1$ такого, что $A_2 \sim B_1$. Но по условию теоремы $A \sim B_1$ и, следовательно, $A \sim A_2$. Это означает, что существует взаимно-однозначное соответствие φ элементов множеств A и A_2, $x \overset{\varphi}{\leftrightarrow} x' \mid x \in A,\ x' \in A_2$. Далее, в силу принятого закона соответствия φ подмножеству $A_1 \subset A$ соответствует некоторое подмножество $A_3 \subset A_2$ такое, что $A_1 \sim A_3$. В свою очередь, по этому же закону φ, подмножеству $A_2 \subset A_1$ соответствует подмножество $A_4 \subset A_3$ такое, что $A_2 \sim A_4$. Продолжив эти рассуждения, получим бесконечную цепочку строгих включений $A \supset A_1 \supset A_2 \supset A_3 \supset A_4 \supset \cdots \supset A_n \supset \cdots$, причем

$$A \sim A_2 \sim A_4 \sim \cdots \sim A_{2k} \sim A_{2(k+1)} \sim \cdots,$$
$$A_1 \sim A_3 \sim A_5 \sim \cdots \sim A_{2k+1} \sim A_{2k+3} \sim \cdots$$

Кроме того, в силу принятого закона соответствия φ из построения множеств A_2, A_3, ... следуют эквивалентности

$$(A \setminus A_1) \sim (A_2 \setminus A_3) \sim (A_4 \setminus A_5) \sim \cdots \sim (A_{2k} \setminus A_{2k+1}) \sim \cdots,$$
$$(A_1 \setminus A_2) \sim (A_3 \setminus A_4) \sim (A_5 \setminus A_6) \sim \cdots \sim (A_{2k+1} \setminus A_{2k+2}) \sim \cdots$$

Действительно, при любом n по принятому закону взаимно-однозначного соответствия φ $A_n \sim A_{n+2}$ и $A_{n+1} \sim A_{n+3}$, причем $A_{n+1} \subset A_n$, $A_{n+3} \subset A_{n+2}$. Если теперь из множеств A_n и A_{n+2} исключить подмножества A_{n+1} и A_{n+3} соответственно, то разности $A_n \setminus A_{n+1}$ и $A_{n+2} \setminus A_{n+3}$ останутся эквивалентными:

$$(A_n \setminus A_{n+1}) \sim (A_{n+2} \setminus A_{n+3}). \tag{1.4}$$

Пусть $D = \bigcap_{i=1}^{\infty} A_i$ — пересечение множеств $A_1, A_2, \ldots$ (в частности, D может оказаться пустым). Тогда множества A и A_1 можно представить в виде объединения попарно-непересекающихся множеств так, что

$$\left.\begin{aligned} A &= D \cup (A \setminus A_1) \cup (A_1 \setminus A_2) \cup (A_2 \setminus A_3) \cup \cdots \\ &\quad \cdots \cup (A_n \setminus A_{n+1}) \cup \cdots, \\ A_1 &= D \cup (A_1 \setminus A_2) \cup (A_2 \setminus A_3) \cup (A_3 \setminus A_4) \cup \cdots \\ &\quad \cdots \cup (A_{n+1} \setminus A_{n+2}) \cup \cdots \end{aligned}\right\} \quad (*)$$

Пусть, далее,

$$D_0 = D \cup (A_1 \setminus A_2) \cup (A_3 \setminus A_4) \cup \cdots \cup (A_{2k+1} \setminus A_{2k+2}) \cup \cdots$$

Тогда равенства $(*)$ можно переписать в виде

$$\left.\begin{aligned} A &= D_0 \cup (A \setminus A_1) \cup (A_2 \setminus A_3) \cup (A_4 \setminus A_5) \cup \cdots \\ &\quad \cdots \cup (A_{2k} \setminus A_{2k+1}) \cup \cdots, \\ A_1 &= D_0 \cup (A_2 \setminus A_3) \cup (A_4 \setminus A_5) \cup (A_6 \setminus A_7) \cup \cdots \\ &\quad \cdots \cup (A_{2(k+1)} \setminus A_{2(k+1)+1}) \cup \cdots \end{aligned}\right\} \quad (**)$$

Теперь нетрудно установить взаимно-однозначное соответствие элементов множеств A и A_1, сопоставив их по следующему закону: каждому элементу $x \in A$ такому, что $x \in D_0$, сопоставим тот же элемент $x \in A_1$; одновременно каждому элементу $x \in A \setminus D_0$ сопоставим элемент $x' \in A_1 \setminus D_0$ такой, что $x \overset{\varphi}{\leftrightarrow} x'$. Таким образом, из равенств $(**)$ и соотношения (1.4) следует эквивалентность $A \sim A_1$, а значит, и $A \sim B$. Теорема доказана.

Следствие 1. Если справедливы включения $A \supset A_1 \supset A_2$, причем $A \sim A_2$, то множества A и A_1 эквивалентны. Иными словами, мощность промежуточного множества A_1 совпадает с мощностями крайних множеств A, A_2; $\overline{\overline{A}}_1 = \overline{\overline{A}} = \overline{\overline{A}}_2$.

Справедливость следствия вытекает из эквивалентностей $A \sim A_2 \subset A_1$; $A_1 \sim A_1 \subset A$.

Следствие 2. Если $A \supseteq B$, то $\overline{\overline{A}} \geqslant \overline{\overline{B}}$.

Кардинальные числа бесконечных множеств называются **бесконечными** или **трансфинитными**, кардинальными числами. Кардинальное число множества N всех натуральных чисел, а следовательно, и любого счетного множества обозначается через $\aleph_0$ (алеф-нуль).

Следствие 3. Пусть A — произвольное конечное множество. Тогда $\overline{\overline{A}} < \aleph_0$.

Теорема 1.5. *Во всяком бесконечном множестве A можно выделить некоторое счетное подмножество.*

Действительно, выберем в множестве A некоторый элемент a_1 и рассмотрим множество $A_1 = A \setminus \{a_1\}$. Очевидно, множество A_1 также бесконечно и, следовательно, можно выбрать из него некоторый элемент $a_2 \in A_1$ и перейти к множеству $A_2 = A_1 \setminus \{a_2\}$. Продолжив этот процесс неограниченно, в множестве A выделим счетное подмножество $B = \{a_1, a_2, \ldots, a_n, \ldots\}$.

Таким образом, среди бесконечных множеств наименьшие по мощности счетные множества с кардинальным числом $\aleph_0$. Возникает вопрос: существуют ли множества наибольшей мощности? В § 1.4 рассмотрены множества мощности континуума с кардинальным числом $\aleph_1 = 2^{\aleph_0}$. Обобщая приведенные выше результаты, можно доказать следующее важное утверждение, принадлежащее Г. Кантору.

Теорема 1.6. *Мощность множества $\mathfrak{M}(A)$ всех подмножеств любого непустого множества A больше мощности данного множества A, $\overline{\overline{\mathfrak{M}(A)}} > \overline{A}$.*

Доказательство. Достаточно доказать, что множества $\mathfrak{M}(A)$ и A неэквивалентны, причем существует подмножество $B \subset \mathfrak{M}(A)$ такое, что $A \sim B$. Множество $\mathfrak{M}(A)$ наряду с другими подмножествами содержит в качестве своих элементов несобственные подмножества множества A (т. е. подмножество $\varnothing$ и само множество A), а также его одноэлементные подмножества.

Докажем, что множества $\mathfrak{M}(A)$ и A неэквивалентны. Предположим противное: пусть $\mathfrak{M}(A) \sim A$. Тогда существует взаимнооднозначное соответствие φ элементов множества A и его подмножеств, $a \overset{\varphi}{\leftrightarrow} A'$ $(a \in A;\ A' \in \mathfrak{M}(A))$.

Возможна следующая альтернатива: элемент a принадлежит соответствующему ему подмножеству A' или не принадлежит. В первом случае элемент a назовем отмеченным, во втором — неотмеченным. В множестве A существуют как отмеченные, так и неотмеченные элементы. Действительно, несобственному подмножеству $A \in \mathfrak{M}(A)$ в соответствии φ, очевидно, сопоставлен отмеченный элемент множества A; в то же время пустому подмножеству $\varnothing \in \mathfrak{M}(A)$ сопоставлен неотмеченный элемент данного множества A.

Рассмотрим подмножество $A_0 \subset A$, состоящее из всех неотмеченных элементов множества A. Пусть этому элементу $A_0 \in \mathfrak{M}(A)$ в соответствии φ сопоставлен элемент $a_0 \in A$. Очевидно, что элемент a_0 не относится к отмеченным, так как принадлежность $a_0 \in A_0$ противоречит определению подмножества A_0, состоящего из всех неотмеченных элементов множества A. Вместе с тем элемент a_0 нельзя считать неотмеченным, так как тогда он входил бы в соответствующее ему подмножество A_0 и поэтому был бы отмеченным. Таким образом, элемент a_0 не может быть ни отмеченным, ни неотмеченным, что противоречит высказанной выше альтернативе, а следовательно, и предположению об эквивалентности множеств $\mathfrak{M}(A)$ и A. Выбрав в качестве $B \subset \mathfrak{M}(A)$ множества всех одноэлементных подмножеств множества A, легко убедиться, что $A \sim B$. Значит, множества $\mathfrak{M}(A)$ и A неэквивалентны и существует подмножество $B \subset \mathfrak{M}(A)$ такое, что $A \sim B$. Следовательно, $\overline{\overline{\mathfrak{M}(A)}} > \overline{A}$. Теорема доказана.

Следствие. Не существует множеств наибольшей мощности. Иными словами, множество всех кардинальных чисел не имеет максимального элемента.

Действительно, при рассмотрении множеств N, $\mathfrak{M}(N)$, $\mathfrak{M}(\mathfrak{M}(N))$, $\mathfrak{M}(\mathfrak{M}(\mathfrak{M}(N)))$, ... получим бесконечно возрастающую последовательность трансфинитных кардинальных чисел $\aleph_0$, $\aleph_1 = 2^{\aleph_0}$, $\aleph_2 = 2^{2^{\aleph_0}}$, ...

К наиболее интересным проблемам теории множеств относятся **континуум-гипотеза**, состоящая в том, что любое несчетное множество содержит подмножество мощности континуума (т. е. для счетных и континуальных множеств не существует множеств с кардинальным числом $\aleph$ таким, что $\aleph_0 < \aleph < \aleph_1$), и **обобщенная континуум-гипотеза**, состоящая в том, что для любого бесконечного кардинального числа $\aleph$ любое кардинальное число $\aleph' > \aleph$ удовлетворяет неравенству $\aleph' \geqslant 2^{\aleph}$. Решение этих проблем является одним из самых интересных математических результатов. Именно за этот результат на Международном математическом конгрессе в Москве (1966 г.) П. Коэн — профессор Стэнфордского университета — был удостоен медали Филдса, присуждающейся за выдающиеся достижения в области математики [64].

§ 2.1. Основные определения

Теория отношений — одна из ветвей общей алгебры — оказалась полезной при изучении многих математических проблем, в том числе проблем обоснования математики. Широко используется она и в различных областях кибернетики, в частности в программировании.

Рассмотрим произвольные множества $A_1, A_2, \ldots, A_n$ (не обязательно различные). Под ***n*-арным** отношением, или ***n*-отношением**, ρ^n на множествах $A_1, A_2, \ldots, A_n$ понимается закон (характеристическое свойство), выделяющий в декартовом произведении $A_1 \times A_2 \times \cdots \times A_n$ некоторое подмножество $\rho^n_{A_1, \ldots, A_n} \subseteq A_1 \times A_2 \times \cdots \times A_n$, называемое **графиком** отношения ρ^n. Если $A_1 = A_2 = \cdots = A_n = A$, то говорят об n-отношении ρ^n на множестве A с графиком $\rho^n_A \subseteq A^n$. Отношения обозначаются греческими буквами (с индексами или без них) α, β, γ, ... и специальными символами $=, <, >, \leqslant, \geqslant, \infty$ и др.

Часто понятие n-отношения отождествляется с его графиком, т. е. под n-отношением ρ^n на множествах $A_1, A_2, \ldots, A_n$ понимается само подмножество $\rho^n_{A_1, \ldots, A_n} \subseteq A_1 \times A_2 \times \cdots \times A_n$.

Если $\tilde{\alpha} = (a_i^{(1)}, a_i^{(2)}, \ldots, a_i^{(n)}) \in \rho^n_{A_1, \ldots, A_n}$ (где $a_i^{(j)} \in A_j$, $j = 1, 2, \ldots, n$), то говорят, что элементы $a_i^{(1)}, a_i^{(2)}, \ldots, a_i^{(n)}$ находятся в отношении ρ^n: $\rho^n(a_i^{(1)}, a_i^{(2)}, \ldots, a_i^{(n)})$, так что обозначения $(a_i^{(1)}, a_i^{(2)}, \ldots, a_i^{(n)}) \in \rho^n_{A_1, \ldots, A_n}$ и $\rho^n(a_i^{(1)}, a_i^{(2)}, \ldots, a_i^{(n)})$ равносильны. Последовательность $\tilde{\alpha} = (a_i^{(1)}, a_i^{(2)}, \ldots, a_i^{(n)}) \in \rho^n_{A_1, \ldots, A_n}$ называется **элементом** или **вектором** n-отношения ρ^n. Отношения, графики которых состоят из конечного множества векторов, называются **конечными** n-отношениями. Если $\rho^n_{A_1, \ldots, A_n} = \varnothing$, то n-отношение ρ^n называют **пустым** ($\wedge^n$); если $\rho^n_{A_1, \ldots, A_n} = A_1 \times A_2 \times \cdots \times A_n$, говорят об **универсальном** отношении ($\vee^n$).

Поскольку n-отношения ρ^n можно рассматривать как подмножества декартова произведения $A_1 \times A_2 \times \cdots \times A_n$, существуют различные способы задания n-отношений, аналогичные способам задания множеств (см. § 1.1). В частности, график $\rho^n_{A_1, \ldots, A_n}$ удобно задавать матрицей, строками которой являются векторы отношения ρ^n. Например, если $\{\tilde{\alpha}_j = (a_{i_j}^{(1)}, a_{i_j}^{(2)}, \ldots, a_{i_j}^{n}) \mid j = 1, 2, \ldots, r\}$ — множество всех векторов отношения ρ^n, то график этого отношения представим в матричной форме

$$\rho^n_{A_1, \ldots, A_n} = \begin{bmatrix} a_{i_1}^{(1)}, & a_{i_1}^{(2)}, & \ldots, & a_{i_1}^{(n)} \\ a_{i_2}^{(1)}, & a_{i_2}^{(2)}, & \ldots, & a_{i_2}^{(n)} \\ \ldots & \ldots & \ldots & \ldots \\ a_{i_r}^{(1)}, & a_{i_r}^{(2)}, & \ldots, & a_{i_r}^{(n)} \end{bmatrix}.$$

Отношения ρ^1 ($n = 1$) на множестве A называют **унарными**, отношения ρ^2 ($n = 2$) на множествах A, B — **бинарными**, отношения ρ^3 ($n = 3$) на множествах A, B, C — **тернарными**.

Унарное отношение ρ^1 на множестве A является характеристическим свойством некоторого подмножества $\rho^1_A \subseteq A$ — графика данного отношения. Таким образом, множество всех унарных отношений на A совпадает с множеством всех подмножеств множества A. Если $A = \{a_1, a_2, \ldots, a_n\}$, то число унарных отношений на A равно 2^n. Так, унарными на множестве $N = \{1, 2, \ldots\}$ являются отношения α^1, β^1, γ^1 с графиками

$$\alpha^1_N = \{3, 7, 10, 11, \ldots, 19\},$$
$$\beta^1_N = \{n \mid \text{при любом } n \geqslant k\},$$
$$\gamma^1_N = \{n \mid \text{при любом } n \neq r\}.$$

Отношения «быть четным числом», «быть нечетным числом» на множестве N такие унарны, их графиками являются множества $N_0 = \{2n \mid n = 1, 2, \ldots\}$ и $N_1 = \{2n - 1 \mid n = 1, 2, \ldots\}$ соответственно.

Заметим, что любому n-отношению ρ^n на множествах $A_1, A_2, \ldots, A_n$ соответствует унарное отношение ρ^1 на множестве $A_1 \times A_2 \times \cdots \times A_n$ такое, что $\rho^1(\tilde{\alpha})$ тогда и только тогда, когда $\rho^n(a_i^{(1)}, a_i^{(2)}, \ldots, a_i^{(n)})$, где $\tilde{\alpha} = (a_i^{(1)}, a_i^{(2)}, \ldots, a_i^{(n)})$, — произвольный вектор отношения ρ^n.

Бинарное отношение ρ на множествах A, B определяется графиком $\rho_{A,B} \subseteq A \times B$. Если элементы $a \in A$ и $b \in B$ находятся в отношении ρ, то наряду с обозначениями $(a, b) \in \rho_{A,B}$ и $\rho(a, b)$ часто пишут также $a \rho b$. Существуют различные (наряду с матричным) способы задания графиков бинарных отношений. К наиболее

употребляемым относятся методы представления бинарных отношений с помощью таблиц и стрелок. Табличный метод представления бинарного отношения ρ на множествах $A = \{a_1, a_2, \ldots, a_r\}$, $B = \{b_1, b_2, \ldots, b_s\}$ сводится к построению таблицы, строки которой соответствуют элементам множества A, а столбцы — элементам множества B; на пересечении i-й строки и j-го столбца ставится знак $\times$, если элементы a_i, b_j находятся в ρ-отношении, $a_i \rho b_j$. При стрелочной записи бинарного отношения ρ на множествах A, B элементы множеств A и B изображаются в виде точек плоскости, а затем точки a_i и b_j соединяются стрелкой, направленной от a_i к b_j, тогда и только тогда, когда $a_i \rho b_j$.

ρ	2	3	4	5	6
2	×		×		×
3		×			×
5				×	

$\{A\}$

Рис. 8.

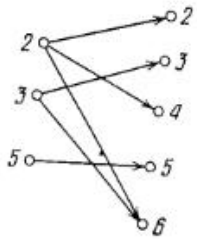


Рис. 9.

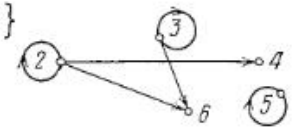


Рис. 10.

График тернарного отношения ρ^3 имеет вид $\rho^3_{A,B,C} \subseteq A \times B \times C$. Например, с каждой бинарной операцией $F(x, y)$, и в частности с арифметическими операциями «+», «—», «·», «:» и другими, связано тернарное отношение φ^3 такое, что $\varphi^3(x, y, z)$ тогда и только тогда, когда $F(x, y) = z$.

Приведем примеры n-арных отношений.

Пример 1. Пусть $A = \{2, 3, 5\}$, $B = \{2, 3, 4, 5, 6\}$. График бинарного отношения ρ на множествах A, B такого, что $a \rho b$ тогда и только тогда, когда a — делитель b, имеет вид

$$\rho_{A,B} = \begin{bmatrix} 2, 2 \\ 2, 4 \\ 2, 6 \\ 3, 3 \\ 3, 6 \\ 5, 5 \end{bmatrix}$$

(табличное представление отношения ρ изображено на рис. 8, стрелочное — на рис. 9; в последнем случае используется включение $A \subset B$, рис. 10, петля a означает $a \rho a$).

Пример 2. На множестве $N = \{1, 2, 3, \ldots\}$ бинарные отношения $<$, $>$, $=$, $\leqslant$, $\geqslant$ определяются соответственно графиками

$$<_N = \begin{bmatrix} 1, 2 \\ 1, 3 \\ \ldots \\ 2, 3 \\ 2, 4 \\ \ldots \\ k, k+1 \\ k, k+2 \\ \ldots \end{bmatrix}, \quad >_N = \begin{bmatrix} 2, 1 \\ 3, 1 \\ \ldots \\ 3, 2 \\ 4, 2 \\ \ldots \\ k+1, k \\ k+2, k \end{bmatrix}, \quad =_N = \begin{bmatrix} 1, 1 \\ 2, 2 \\ 3, 3 \\ \ldots \end{bmatrix},$$

$$\geqslant_N = \begin{bmatrix} 1, 1 \\ 2, 1 \\ 3, 1 \\ \ldots \\ 2, 2 \\ 3, 2 \\ 4, 2 \\ \ldots \\ k, k \\ k+1, k \\ k+2, k \\ \ldots \end{bmatrix}, \quad \leqslant_N = \begin{bmatrix} 1, 1 \\ 1, 2 \\ 1, 3 \\ \ldots \\ 2, 2 \\ 2, 3 \\ 2, 4 \\ \ldots \\ k, k \\ k, k+1 \\ k, k+2 \\ \ldots \end{bmatrix}.$$

Пример 3. Любое подмножество $\rho_V \subseteq V^2$, где V — множество всех действительных чисел, является графиком некоторого бинарного отношения ρ на множестве V. Так, отношения $x = a$, $y = b$, $x = y$, $x < y$ определяются графиками, показанными на рис. 11.

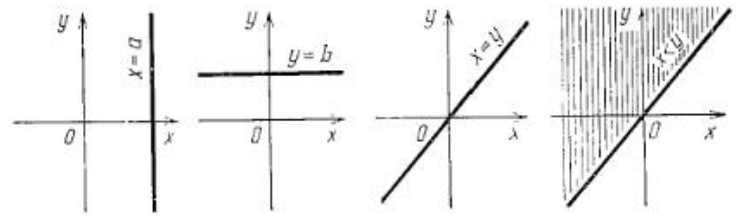


Рис. 11.

В частности, бинарным отношением на множестве V является любая функция $y = f(x)$, причем графиком этого отношения служит график функции f.

Пример 4. Пусть $F(\mathfrak{A})$ — множество всех слов конечной длины в алфавите $\mathfrak{A} = \{a_1, a_2, \ldots, a_n\}$. Отношение $<$ на множестве $F(\mathfrak{A})$ можно определить графиком $<_{F(\mathfrak{A})}$ таким, что $(s_1, s_2) \in <_{F(\mathfrak{A})}$ тогда и только тогда, когда $|s_1| < |s_2|$, где $|s|$ — длина слова $s \in F(\mathfrak{A})$. Отношение $=$ на множестве $F(\mathfrak{A})$ определяется

графиком $=_{F(\mathfrak{A})}$ так, что $(s_1, s_2) \in =_{F(\mathfrak{A})}$ тогда и только тогда, когда $|s_1| = |s_2|$.

Пример 5. Пусть $\mathfrak{M}$ — некоторое множество, $F(\mathfrak{A})$ — некоторое множество слов в алфавите $\mathfrak{A}$. Закодируем элементы из множества $\mathfrak{M}$ в алфавите $\mathfrak{A}$, сопоставив каждому $q \in \mathfrak{M}$ некоторое множество C_q слов в алфавите $\mathfrak{A}$ — кодов или имен элементов q. При этом $C_{q_1} \bigcap C_{q_2} = \varnothing$ для любых $q_1 \neq q_2$ ($q_1, q_2 \in \mathfrak{M}$). Отсюда следует, что каждый элемент $q \in \mathfrak{M}$ может иметь, вообще говоря, несколько кодов в алфавите $\mathfrak{A}$. В случае, когда каждому $q \in \mathfrak{M}$ соответствует лишь один код, кодирование взаимно-однозначно (см. § 1.3, пример 3). Кодирование элементов множества $\mathfrak{M}$ в алфавите $\mathfrak{A}$ можно задать, выделив в декартовом произведении $\mathfrak{M} \times F(\mathfrak{A})$ подмножество $\psi_{\mathfrak{M}, F(\mathfrak{A})} = \{(q, s) \mid q \in \mathfrak{M}, s \in C_q\}$ — график отношения ψ.

Пример 6. Пусть $A_1 = \{1, 2, \ldots, k\}$, $A_2 = \{2, 3, \ldots, k-1\}$, $\ldots$, $A_n = \{n, n+1, \ldots, k-n+1\}$. n-Отношения α^n, β^n, γ^n определим следующими графиками:

$$\alpha^n_{A_1, \ldots, A_n} = \begin{bmatrix} 1, & 2, & \ldots, & n \\ 2, & 3, & \ldots, & n+1 \\ \ldots & \ldots & \ldots & \ldots \\ k-2n+2, & k-2n+3, & \ldots, & k-n+1 \end{bmatrix},$$

$$\beta^n_{A_1, \ldots, A_n} = \begin{bmatrix} k, & k-1, & \ldots, & k-n+1 \\ k-1, & k-2, & \ldots, & k-n \\ \ldots & \ldots & \ldots & \ldots \\ 2n-1, & 2n-2, & \ldots, & n \end{bmatrix},$$

$$\gamma^n_{A_n, \ldots, A_1} = \begin{bmatrix} n, & n-1, & \ldots, & 1 \\ n+1 & n, & \ldots, & 2 \\ \ldots & \ldots & \ldots & \ldots \\ k-n+1, & k-n, & \ldots, & k-2n+2 \end{bmatrix}.$$

Ввиду справедливости включений $A_1 \supset A_2 \supset \cdots \supset A_n$ отношения α^n, β^n, γ^n можно рассматривать как n-отношения на множестве A_1.

Пример 7. Пусть $\mathfrak{M}$ — некоторое множество, $F(\mathfrak{A}_1)$, $F(\mathfrak{A}_2)$, $\ldots$, $F(\mathfrak{A}_n)$ — множества слов в алфавитах $\mathfrak{A}_1, \mathfrak{A}_2, \ldots, \mathfrak{A}_n$ соответственно, $\psi_{\mathfrak{M}, F(\mathfrak{A}_1)}$, $\psi_{\mathfrak{M}, F(\mathfrak{A}_2)}$, $\ldots$, $\psi_{\mathfrak{M}, F(\mathfrak{A}_n)}$ — графики бинарных отношений кодирования (см. пример 5). Отношение ψ^n на множествах $F(\mathfrak{A}_1)$, $F(\mathfrak{A}_2)$, $\ldots$, $F(\mathfrak{A}_n)$ определим следующим образом: $\psi(s_i^{(1)}, s_i^{(2)}, \ldots, s_i^{(n)})$ тогда и только тогда, когда найдется элемент $q \in \mathfrak{M}$, для которого $(q, s_i^{(1)}) \in \psi_{\mathfrak{M}, F(\mathfrak{A}_1)}$, $(q, s_i^{(2)}) \in \psi_{\mathfrak{M}, F(\mathfrak{A}_2)}$, $\ldots$, $(q, s_i^{(n)}) \in \psi_{\mathfrak{M}, F(\mathfrak{A}_n)}$.

Аппарат теории отношений применяется в программировании при создании формализмов для описания синтаксиса языков программирования, а также при решении задач, связанных с конструированием систем программирования, в частности при решении проблем перевода с одного языка на другой (см. гл. 5, 6). Отношение перевода π на языках L_1 и L_2 определяется так, что пара цепочек $s \in L_1$ и $s' \in L_2$ находится в отношении π, $s \pi s'$, если указанные цепочки синтаксически соответствуют друг другу, сохраняя один и тот же смысл. Применение ЭВМ в некоторой новой области связано с разработкой входного языка программирования высокого уровня, ориентированного на описание задач, относящихся к данной области, и системы программирования, реализующей отношение перевода с выбранного входного языка на язык конкретной ЭВМ. Различают системы программирования транслирующего и интерпретирующего типов. К трансляторам относятся системы, осуществляющие перевод с некоторого входного языка на язык ЭВМ с последующим выполнением переведенной программы на данной ЭВМ. Интерпретаторы совмещают процесс перевода программы с ее выполнением на ЭВМ. Подробному изучению формализмов, ориентированных на решение задач, связанных с конструированием систем программирования, посвящена гл. 6.

§ 2.2. Операции над отношениями

Изучение n-отношений на множествах $A_1, A_2, \ldots, A_n$ в значительной степени связано с рассмотрением их графиков — подмножеств множества $A_1 \times A_2 \times \cdots \times A_n$. Так, на множество n-отношений легко распространяются известные теоретико-множественные операции (см. § 1.2). В частности, можно говорить о включении n-отношений $\rho^n \subseteq \sigma^n$, если $\rho^n_{A_1, A_2, \ldots, A_n} \subseteq \sigma^n_{A_1, \ldots, A_n}$, а также о пересечении, объединении и дополнении n-отношений. Отношение $\theta^n = \rho^n \bigcap \sigma^n$, график которого имеет вид $\theta^n_{A_1, \ldots, A_n} = \rho^n_{A_1, \ldots, A_n} \bigcap \sigma^n_{A_1, \ldots, A_n}$, является пересечением отношений ρ^n и σ^n. Под объединением отношений ρ^n и σ^n понимается отношение $\theta^n = \rho^n \bigcup \sigma^n$ с графиком $\theta^n_{A_1, \ldots, A_n} = \rho^n_{A_1, \ldots, A_n} \bigcup \sigma^n_{A_1, \ldots, A_n}$. Отношение $\overline{\rho^n}$ называется дополнением ρ^n, если $\tilde{\alpha} \in \overline{\rho^n}_{A_1, \ldots, A_n}$ тогда и только тогда, когда $\tilde{\alpha} \overline{\in} \rho^n_{A_1, \ldots, A_n}$. Например, для бинарных отношений $<$, $=$, $\leqslant$, $\geqslant$, $>$ (см. § 2.1, пример 2) выполняются следующие равенства:

$$\leqslant_N = <_N \bigcup =_N, \quad =_N = \leqslant_N \bigcap \geqslant_N, \quad <_N = \overline{\geqslant}_N,$$

т. е. отношение $\leqslant$ является объединением отношений $<$ и $=$, отношение $=$ — пересечением отношений $\leqslant$ и $\geqslant$, отношение $<$ — дополнением отношения $\geqslant$. В то же время отношение $< \bigcap =$ пусто, а отношение $< \bigcup \geqslant$ универсально.

Пусть ρ^n — отношение на множествах $A_1, A_2, \ldots, A_n$, а σ^m — отношение на множествах $A_{n+1}, \ldots, A_{n+m}$. Под декартовым

произведением отношений ρ^n и σ^m понимается отношение $\theta^{n+m} = \rho^n \times \sigma^m$, график которого имеет вид $(\theta^{n+m}_{A_1, \ldots, A_{n+m}} = \rho^n_{A_1, \ldots, A_n} \times \sigma^n_{A_{n+1}, \ldots, A_{n+m}}$, где $\rho^n_{A_1, \ldots, A_n}$ и $\sigma^n_{A_{n+1}, \ldots, A_{n+m}}$ — графики отношений ρ^n и σ^m соответственно.

Пример 1. Пусть $A_1 = \{0, 1, 2\}$, $A_2 = \{a, b\}$, $A_3 = \{b, c, d\}$, α^2, β^3 — отношения с графиками

$$\alpha^2_{A_1, A_2} = \begin{bmatrix} 0, & a \\ 1, & b \end{bmatrix}, \quad \beta^3_{A_1, A_2, A_3} = \begin{bmatrix} 0, & a, & b \\ 1, & b, & c \\ 1, & b, & d \end{bmatrix}.$$

Тогда отношение $\gamma^5 = \alpha^2 \times \beta^3$ определяется графиком

$$\gamma^5_{A_1, A_2, A_3, A_4, A_5} = \begin{bmatrix} 0, & a \\ 1, & b \end{bmatrix} \times \begin{bmatrix} 0, & a, & b \\ 1, & b, & c \\ 1, & b, & d \end{bmatrix} = \begin{bmatrix} 0, & a, & 0, & a, & b \\ 0, & a, & 1, & b, & c \\ 0, & a, & 1, & b, & d \\ 1, & b, & 0, & a, & b \\ 1, & b, & 1, & b, & c \\ 1. & b, & 1, & b, & d \end{bmatrix}.$$

Кроме теоретико-множественных операций в теории n-отношений важное значение имеют операции перестановки и отождествления координат (столбцов), приписывания фиктивной координаты, а также свертка де Моргана, существенно обогащающие ее по сравнению с обычной теорией множеств.

Пусть ρ^n — отношение на множествах $A_1, A_2, \ldots, A_n$, $\tilde{k} = (1, 2, \ldots, n)$ — набор координат (номеров) столбцов в матрице графика $\rho^n_{A_1, \ldots, A_n}$, $\tilde{k}' = (i_1, i_2, \ldots, i_n)$ — набор, полученный из $\tilde{k}$ в результате некоторой перестановки элементов. Операция $\tau_{i\tilde{k}}(\rho^n)$ **перестановки координат** порождает n-отношение γ^n на множествах $A_{i_1}, A_{i_2}, \ldots, A_{i_n}$, график которого $\gamma^n_{A_{i_1}, \ldots, A_{i_n}}$ получается из графика $\rho^n_{A_1, \ldots, A_n}$ перестановкой его столбцов в соответствии с набором $\tilde{k}'$. Так, для отношений α^n и γ^n (см. § 2.1, пример 6) $\eta_{\tilde{k}'}(\alpha^n) = \gamma^n$, где $\tilde{k}' = (n, n-1, \ldots, 1)$. Рассмотрим наборы $\tilde{k}_0 = (2, 3, \ldots, n, 1)$ и $\tilde{k}_1 = (2, 1, 3, \ldots, n)$. Перестановка $\tau_{i\tilde{k}_0}(\rho^n) \cdot (\tau_{i\tilde{k}_1}(\rho^n))$ называется **циклом (транспозицией)** и обозначается через $\xi(\rho^n)$ ($\tau(\rho^n)$). С помощью цикла и транспозиции можно выразить любую перестановку $\tau_i(\rho^n)$ [66].

Пусть $\bar{k} = (n, n-1, \ldots, 2, 1)$. Тогда отношение $(\rho^n)^{-1} = \tau_{i\bar{k}}(\rho_n)$ над множествами $A_n, A_{n-1}, \ldots, A_2, A_1$, полученное из отношения ρ^n в результате перестановки, соответствующей набору $\bar{k}$, называется **обратным**. Очевидно, $\tilde{\alpha} = (a_i^{(n)}, a_i^{(n-1)}, \ldots, a_i^{(2)}, a_i^{(1)}) \in (\rho^n)^{-1}_{A_n, \ldots, A_1}$ тогда и только тогда, когда $\tilde{\alpha}' = (a_i^{(1)}, a_i^{(2)}, \ldots, a_i^{(n)}) \in \rho^n_{A_1, \ldots, A_n}$. Например, для бинарных отношений $<, >, \leqslant, \geqslant$ (см. § 2.1, пример 2) справедливы соотношения $<^{-1} = >$ и, наоборот, $>^{-1} = <$; точно так же $\leqslant^{-1} = \geqslant$ и $\geqslant^{-1} = \leqslant$. Для обращения n-отношений выполняются следующие равенства:

$$((\rho^n)^{-1})^{-1} = \rho^n; \tag{2.1}$$

если $\rho^n \subseteq \sigma^n$, то

$$(\rho^n)^{-1} \subseteq (\sigma^n)^{-1}, \tag{2.2}$$

$$(\bigcap_i \rho_i^n)^{-1} = \bigcap_i (\rho_i^n)^{-1}, \tag{2.3}$$

$$(\bigcup_i \rho_i^n)^{-1} = \bigcup_i (\rho_i^n)^{-1}, \tag{2.4}$$

$$\overline{(\rho^n)^{-1}} = \overline{(\rho^n)^{-1}}. \tag{2.5}$$

Докажем, например, справедливость равенства (2.5). Пусть $\tilde{\alpha} = (a_i^{(1)}, a_i^{(2)}, \ldots, a_i^{(n)})$ — произвольный вектор отношения $\overline{(\rho^n)^{-1}}$. Тогда $\overline{\rho^n}(\tilde{\alpha}')$, где $\tilde{\alpha}' = (a_i^{(n)}, a_i^{(n-1)}, \ldots, a_i^{(2)}, a_i^{(1)})$. Поэтому $\tilde{\alpha}' \in \overline{\rho^n}_{A_n, \ldots, A_1}$. Но тогда $\tilde{\alpha} \in \overline{(\rho^n)^{-1}_{A_1, \ldots, A_n}}$ и, следовательно, $\overline{(\rho^n)^{-1}}(\tilde{\alpha})$, т. е. $\overline{(\rho^n)^{-1}} \subseteq \overline{(\rho^n)^{-1}}$. Остается доказать справедливость обратного включения $\overline{(\rho^n)^{-1}} \subseteq \overline{(\rho^n)^{-1}}$. Действительно, пусть $\overline{(\rho^n)^{-1}}(\tilde{\alpha})$. Тогда $\tilde{\alpha} \in \overline{(\rho^n)^{-1}_{A_1, \ldots, A_n}}$ и $\tilde{\alpha}' \in (\rho^n)_{A_n, \ldots, A_1}$. Отсюда следует, что $\overline{\rho^n}(\tilde{\alpha}')$, а поэтому $\overline{(\rho^n)^{-1}}(\tilde{\alpha})$. Таким образом, $\overline{(\rho^n)^{-1}} \subseteq \overline{(\rho^n)^{-1}}$ и справедливость равенства (2.5) доказана.

Из равенств (2.3)—(2.5) следует, что, если $\mathfrak{A}(\rho_1^n, \rho_2^n, \ldots, \rho_k^n)$ — выражение, построенное из отношений $\rho_1^n, \ldots, \rho_k^n$ с помощью операций $\cap$, $\cup$, $-$, то

$$\mathfrak{A}^{-1}(\rho_1^n, \rho_2^n, \ldots, \rho_k^n) = \mathfrak{A}((\rho_1^n)^{-1}, (\rho_2^n)^{-1}, \ldots, (\rho_k^n)^{-1}).$$

Справедливо также равенство

$$(\rho^n \times \sigma^m)^{-1} = (\sigma^m)^{-1} \times (\rho^n)^{-1}. \tag{2.6}$$

Пусть, далее, ρ^n — отношение на множествах $A_1, \ldots, A_n$; $\{j_1, j_2, \ldots, j_l\}$ — некоторое подмножество множества номеров $\{1, 2, \ldots, n\}$. Операция **отождествления координат** $\delta^{(\rho^n)}_{j_1 \ldots j_l}$ порождает отношение $\theta^{n-l+1} = \delta^{(\rho^n)}_{j_1 \ldots j_l}$ на множествах $A_1, \ldots, A_{j_2-1}, A_{j_2+1}, \ldots, A_{j_3-1}, A_{j_3+1}, \ldots, A_{j_l-1}, A_{j_l+1}, \ldots, A_n$, график которого получается из графика $\rho^n_{A_1, \ldots, A_n}$ в результате выделения множества векторов $\{\tilde{\alpha} = (a_i^{(1)}, a_i^{(2)}, \ldots, a_i^{(n)}) \mid a_i^{(j_1)} = a_i^{(j_2)} = a_i^{(j_3)} = \ldots = a_i^{(j_l)}\} \subseteq \rho^n_{A_1, \ldots, A_n}$ с последующим вычеркиванием в каждом $\tilde{\alpha}$ элементов $a_i^{(j_2)}, a_i^{(j_3)}, \ldots, a_i^{(j_l)}$. Иными словами,

в отношении ρ^n выделяются все векторы, в которых совпадают элементы, расположенные в столбцах с координатами $j_1, \ldots, j_l$, с последующим исключением «столбцов-копий», имеющих координаты $j_2, j_3, \ldots, j_l$.

Пример 2. Для отношения γ^5 (см. пример 1) $\underset{2,4}{\delta}(\gamma^5) = \theta^4$, где отношение θ^4 определяется матрицей

$$\theta^4_{A_1, A_2, A_1, A_3} = \begin{bmatrix} 0, & a, & 0, & b \\ 1, & b, & 1, & c \\ 1, & b, & 1, & d \end{bmatrix}.$$

Для отношения θ^4 $\underset{1,3}{\delta}(\theta^4) = \xi^3$, где график отношения ξ^3 имеет вид

$$\xi^3_{A_1, A_2, A_3} = \begin{bmatrix} 0, & a, & b \\ 1, & b, & c \\ 1, & b, & d \end{bmatrix}.$$

Если $l = 2$, $j_1 = 1$, $j_2 = 2$, то отождествление $\underset{1,2}{\delta}(\rho^n)$ обозначается $\delta(\rho^n)$. Очевидно, с помощью $\delta(\rho^n)$ и перестановки координат можно произвести любое отождествление координат.

Отношение δ^n на множестве A называется **диагональю**, если его график имеет вид $\delta^n_A = (a, \ldots, a)$ для любого $a \in A$. Диагональю, например, является бинарное отношение $=$ на множестве N (см. § 2.1, пример 2), как, впрочем, и на любом другом числовом множестве. Если $\underset{1,\ldots,n}{\delta}(\rho^n)$ — отношение на множестве A, то $\underset{1,\ldots,n}{\delta}(\rho^n) \subseteq \delta^n$. Кроме того, для любого отношения $\rho^n \subseteq \delta^n$ справедливо $(\rho^n)^{-1} = \rho^n$.

Операция **приписывания фиктивной координаты** $\nabla(\rho^n)$ порождает отношение $\theta^{n+1} = \nabla(\rho^n)$ над множествами $A, A_1, A_2, \ldots, A_n$ такое, что при $\rho^n(a_i^{(1)}, a_i^{(2)}, \ldots, a_i^{(n)})$ $\theta^{n+1}(a, a_i^{(1)}, a_i^{(2)}, \ldots, a_i^{(n)})$ для любого $a \in A$. Операция приписывания отношению ρ^n фиктивной координаты по множеству A состоит в образовании отношения $\theta^{n+1} = \nabla(\rho^n)$ с графиком, который получается следующим образом: для каждого вектора $\tilde{a}_i = (a_i^{(1)}, a_i^{(2)}, \ldots, a_i^{(n)}) \in \rho^n_{A_1, \ldots, A_n}$ строятся векторы $(a, a_i^{(1)}, a_i^{(2)}, \ldots, a_i^{(n)}) \in \theta^{n+1}_{A, A_1, \ldots, A_n}$, полученные приписыванием к вектору $\tilde{a}_i$ слева поочередно всех элементов множества A. Например, для отношения α^3 на множестве $A = \{0, 1\}$ с графиком

$$\alpha^3_A = \begin{bmatrix} 1, & 0, & 0 \\ 0, & 1, & 0 \\ 0, & 0, & 1 \end{bmatrix}$$

в результате приписывания фиктивной (для множества A) координаты получаем

$$\alpha^4_A = \begin{bmatrix} 0, & 1, & 0, & 0 \\ 0, & 0, & 1, & 0 \\ 0, & 0, & 0, & 1 \\ 1, & 1, & 0, & 0 \\ 1, & 0, & 1, & 0 \\ 1, & 0, & 0, & 1 \end{bmatrix},$$

С помощью операции ∇ можно производить выравнивание арностей в отношениях.

Одной из наиболее важных операций над отношениями является **свертка де Моргана.** Пусть ρ^n — отношение на множествах $A_1, A_2, \ldots, A_{n-1}, A$, а σ^m — отношение на множествах $A, A_n, A_{n+1}, \ldots, A_{n+m-2}$. Тогда операция свертки отношений ρ^n и σ^m порождает отношение $\theta^{n+m-2} = \rho^n \divideontimes \sigma^m$ над множествами $A_1, A_2, \ldots, A_{n+m-2}$ такое, что $\theta^{n+m-2}(a_i^{(1)}, a_i^{(2)}, \ldots, a_i^{(n+m-2)})$ тогда и только тогда, когда найдется $a \in A$, для которого $\rho^n(a_i^{(1)}, a_i^{(2)}, \ldots, a_i^{(n-1)}, a)$ и $\sigma^m(a, a_i^{(n)}, a_i^{(n+1)}, \ldots, a_i^{(n+m-2)})$. Например, свертка отношений α^n и γ^n (см. § 2.1, пример 6) порождает отношение $\theta^{2n-2} = \alpha^n \divideontimes \gamma^n$ на множествах $A_1, \ldots, A_{n-2}, A_{n-1}, \ldots, A_1$, график которого имеет вид

$$\theta^{2n-2}_{A_1, \ldots, A_{n-1}, A_{n-1}, \ldots, A_1} = \begin{bmatrix} 1, & 2, & \ldots, & n-1, & n-1, & \ldots, & 2, & 1, \\ 2, & 3, & \ldots, & n, & n, & \ldots, & 3, & 2, \\ \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots \\ k-2n+2, & k-2n+3, & \ldots, & k-n, & k-n, & \ldots, & k-2n+3, & k-2n+2 \end{bmatrix}.$$

График свертки $\alpha \divideontimes \beta$ бинарных отношений α, β на множествах $A = \{a, b, c\}$ и $B = \{0, 1, 2\} = C$ с графиками

$$\alpha_{A,B} = \begin{bmatrix} a, & 0 \\ b, & 1 \\ c, & 2 \end{bmatrix}, \quad \beta_{B,C} = \begin{bmatrix} 0, & 1 \\ 0, & 2 \\ 1, & 2 \\ 2, & 2 \end{bmatrix}$$

имеет вид

$$(\alpha \divideontimes \beta) = \begin{bmatrix} a, & 1 \\ a, & 2 \\ b, & 2 \\ c, & 2 \end{bmatrix}.$$

Операция свертки ассоциативна,

$$(\rho^n \divideontimes \sigma^m) \divideontimes \theta^k = \rho^n \divideontimes (\sigma^m \divideontimes \theta^k), \qquad (2.7)$$

но не коммутативна. Так, в приведенном выше примере свертка

$\beta \nleftarrow \alpha$, очевидно, не определена. Кроме того, справедливо равенство

$$(\rho^n \nleftarrow \sigma^m)^{-1} = (\sigma^m)^{-1} \nleftarrow (\rho^n)^{-1}. \tag{2.8}$$

Пусть ρ — бинарное отношение на множествах A, B; σ — бинарное отношение на множествах B, C. Свертка бинарных отношений ρ и σ называется их **композицией или произведением.** Операция композиции бинарных отношений допускает и другие обобщения на n-арный случай (см. § 2.3).

Пусть L_1, L_2, L_3 — алгоритмические языки и τ — отношение перевода с L_1 на L_2, а τ' — отношение перевода с L_2 на L_3 (см. § 2.1). Тогда композиция $\tau'' = \tau \nleftarrow \tau'$ отношений τ и τ' также является отношением перевода с языка L_1 на L_3.

§ 2.3. Функциональные отношения. Отображения

Пусть ρ^n — отношение на множествах $A_1, A_2, \ldots, A_n$ с графиком

$$\rho^n_{A_1, \ldots, A_n} = \begin{bmatrix} a^{(1)}_{i_1}, & a^{(2)}_{i_1}, & \ldots, & a^{(n)}_{i_1} \\ a^{(1)}_{i_2}, & a^{(2)}_{i_2}, & \ldots, & a^{(n)}_{i_2} \\ \ldots & \ldots & \ldots & \ldots \\ a^{(1)}_{i_r}, & a^{(2)}_{i_r}, & \ldots, & a^{(n)}_{i_r} \end{bmatrix}.$$

Проекцией отношения ρ^n на множество A_j (при любом $j = 1, 2, \ldots, n$) называется множество $\mathrm{Пр}^{(j)}(\rho^n) = \{a^{(j)}_{i_1}, a^{(j)}_{i_2}, \ldots, a^{(j)}_{i_r}\} \subseteq A_j$ всех элементов j-го столбца матрицы $\rho^n_{A_1, \ldots, A_n}$. Иными словами, проекция отношения ρ^n на множество A_j — это совокупность j-х компонент всех векторов отношения ρ^n.

Сечением $S^j_{a^{(j)}_i, \ldots, a^{(j-1)}_i, a^{(j+1)}_i, \ldots, a^{(n)}_i}(\rho^n)$ **отношения** ρ^n по элементам $a^{(1)}_i, \ldots, a^{(j-1)}_i, a^{(j+1)}_i, \ldots, a^{(n)}_i$ называется множество всех элементов $a^{(j)}_{i_q} \in A_j$, для которых $\rho^n(a^{(1)}_i, a^{(2)}_i, \ldots, a^{(j-1)}_i, a^{(j)}_{i_q}, a^{(j+1)}_i, \ldots, a^{(n)}_i)$, т. е. $S^j_{a^{(1)}_i, \ldots, a^{(j-1)}_i, a^{(j+1)}_i, \ldots, a^{(n)}_i}(\rho^n) = \{a^{(j)}_{i_q} \mid \rho^n(a^{(1)}_i, \ldots, a^{(j-1)}_i, a^{(j)}_{i_q}, a^{(j+1)}_i, \ldots, a^{(n)}_i)\}$ $(j = 1, 2, \ldots, n)$. Рассмотрим множество $A_j/\rho^n = \{S^j_{a^{(1)}_i, \ldots, a^{(j-1)}_i, a^{(j+1)}_i, \ldots, a^{(n)}_i}(\rho^n)\}$ сечений отношения ρ^n по всевозможным совокупностям $(a^{(1)}_i, \ldots, a^{(j-1)}_i, a^{(j+1)}_i, \ldots, a^n_i) \in A_1 \times A_2 \times \cdots \times A_{j-1} \times A_{j+1} \times \cdots \times A_n$. Множество A_j/ρ^n называется **фактор-множеством** множества A_j по отношению ρ^n. Вместо одной последовательности $a^{(1)}_i, \ldots, a^{(j-1)}_i, a^{(j+1)}_i, \ldots, a^{(n)}_i$ можно рассматривать их совокупность $\tilde{X} = \{a^1_{i_t}, \ldots, a^{(j-1)}_{i_t}, a^{(j+1)}_{i_t}, \ldots, a^{(n)}_{i_t} \mid t = 1, 2, \ldots, k\}$.

Тогда сечение $S'_{\tilde{X}}(\rho^n)$ отношения ρ^n по совокупности $\tilde{X}$ является объединением сечений отношения ρ^n по всем последовательностям, входящим в $\tilde{X}$:

$$S^j_{\tilde{X}}(\rho^n) = \bigcup_{t=1}^{k} S^j_{a^{(1)}_{i_t}, \ldots, a^{(j-1)}_{i_t}, a^{(j+1)}_{i_t}, \ldots, a^{(n)}_{i_t}}(\rho^n).$$

Пример. Пусть отношение α^3 на множествах $A = \{a_1, a_2\}$, $B = \{b_1, b_2, b_3\}$ и $C = \{0, 1\}$ определяется графиком

$$\alpha^3 = \begin{bmatrix} a_1, & b_1, & 0 \\ a_1, & b_2, & 0 \\ a_2, & b_1, & 1 \\ a_2, & b_2, & 1 \\ a_2, & b_2, & 0 \end{bmatrix},$$

$$\mathrm{Пр}^1(\alpha^3) = A, \quad \mathrm{Пр}^2(\alpha^3) = \{b_1, b_2\}, \quad \mathrm{Пр}^3(\alpha^3) = C,$$
$$S^1_{b_2, 0}(\alpha^3) = \{a_1, a_2\}, \quad S^2_{a_1, 0} = \{b_1, b_2\}.$$

Фактор-множество C/α^3 представимо в виде

$$C/\alpha^3 = \{S^3_{a_1, b_1}, S^3_{a_2, b_1}, S^3_{a_1, b_2}, S^3_{a_2, b_2}, S^3_{a_1, b_3}, S^3_{a_2, b_3}\}$$

($S^3_{a_i, b_j}$ — соответствующие сечения отношения α^3, $i = 1, 2$; $j = 1, 2, 3$), где $S^3_{a_1, b_1} = \{0\}$, $S^3_{a_2, b_1} = \{1\}$, $S^3_{a_1, b_2} = \{0\}$, $S^3_{a_2, b_2} = C$, $S^3_{a_1, b_3} = S^3_{a_2, b_3} = \varnothing$ для множества $\tilde{X} = \{(a_1, b_1), (a_2, b_1)\}$, $S^3_{\tilde{X}}(\alpha^3) = S^3_{a_1, b_1} \cup S^3_{a_2, b_1} = C$.

Отношение φ^n на множествах $A_1, \ldots, A_n$ называется **функциональным,** если для каждой последовательности элементов $(a^{(1)}_i, a^{(2)}_i, \ldots, a^{(n-1)}_i) \in A_1 \times A_2 \times \cdots \times A_{n-1}$ сечение $S^n_{a^{(1)}_i, \ldots, a^{(n-1)}_i}(\varphi^n)$ содержит не более одного элемента $a^{(n)}_i \in A_n$. Если сечение $S^n_{a^{(1)}_i, \ldots, a^{(n-1)}_i}(\varphi^n)$ содержит точно один элемент $a^{(n)}_i \in A_n$ для любой последовательности $(a^{(1)}_i, \ldots, a^{(n-1)}_i) \in A_1 \times \cdots \times A_{n-1}$, то φ^n называется **всюду определенным** функциональным отношением. Функциональными, например, являются отношения α^3 и β^3 на множествах $A = \{a_1, a_2\}$, $B = \{b_1, b_2\}$ и $C = \{0, 1\}$ с графиками

$$\alpha^3_{A, B, C} = \begin{bmatrix} a_1, & b_1, & 0 \\ a_1, & b_2, & 0 \\ a_2, & b_1, & 1 \end{bmatrix}, \tag{2.9}$$

$$\beta^3_{A, B, C} = \begin{bmatrix} a_1, & b_1, & 1 \\ a_1, & b_2, & 0 \\ a_2, & b_1, & 0 \\ a_2, & b_2, & 1 \end{bmatrix}, \tag{2.10}$$

причем отношение β^3 всюду определено.

Рассмотрим функциональное отношение φ^n на множествах $A_1, \ldots, A_n$. Функция $F_{\varphi^n}(x_1, x_2, \ldots, x_{n-1})$ называется **связанной** с отношением φ^n, если каждая ее переменная x_i принимает значения из множества A_i $(i = 1, \ldots, n-1)$, причем $F_{\varphi^n}(a_i^{(1)}, \ldots, a_i^{(n-1)}) = S^n_{a_i^{(1)}, \ldots, a_i^{(n-1)}} \varphi^n$ для любого набора $(a_i^{(1)}, \ldots, a_i^{(n-1)}) \in A_1 \times \cdots \times A_{n-1}$. В приведенном выше примере с функциональным отношением α^3 (2.9) связана частичная функция $F_{\alpha^3}(x, y)$, определенная на множествах A, B и принимающая значения из множества C, так что $F_{\alpha^3}(a_1, b_1) = F_{\alpha^3}(a_1, b_2) = 0$, $F_{\alpha^3}(a_2, b_1) = 1$. В то же время с функциональным отношением β^3 (2.10) связана функция $F_{\beta^3}(x, y)$, заданная на этих же множествах, причем $F_{\beta^3}(a_1, b_1) = F_{\beta^3}(a_2, b_2) = 1$, $F_{\beta^3}(a_1, b_2) = F_{\beta^3}(a_2, b_1) = 0$.

Пусть φ_1^m — отношение на множествах $A_1, \ldots, A_{m-1}, B_1$; φ_2^m — на множествах $A_1, \ldots, A_{m-1}, B_2$; $\ldots$; φ_{n-1}^m — на множествах $A_1, \ldots, A_{m-1}, B_{n-1}$; σ^n — на множествах $B_1, \ldots, B_{n-1}, A_m$. **Суперпозицией отношений** $\varphi_1^m, \varphi_2^m, \ldots, \varphi_{n-1}^m, \sigma^n$ называется m-отношение $\theta^m = \sigma^n(\varphi_1^m, \ldots, \varphi_{n-1}^m)$ на множествах $A_1, \ldots, A_m$ такое, что $\theta^m(a_i^{(1)}, \ldots, a_i^{(m)})$ тогда и только тогда, когда найдутся элементы $b_j^{(1)} \in B_1$, $b_j^{(2)} \in B_2$, $\ldots$, $b_j^{(n-1)} \in B_{n-1}$, для которых $\varphi_s^m(a_i^{(1)}, a_i^{(2)}, \ldots, a_i^{m-1}, b_j^s)$ (при любом $s = 1, 2, \ldots, n-1$), причем $\sigma^n(b_j^{(1)}, b_j^{(2)}, \ldots, b_j^{(n-1)}, a_i^m)$ [111]. Частным случаем суперпозиции отношений является свертка де Моргана (см. § 2.2). Если отношения $\varphi_1^m, \varphi_2^m, \ldots, \varphi_{n-1}^m, \varphi^n$ функциональны и с ним связаны функции $F_{\varphi_1^m}, F_{\varphi_1^m}, \ldots, F_{\varphi_{n-1}^m}, F_{\varphi^n}$ соответственно, то суперпозиция $\varphi^n(\varphi_1^m, \varphi_2^m, \ldots, \varphi_{n-1}^m)$ также является функциональным отношением, причем с ней связана суперпозиция функций $F_{\varphi^n}(F_{\varphi_1^m}, F_{\varphi_2^m}, \ldots, F_{\varphi_{n-1}^m})$. Заметим, что суперпозиция всюду определенных функциональных отношений также является всюду определенным функциональным отношением.

Проиллюстрируем введенные понятия на одном важном классе булевых функций. Функция $f(x_1, \ldots, x_n)$ называется **булевой**, если она и все ее переменные $x_1, \ldots, x_n$ принимают значения из двухэлементного множества $E_2 = \{0, 1\}$. Каждая n-местная булева функция связана с всюду определенным $(n+1)$-арным отношением на множестве E_2. Например, дизъюнкция $x \vee y$ (см. § 1.2) связана с тернарным отношением ρ^3, график которого имеет вид

$$\rho^3_{E_2} = \begin{bmatrix} 0, & 0, & 0 \\ 0, & 1, & 1 \\ 1, & 0, & 1 \\ 1, & 1, & 1 \end{bmatrix}. \qquad (2.11)$$

Рассмотрим суперпозицию $\rho^3(\alpha, \beta)$ бинарных отношений α, β с графиками

$$\alpha_{E_2} = \begin{bmatrix} 0, & 0 \\ 1, & 1 \end{bmatrix}, \quad \beta_{E_2} = \begin{bmatrix} 0, & 1 \\ 1, & 0 \end{bmatrix}$$

и отношения ρ^3 с графиком (2.11). Результатом такой суперпозиции является бинарное отношение $\theta = \rho^3(\alpha, \beta)$ с графиком

$$\theta_{E_2} = \begin{bmatrix} 0, & 1 \\ 1, & 1 \end{bmatrix}.$$

Отношения α, β, ρ^3 функциональны и всюду определены; с ними связаны соответственно функции $f_\alpha(x) = x$, $f_\beta(x) = \bar{x}$, $f_{\rho^3}(x, y) = x \vee y$. В то же время суперпозиция $\theta = \rho^3(\alpha, \beta)$ также является всюду определенным функциональным отношением, с которым связана функция $f_\theta(x) = x \vee \bar{x} = 1$.

Пусть φ — бинарное отношение на множествах A, B. Отношение φ называется **отображением множества A в B**, если φ функционально и всюду определено; иными словами, для любого $a \in A$ сечение $S_a(\varphi)$ не пусто и содержит точно один элемент $s_a(\varphi) = b \in B$. Элемент $b = (a)\varphi$ называется **образом** элемента a при отображении φ, а элемент a — **прообразом** элемента b при данном отображении. Множество всех прообразов элемента b, т. е. совокупность всех $a \in A$ таких, что $(a)\varphi = b$, называется **полным прообразом** элемента b в A при отображении φ.

Пусть φ — отображение множества A в B, а ψ — отображение множества B в C. Композиция $\varphi\psi$ является отображением множества A в C, при котором $(a)\varphi\psi = ((a)\varphi)\psi$ для любого $a \in A$.

Отображение φ множества A в B называется **отображением A на B**, если каждый элемент в $b \in B$ имеет в A хотя бы один прообраз. Например, отображением множества $A = \{a, b, c, d\}$ на множество $B = \{1, 2, 3\}$ является отношение α с графиком

$$\alpha_{A,B} = \begin{bmatrix} a, & 1 \\ b, & 1 \\ c, & 2 \\ d, & 3 \end{bmatrix}.$$

В то же время бинарное отношение β на множестве B с графиком

$$\beta_B = \begin{bmatrix} 1, & 1 \\ 2, & 1 \\ 3, & 2 \end{bmatrix}$$

является отображением множества B в себя. Отображением мно-

жества A в множество B является композиция $\alpha\beta$ с графиком

$$(\alpha\beta)_{A,\,B} = \begin{bmatrix} a, & 1 \\ b, & 1 \\ c, & 1 \\ d, & 2 \end{bmatrix}.$$

Отображение φ множества A на множество B называется **взаимно-однозначным** (взаимно-однозначным соответствием, см. § 1.3), если обратное отношение φ^{-1} является отображением множества B на множество A. В частности, взаимно-однозначным отображением множества A на A является бинарное диагональное отношение δ_A (см. § 2.2), которое часто называют **тождественным отображением** множества A в себя.

Теорема 2.1. *Отображение φ множества A на множество B взаимно-однозначно (в этом случае множества A и B эквивалентны) тогда и только тогда, когда*

$$\varphi\varphi^{-1} = \delta_A, \quad \varphi^{-1}\varphi = \delta_B, \tag{2.12}$$

где δ_A, δ_B — тождественные отображения множеств A и B соответственно.

Действительно, пусть φ — взаимно-однозначное отображение множества A на B. Тогда по определению взаимно-однозначного отображения отношение φ^{-1} также является отображением множества B на A. Отсюда следует, что $(a)\,\varphi\varphi^{-1} = ((a)\,\varphi)\,\varphi^{-1} = a$; $(b)\,\varphi^{-1}\varphi = ((b)\,\varphi^{-1})\,\varphi = b$ для любых $a \in A$ и $b \in B$. Таким образом, $\varphi\varphi^{-1} = \delta_A$, $\varphi^{-1}\varphi = \delta_B$ и необходимость условий (2.12) доказана.

Пусть выполняются условия (2.12). Докажем, что отношение φ является взаимно-однозначным отображением множества A на B. Для этого надо показать, что φ служит отображением множества A на B, а φ^{-1} — отображением множества B на A. Пусть для произвольного элемента $a \in A$ найдутся элементы b_1, $b_2 \in B$, для которых $a\varphi b_1$ и $a\varphi b_2$. Тогда $b_1\varphi^{-1}a$ и, вследствие $a\varphi b_2$, для композиции $\varphi^{-1}\varphi$ выполняется отношение $b_1\,(\varphi^{-1}\varphi)\,b_2$. Но в силу (2.12) $\varphi^{-1}\varphi = \delta_B$, откуда следует, что $b_1 = b_2$. Таким образом, φ является отображением множества A на B. Аналогично можно показать, что отношение φ^{-1} есть отображение множества B на A и, следовательно, φ является взаимно-однозначным отображением множества A на B. Тем самым доказана необходимость и достаточность условий (2.12).

Заметим, что, если $A = B$, отношение φ является взаимно-однозначным отображением множества A на себя тогда и только тогда, когда $\varphi\varphi^{-1} = \varphi^{-1}\varphi = \delta_A$.

Отображения и связанные с ними понятия широко используются в кибернетике. Так, в теории алгоритмов и теории автоматов важную роль играют алфавитные и автоматные отображения [24], устанавливающие соответствия между цепочками входного и выходного алфавитов. В программировании к основным понятиям адресного языка [128] относится **адресное отображение**, сопоставляющее адресам ячеек памяти ЭВМ хранимую в них информацию. С адресным отображением связана **штрих-операция** $'a = b$, где a — некоторый адрес, b — содержимое адреса a. Таким образом, штрих-операция — это унарная функция, областью определения которой является множество A всех адресов, а областью значений — множество B всех содержимых. Штрих-функция определяет некоторое отображение множества адресов A на множество содержимых B, которое называется **адресным**. При этом каждому адресу соответствует лишь одно содержимое. В то же время обратное отображение может быть неоднозначным, т. е. одно содержимое может находиться в нескольких адресах. В этом случае полным прообразом содержимого является множество всех адресов, в которых оно находится.

При абстрактных рассмотрениях на множество A и B не накладывается никаких ограничений, так что штрих-функция может быть определена на произвольном множестве аргументов A и иметь значения в произвольном множестве B. Однако в практических приложениях оба множества ограничены и являются множествами некоторых кодов. В частности, множества A и B могут пересекаться, причем результатом штрих-операции может быть снова адрес. В этом случае повторное применение штрих-операции приводит к понятию **адреса второго ранга**, или **фиксатора**. Так, если $'a_1 = a_2$ и $a_2 \in A$, т. е. a_2 в свою очередь является адресом, то существует такое b, что $'a_2 = b$. Для понятия адреса второго ранга используется следующая символическая запись: $^2a_1 = '('a_1) = {}'a_2 = b$.

Аналогично вводится понятие **адресов высшего ранга**. Например, запись $^4a = b$ означает, что есть адреса a_1, a_2, $a_3 \in A$ и $'a = a_1$; $'a_1 = a_2$; $'a_2 = a_3$; $'a_3 = b$, т. е. $^4a = {}^3a_1 = {}^2a_2 = {}'a_3 = b$.

Будем считать b адресом нулевого ранга величины b. Существование адресов высшего ранга зависит от заданного адресного отображения. Так, если $B_1 = A \cap B$ и множество адресов $A_1 \subset A$ является объединением всех прообразов для элементов $b \in B_1$ (т. е. $a \in A_1$ при условии, что $'a = b$ и $b \in B_1$), то A_1 является множеством адресов второго ранга.

Пусть $B_0 \subset B$ — множество содержимых адресов, принадлежащих B_1. Тогда существует соответствие между множествами A_1 и B_0, определяющее операцию **двойной штриховки**, которая обозначается $^2a = b$, где $a \in A_1$, $b \in B_0$.

Аналогично можно выделить и соответствия, определяющие адреса высших рангов. Все они однозначно зависят от исходного адресного отображения A на B.

Техника отображения используется также в формализмах, ориентированных на описание систем программирования при установлении соответствия между синтаксисом и семантикой языков программирования (см. § 6.4, 6.7).

§ 2.4. Отношения эквивалентности и порядка

Рассмотрим свойства бинарных отношений, определенных на некотором множестве A.

Бинарное отношение ρ на множестве A называется **рефлексивным**, если $a\rho a$ для любого $a \in A$. Иными словами, отношение ρ рефлексивно, если для его графика ρ_A выполняется включение $\delta_A \subseteq \rho_A$, где δ_A — диагональное отношение на множестве A. Пусть, например, $A_1 = \{a, b, c\}$. Тогда отношение α с графиком

$$\alpha_{A_1} = \begin{bmatrix} a, & a \\ b, & b \\ c, & c \\ a, & b \\ a, & c \\ b, & a \\ c, & a \end{bmatrix} \tag{2.13}$$

рефлексивно, в то время как отношение β с графиком

$$\beta_{A_1} = \begin{bmatrix} a, & a \\ b, & b \\ a, & b \\ b, & c \\ a, & c \end{bmatrix} \tag{2.14}$$

не рефлексивно.

Отношение ρ на множестве A называется **транзитивным**, если из $a\rho b$ и $b\rho c$ следует $a\rho c$ для любых элементов $a, b, c \in A$. Иными словами, отношение ρ транзитивно, если для композиции $\tau = \rho\rho$ выполняется включение $\tau_A \subseteq \rho_A$. Транзитивно, например, отношение β с графиком (2.14), так как композиция имеет график $\beta\beta = \beta$ и, следовательно, выполняется включение $(\beta\beta)_{A_1} \subseteq \beta_{A_1}$. В то же время отношение α с графиком (2.13), очевидно, не транзитивно, так как $(b, c) \overline{\in} \alpha_{A_1}$.

Отношение ρ на множестве A называется **симметричным**, если из $a\rho b$ следует $b\rho a$ для любых $a, b \in A$. Иными словами, отношение ρ симметрично, если $\rho_A = \rho_A^{-1}$, где ρ^{-1} — отношение, обратное ρ. Симметрично, например, отношение α с графиком (2.13), в то время как отношение β с графиком (2.14) не симметрично.

Отношение ρ на множестве A называется **антисимметричным**, если из $a\rho b$ и $b\rho a$ следует $a = b$ для любых $a, b \in A$. Иными словами, отношение ρ антисимметрично, если $\rho_A \cap \rho_A^{-1} \subseteq \delta_A$. Антисимметрично, например, отношение β с графиком (2.14), тогда как отношение α с графиком (2.13), очевидно, не антисимметрично.

Если бинарное отношение ρ удовлетворяет любому из перечисленных четырех свойств, то обратное отношение ρ^{-1} обладает этим же свойством. Действительно, если $\delta_A \subseteq \rho_A$, то $\delta_A = \delta_A^{-1} \subseteq \rho_A^{-1}$. Далее, из $(\rho\rho)_A \subseteq \rho_A$ следует $(\rho^{-1}\rho^{-1})_A = (\rho\rho)_A^{-1} \subseteq \rho_A^{-1}$. Таким образом, операция обращения (см. § 2.2) сохраняет свойства рефлексивности и транзитивности, а также симметричности и антисимметричности, что непосредственно следует из определения этих свойств.

Бинарное отношение ρ на некотором непустом множестве A, удовлетворяющее свойствам рефлексивности, транзитивности и симметричности, называется **отношением эквивалентности**. Очевидно, если ρ — отношение эквивалентности на множестве A, то обратное отношение ρ^{-1} также является отношением эквивалентности на данном множестве.

Пример 1. Отношением эквивалентности является отношение равенства на некотором числовом множестве и вообще диагональное отношение δ_A на любом непустом множестве A.

Пример 2. Пусть $F(\mathfrak{A})$ — множество всех слов конечной длины в алфавите $\mathfrak{A} = \{a_1, a_2, \ldots, a_n\}$. Отношением эквивалентности на множестве $F(\mathfrak{A})$ является отношение $\overline{\underset{(L)}{=}}$ такое, что $s_1 \overline{\underset{(L)}{=}} s_2$ тогда и только тогда, когда $|s_1| = |s_2|$ (где $|s|$ — длина слова $s \in F(\mathfrak{A})$) для любых $s_1, s_2 \in F(\mathfrak{A})$.

Пример 3. Отношением эквивалентности на множестве $F(\mathfrak{A})$ является отношение $\underset{(\widetilde{n})}{}$, такое, что для любой пары $s_i = a_{i_1}a_{i_2} \ldots a_{i_k}$, $s_j = a_{j_1}a_{j_2} \ldots a_{j_r}$ справедливо отношение $s_i \underset{(\widetilde{n})}{} s_j$ тогда и только тогда, когда в словах s_i и s_j совпадают начальные символы: $a_{i_1} \equiv a_{j_1}$.

Отношение эквивалентности на множестве A связано с разбиениями этого множества на попарно-непересекающиеся подмножества (см. § 1.2).

Пусть ρ — некоторое отношение эквивалентности на непустом множестве A. Рассмотрим фактор-множество $A/\rho = \{S_a(\rho) \mid a \in A\}$. Сечение $S_a(\rho)$ назовем смежным классом элемента a по отношению ρ. Покажем, что фактор-множество A/ρ по отношению эквивалентности ρ является разбиением множества A на смежные классы:

$$\bigcup_{a \in A} S_a(\rho) = A, \tag{2.15}$$

причем различные смежные классы не пересекаются. Действи-

тельно, отношение ρ рефлексивно, и поэтому $a \in S_a(\rho)$, откуда следует справедливость равенства (2.15).

Пусть, далее, $S_a(\rho)$, $S_b(\rho)$ — различные смежные классы. Допустим, что найдется элемент $c \in A$ такой, что $c \in (S_a(\rho) \bigcap S_b(\rho))$. Тогда из $c \in S_a(\rho)$ следует $a\rho c$. В то же время из $c \in S_b(\rho)$ следует $b\rho c$. Но отношение ρ симметрично и транзитивно, поэтому наряду с $a\rho c$ справедливо $c\rho b$, а значит, и $a\rho b$. Вследствие транзитивности ρ из $a\rho b$ следует $S_a(\rho) \supseteq S_b(\rho)$. В то же время из $b\rho a$ следует $S_b(\rho) \supseteq S_a(\rho)$. Таким образом, $S_a(\rho) = S_b(\rho)$ и различные смежные классы не пересекаются, т. е. множество A/ρ всех смежных классов образует разбиение множества A.

Итак, каждому отношению эквивалентности на множестве A соответствует разбиение этого множества на смежные классы по данному отношению эквивалентности. Нетрудно показать, что и, наоборот, каждому разбиению $R(A) = \{A_1, A_2, \ldots, A_k\}$ множества A соответствует отношение эквивалентности σ на A, смежные классы которого совпадают с классами данного разбиения. Иными словами, $a\sigma b$ тогда и только тогда, когда элементы a, b принадлежат одному и тому же классу разбиения a, $b \in A_i$ (где $i = 1, 2, \ldots, k$). Очевидно, определенное таким способом отношение σ рефлексивно, симметрично и транзитивно, т. е. является отношением эквивалентности.

Заметим, что если построить разбиение A/ρ по некоторому отношению эквивалентности ρ на множестве A, а затем рассмотреть отношение эквивалентности, соответствующее данному разбиению, то снова получится отношение ρ. И, наоборот, если по некоторому разбиению $R(A)$ последовательно осуществить переход к соответствующему отношению эквивалентности ρ, а затем к определенному им разбиению A/ρ, то $A/\rho = R(A)$.

Справедлива следующая теорема.

Теорема 2.2. *Каждому отношению эквивалентности на множестве A соответствует единственное разбиение $R(A)$ данного множества, и, наоборот, любому разбиению множества A однозначно соответствует некоторое отношение эквивалентности на A.*

Теоремой установлено взаимно-однозначное соответствие между отношениями эквивалентности на множестве A и разбиениями данного множества. Так, приведенным выше отношениям эквивалентности на множестве $F(\mathfrak{A})$ (см. примеры 2,3) соответствуют следующие разбиения данного множества:

эквивалентности $\underset{(L)}{=}$ — разбиение $F(\mathfrak{A})/\underset{(L)}{=} = \{S_{|s|\underset{(L)}{=}} \mid s \in F(\mathfrak{A})\}$, где $S_{|s|\underset{(L)}{=}}$ — множество всех слов, равных по длине слову $s \in F(\mathfrak{A})$;

эквивалентности $\underset{(\text{н})}{\sim}$ — разбиение $F(\mathfrak{A})/\underset{(\text{н})}{\sim} = \{S_{a_i(\underset{(\text{н})}{\sim})} \mid a_i \in \mathfrak{A}\}$, где $S_{a_i(\underset{(\text{н})}{\sim})}$ — множество всех слов с начальным символом $a_i \in \mathfrak{A}$

Связь отношений эквивалентности и разбиений множеств можно использовать при определении понятия кардинального числа

(см. § 1.5), если считать, что два множества эквивалентны тогда и только тогда, когда они равномощны. В этом случае каждому классу эквивалентности соответствует определенная мощность (кардинальное число), причем некоторому классу разбиения конечных множеств соответствует натуральное число — число элементов в множествах из данного класса.

Пусть ρ — отношение эквивалентности на множестве A; $A/\rho = \{S_a(\rho) \mid a \in A\}$ — фактор-множество множества A по данному отношению эквивалентности. Отображение множества A на фактор-множество A/ρ, сопоставляющее каждому элементу $a \in A$ смежный класс $S_a(\rho)$, которому принадлежит элемент a, называется **естественным** отображением множества A на фактор-множество A/ρ.

Рассмотрим отображение φ множества A на множество B. Покажем, что отображению φ соответствует вполне определенное отношение эквивалентности σ на множестве A. Положим, что для элементов $a, b \in A$ $a\sigma b$ тогда и только тогда, когда $(a)\varphi = (b)\varphi$. Сопоставив каждому элементу $a' \in B$ его полный прообраз при отображении φ, получим взаимно-однозначное отображение ψ множества B на фактор-множество A/σ, причем композиция $\varphi\psi$ совпадает с естественным отображением множества A на фактор-множество A/σ.

Отношения эквивалентности и связанные с ними классы разбиений используются, в частности, при переводе с одного входного языка на другой для выделения семантически равносильных программ (см. § 6.5).

Бинарное отношение ρ на некотором множестве A, удовлетворяющее свойствам рефлексивности, транзитивности и антисимметричности, называется **отношением частичного порядка.**

Пример 4. Отношениями частичного порядка являются отношение $\leqslant$, а также обратное отношение $\geqslant$, обычно определенные на некотором числовом множестве.

Пример 5. Отношением частичного порядка является также диагональное отношение δ_A на любом непустом множестве $A = \{a_1, a_2, \ldots, a_r\}$.

Пример 6. На множестве $F(\mathfrak{A})$ всех слов конечной длины в алфавите $\mathfrak{A}$ частичным порядком является отношение $\underset{F(\mathfrak{A})}{\leqslant}$ такое, что $s_1 \underset{F(\mathfrak{A})}{\leqslant} s_2$ тогда и только тогда, когда $|s_1| \leqslant |s_2|$, где $s_1, s_2 \in F(\mathfrak{A})$.

Пример 7. Отношением частичного порядка на множестве $F(\mathfrak{A})$ является отношение $\underset{F(\mathfrak{A})}{\subseteq}$ такое, что $s_1 \underset{F(\mathfrak{A})}{\subseteq} s_2$ тогда и только тогда, когда слово s_1 является подсловом слова s_2, т. е. $s_2 = s's_1s''$, где $s_1, s_2, s', s'' \in F(\mathfrak{A})$.

Пример 8. Пусть $\mathfrak{M}(A)$ — множество всех подмножеств некоторого множества A. Отношение включения $\subseteq$ и обратное отно-

шение $\supseteq$ являются отношениями частичного порядка на множестве $\mathfrak{M}(A)$. Например, отношение $\subseteq$ является отношением частичного порядка на множестве $\mathfrak{M}[F(\mathfrak{A})]$ всех языков в алфавите $\mathfrak{A}$.

Отношение порядка на некотором непустом множестве A символически записывается $\leqslant$ или $\underset{(A)}{\leqslant}$ (если множество A подразумевается): $a \underset{(A)}{\leqslant} b$ читается «элемент a предшествует элементу b» или «b следует за a $(a, b \in A)$». Равнозначными считаются записи $b \underset{(A)}{\geqslant} a$ и $b \geqslant a$ (если множество A подразумевается). Если для отношения частичного порядка $\leqslant$ на множестве A справедливо $a \leqslant b$, причем $a \neq b$, то записывают также $a < b$ или $b > a$.

Исходя из отношения $\leqslant$, на множестве A можно определить отношение $<$ такое, что $a < b$ тогда и только тогда, когда $a \leqslant b$, $a \neq b$. Отношение $<$ на множестве A называется отношением **строгого порядка**, оно удовлетворяет свойству транзитивности (а также сильной антисимметричности), состоящему в том, что $a \rho b$ и $b \rho a$ не могут одновременно выполняться ни для каких a, $b \in A$. Отношение $<$ на множестве A, в свою очередь, однозначно определяет связанное с ним отношение $\leqslant$ на данном множестве.

Часто используются также отношения **квазипорядка**, удовлетворяющие свойствам рефлексивности и транзитивности. Например, отношением квазипорядка является отношение $\tilde{\pi}$, определенное на множестве формальных языков, в частности языков программирования, так что языки L и L' находятся в отношении $\tilde{\pi}$, $L \tilde{\pi} L'$, тогда и только тогда, когда существует перевод с языка L на язык L' (см. § 2.1).

Множество A с определенным на нем отношением частичного порядка ρ называется **частично упорядоченным**. Так, частично упорядоченными являются множества, рассмотренные в примерах 4—8. Элементы a, $b \in A$ такие, что $a \rho b$ или $b \rho a$, называются **сравнимыми** элементами частично упорядоченного множества A. Заметим, что в частично упорядоченное множество могут входить и несравнимые элементы. Так, в примере 5 (при $r > 1$) любая пара различных элементов из множества A несравнима.

Частично упорядоченное множество, в котором каждая пара элементов сравнима, называется **линейно упорядоченным** множеством или **цепью**. Например, линейно упорядоченными являются множество N всех натуральных чисел, множество V всех действительных чисел с естественно определенным на этих множествах отношением $\leqslant$. Множества в примерах 5—8 частично (но не линейно) упорядочены.

Пусть $A' \subseteq A$ — подмножество множества A, $A' \subseteq A$; ρ — бинарное отношение, определенное на множестве A. Отношение ρ на множестве A естественно индуцирует бинарное отношение $\tilde{\rho}$ на множестве A': если $a, b \in A'$, то $a \tilde{\rho} b$ тогда и только тогда, когда $a \rho b$

в множестве A. Иными словами, $\tilde{\rho}_{A'} = \rho_A \cap (A' \times A')$. Очевидно, если ρ — отношение эквивалентности (частичного порядка) на множестве A, то $\tilde{\rho}$ — также отношение эквивалентности (соответственно частичного порядка) на множестве A'.

Рассмотрим множество $\mathfrak{M}(A^2)$ всех бинарных отношений, определенных на некотором множестве A.

Пусть $\leqslant$ — бинарное отношение на множестве $\mathfrak{M}(A^2)$ такое, что для отношений $\rho, \sigma \in \mathfrak{M}(A^2)$ $\rho \leqslant \sigma$ тогда и только тогда, когда из $a \rho b$ следует $a \sigma b$ $(a, b \in A)$. Иными словами, $\rho \leqslant \sigma$, если для графиков ρ_A, σ_A выполняется включение $\rho_A \subseteq \sigma_A$. Множество $\mathfrak{M}(A^2)$ частично упорядочено относительно определенного указанным способом отношения $\leqslant$.

Пусть $T(A^2) \subseteq \mathfrak{M}(A^2)$ — совокупность всех отношений эквивалентности на множестве A. Определенная выше частичная упорядоченность множества $\mathfrak{M}(A^2)$ индуцирует частичную упорядоченность множества $T(A^2)$, причем если характеризовать отношения эквивалентности соответствующими им разбиениями множества A, то $\rho \leqslant \sigma$ означает, что разбиение $A/\rho = \{S_a(\rho) \mid a \in A\}$ более дробно, чем разбиение $A/\sigma = \{S_a(\sigma) \mid a \in A\}$. Точнее, для каждого смежного класса $S_a(\sigma) \in A/\sigma$ существует разбиение $R(S_a(\sigma)) = \{S_{a'}(\rho) \mid a' \in S_a(\sigma)\} \subseteq A/\rho$. В этом случае разбиение A/ρ является подразбиением разбиения A/σ (см. § 1.2).

Частично упорядоченные множества A_1 и A_2 называются **изоморфными**, если существует взаимно-однозначное отображение φ множества A_1 на множество A_2 такое, что $a \underset{(A_1)}{\leqslant} b$ тогда и только тогда, когда $(a)\varphi \underset{(A_2)}{\leqslant} (b)\varphi$, где $a, b \in A_1$. Отображение φ называется **изоморфным отображением** множества A_1 на множество A_2.

Частично упорядоченные множества A_1, A_2 называются **инверсно изоморфными**, если одно из них изоморфно другому, взятому с обратной частичной упорядоченностью. Иными словами, существует взаимно-однозначное отображение φ множества A_1 на множество A_2 такое, что $a \leqslant b$ тогда и только тогда, когда $(a)\varphi \geqslant (b)\varphi$. Частично упорядоченное множество A изоморфно вкладывается в частично упорядоченное множество A', если A изоморфно подмножеству $B \subseteq A'$, частичный порядок в котором индуцирован частичным порядком множества A'.

Возвращаясь к примеру 8, заметим, что любое частично упорядоченное множество A изоморфно вкладывается в некоторое множество $\mathfrak{M}(B)$, причем в качестве последнего может быть выбрано множество $\mathfrak{M}(A)$ всех подмножеств A, частично упорядоченное по отношению включения [66]. Покажем, что множество A изоморфно подмножеству $U = \{U_a \mid a \in A\} \subseteq \mathfrak{M}(A)$, где $U_a \subseteq A$ — подмножество всех элементов a' таких, что $a' \leqslant a$. Действительно, отображение φ множества A на множество U такое, что $(a)\varphi =$

$= U_a$, взаимно-однозначно, так как из $U_a = U_b$ (где $U_a = (a)\varphi$, $U_b = (b)\varphi$, $a, b \in A$) следует $a \in U_b$ и $b \in U_a$. Это в свою очередь означает, что $a \leqslant b$ и $b \leqslant a$, т. е. $a = b$. В то же время из $a \leqslant b$ следует $U_a \subseteq U_b$, так как при любом $x \leqslant a$ $x \leqslant b$. Наоборот, из $U_a \subseteq U_b$ следует $a \in U_b$, и поэтому $a \leqslant b$. Таким образом, множество A изоморфно подмножеству $U \subseteq \mathfrak{M}(A)$, следовательно, A изоморфно вкладывается в множество $\mathfrak{M}(A)$.

§ 2.5. Вполне упорядоченные множества. Структуры

Пусть A — частично упорядоченное множество; B — некоторое непустое подмножество множества A, $B \subseteq A$. **Верхней гранью** $B \subseteq A$ в множестве A называется элемент $a \in A$ такой, что $b \leqslant a$ для любого $b \in B$. Элемент a — **наибольший** в множестве A, если a — верхняя грань самого множества A. Соответственно элемент $a \in A$ называется **нижней гранью** $B \subseteq A$ в множестве A, если $a \leqslant b$ для любого $b \in B$. Элемент a — **наименьший** в множестве A, если a — нижняя грань самого A.

Элемент $m \in A$ называется **максимальным** в множестве A, если не существует $b \in A$ такого, что $m < b$. Соответственно элемент $m \in A$ **минимальный** в множестве A, если не существует $b \in A$ такого, что $b < m$. Очевидно, наибольший элемент множества A является единственным максимальным элементом данного множества; точно так же наименьший элемент множества A — единственный минимальный элемент в данном множестве. Заметим, что множество A может не иметь наибольшего (наименьшего) элемента и в то же время иметь один или несколько максимальных (минимальных) элементов.

Пример 1. Множество $\mathfrak{M}(A)$ всех подмножеств некоторого непустого множества A (см. § 2.4, пример 8) имеет наименьший элемент — множество $\varnothing$ и наибольший элемент — само множество A.

Пример 2. Множество N всех натуральных чисел в его естественной упорядоченности имеет наименьший элемент (число 1) и не имеет наибольшего элемента.

Пример 3. Множество Z всех целых чисел с естественной упорядоченностью не имеет наименьшего, наибольшего, а также минимального и максимального элементов.

Пример 4. В непустом множестве $A = \{a_1, \ldots, a_r\}$ $(r > 1)$ с диагональным отношением порядка (см. § 2.4, пример 5) каждый элемент $a_i \in A$ является одновременно минимальным и максимальным в данном множестве. Наибольшего и наименьшего элементов в множестве A нет.

Пример 5. В множестве всех непустых цепочек из $F(\mathfrak{A})$ (см. § 2.4, пример 6) минимальной является каждая одноэлементная цепочка, в то же время наименьшего, наибольшего и максимального элементов в данном множестве нет.

Линейно упорядоченное множество A называется **вполне упорядоченным**, если каждое непустое подмножество $B \subseteq A$ имеет наименьший элемент относительно данной упорядоченности. Примером упорядоченного множества может служить множество N всех натуральных чисел в их естественной упорядоченности. В то же время множество Z всех целых чисел, также взятое в его естественном порядке, не является вполне упорядоченным, так как любое бесконечное подмножество отрицательных чисел $B \subseteq Z$, очевидно, не имеет наименьшего элемента.

Для каждого элемента $a \in A$ произвольного вполне упорядоченного множества A, который не является наибольшим в данном множестве, существует единственный элемент $b \in A$ такой, что $a \leqslant b$, причем из $a \leqslant c$ следует $b \leqslant c$ для любого $c \in A$. Элемент b в этом случае называется **непосредственно следующим** за элементом a, элемент a — **непосредственно предшествующим** элементу b.

Для частично упорядоченных множеств справедливы следующие утверждения.

Теорема Цермело. *Всякое множество можно вполне упорядочить.*

Теорема Цермело эквивалентна лемме Цорна, применяемой в различных областях общей алгебры.

Лемма Цорна. *Если в произвольном частично упорядоченном множестве A любая цепь имеет верхнюю (нижнюю) грань, то A имеет максимальный (минимальный) элемент*[1].

Утверждение U относительно элементов некоторого вполне упорядоченного множества A можно доказать, используя трансфинитную индукцию, которая является естественным обобщением известного метода математической индукции. Суть этого обобщения состоит в следующем.

Базис. Утверждение U выполняется для наименьшего элемента множества A.

Индукционный шаг. Если утверждение U выполняется для любого элемента $b < a$, то это утверждение справедливо также для элемента a.

Пусть выполняются условия базиса и индукционного шага. Тогда можно заключить, что утверждение U истинно для произвольного элемента $x \in A$. Допустим противное: пусть в множестве A найдутся элементы, для которых не выполняется утверждение U, и $B \subseteq A$ — совокупность всех таких элементов. Множество A вполне упорядоченно, и, следовательно, подмножество $B \subseteq A$ имеет наименьший элемент $a \in B$. Из справедливости базиса следует, что a не является наименьшим элементом в множестве A. Тогда утверждение

[1] Теорема Цермело и лемма Цорна равносильны аксиоме выбора, а также теореме Хаусдорфа (см., например, [66]).

U выполняется для любого элемента $b \in A$ такого, что $b < a$. Однако по индукционному шагу утверждение U должно выполняться и для элемента a, что противоречит предположению.

Пусть A — частично упорядоченное множество с отношением порядка $\leqslant$. Отношение $\leqslant^{-1} = \geqslant$ (обратное отношению $\leqslant$) также является отношением частичного порядка на множестве A (см. § 2.4). Элементы множества A по отношению $\geqslant$ образуют частично упорядоченное множество A^*, **двойственное** множеству A. Очевидно, $A^{**} = A$.

Пусть G — некоторое высказывание о частично упорядоченном множестве A. Если в G всюду заменить знак $\leqslant$ знаком $\geqslant$, то получится новое высказывание G^* о частично упорядоченном множестве A^*. Высказывания G^* и G называются **двойственными.** Например, понятия верхней грани, наибольшего и максимального элементов в множестве A двойственны соответственно понятиям нижней грани, наименьшего и минимального элементов в множестве A^*.

Теорема 2.3. *Пусть U — некоторое утверждение, доказанное для частично упорядоченного множества A. Тогда справедливо и двойственное утверждение U^* относительно частично упорядоченного множества A^*.*

Действительно, пусть в частично упорядоченном множестве A^* выполняются условия утверждения U^*. Тогда в двойственном частично упорядоченном множестве $A^{**} = A$ выполняются условия утверждения $U^{**} = U$, а значит, и заключение утверждения U. Следовательно, в множестве A^* выполняется заключение утверждения U^*.

Используя принцип двойственности, для обоснования истинности двойственного утверждения необходимо в доказанном утверждении заменить двойственными все высказывания и понятия, относящиеся к частичному порядку, не изменяя остальных.

Пусть $B \subseteq A$ — непустое подмножество частично упорядоченного множества A. **Верхним конусом** $B^\triangle$ множества B называется совокупность всех верхних граней данного множества, **точной верхней гранью** $B \subseteq A$ в множестве A — элемент $a \in B^\triangle$ такой, что $a \leqslant b$ для любого $b \in B^\triangle$. Двойственным образом определяются понятия **нижнего конуса** $B^\triangledown$ множества B (совокупность всех нижних граней $B \subseteq A$) и **точной нижней грани** множества B (элемент $a \in B^\triangledown$ такой, что $a \geqslant b$ для любого $b \in B^\triangledown$). Точные верхняя и нижняя грани множества $B \subseteq A$ в частично упорядоченном множестве A обозначаются символами $\sup_A B$ и $\inf_A B$ соответственно (или $\sup B$, $\inf B$, если множество A подразумевается).

Перечислим некоторые свойства $\sup B$ и $\inf B$ в частично упорядоченном множестве A.

1. Пусть $B = \{a, b\} \subseteq A$, где $a \leqslant b$. Тогда $\sup B = b$, $\inf B = a$.
2. Пусть B, C — непустые подмножества множества A, причем

$B \subseteq C$. Если существуют $\sup B$ и $\sup C$ ($\inf B$ и $\inf C$ соответственно), то $\sup B \leqslant \sup C$ ($\inf B \geqslant \inf C$).

3. Пусть $B = \bigcup_{\alpha \in I} B_\alpha$, где B_α — подмножества частично упорядоченного множества A. Если существуют $\sup B$ и $\sup B_\alpha$ ($\inf B$ и $\inf B_\alpha$) для любого α, то $\sup B = \sup \{\sup B_\alpha\}$ ($\inf B = \inf \{\inf B_\alpha\}$).

Действительно, пусть $a = \sup B$, $a_\alpha = \sup B_\alpha$ для любого α. Тогда $a \geqslant b_\alpha$ для любого $b_\alpha \in B_\alpha$ и, следовательно, $a \geqslant a_\alpha$ для любого α, т. е. $a \in \{a_\alpha\}^\triangle$. Пусть, далее, $a' \in \{a_\alpha\}^\triangle$. Тогда $a' \geqslant a_\alpha$ для любого α и, следовательно, $a' \geqslant b_\alpha$ для всех $b_\alpha \in B_\alpha$. Это означает, что $a' \geqslant b$ для всех $b \in B$. Таким образом, $a' \geqslant a$, т. е. $\sup B = a = \sup \{a_\alpha\} = \sup \{\sup B_\alpha\}$. Для точных нижних граней утверждение доказывается двойственным образом.

4. Пусть A' — непустое подмножество частично упорядоченного множества A. Как отмечалось (см. § 2.4), частичный порядок множества A индуцирует частичный порядок в множестве $A' \subseteq A$, поэтому множество A' также частично упорядоченно. Если $B \subseteq A'$ и существует $a = \sup_A B$, причем $a \in A'$, то $a = \sup_{A'} B$. Аналогично из существования $a = \inf_A B$ такого, что $a \in A'$, следует $a = \inf_{A'} B$.

Обратное утверждение, вообще говоря, неверно. Из существования $\sup_{A'} B$ не следует существование $\sup_A B$. Более того, точные верхние грани $\sup_A B$ и $\sup_{A'} B$ могут существовать и не совпадать. Например, пусть $A = \{a, b, c, d\}$, $A' = \{a, b, c\}$, причем на множестве A задано отношение частичного порядка с графиком

$$\leqslant_A = \begin{bmatrix} a, & a \\ b, & b \\ c, & c \\ d, & d \\ a, & c \\ b, & c \\ a, & d \\ b, & d \end{bmatrix}. \tag{2.16}$$

Тогда для подмножества $B = \{a, b\} \subseteq A' \subseteq A$ $\sup_{A'} B = c$, в то время как $\sup_A B$ не существует. Если множество A упорядочить по отношению частичного порядка $\prec$ с графиком $\prec_A \supset \leqslant_A$, причем $(d, c) \in \prec_A$, то $\sup_A B = d \neq \sup_{A'} B = c$.

Элементы $a, b \in A$ (при $a \leqslant b$) определяют в частично упорядоченном множестве A подмножество $[a; b]$ такое, что $x \in [a; b]$ тогда и только тогда, когда $a \leqslant x \leqslant b$. Подмножество $[a; b] \subseteq A$ называется **интервалом** частично упорядоченного множества A. Очевидно, каждый интервал содержит определяющие его элементы

$a, b \in [a; b]$ — концы этого интервала. Интервал, состоящий из своих концов $[a; b] = \{a, b\}$, называется **простым.**

Частично упорядоченное множество A может быть задано в виде **структурного графа,** в котором элементы данного множества изображены точками. Точка a соединяется с точкой b, если $[a; b]$ — простой интервал. При этом элемент b располагается выше элемента a $(a, b \in A)$. Так, в приведенном выше примере множеству A, частично упорядоченному по отношению $\leqslant$, соответствует структурный граф, изображенный на рис. 12. Структурный граф этого же множества, частично упорядоченного по отношению $\lhd$, показан на рис. 13.

Структурные графы используются при описании строения множества подалгебр в универсальных алгебрах (см. § 3.3, 3.4)

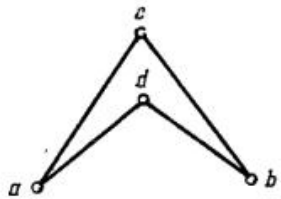


Рис. 12.

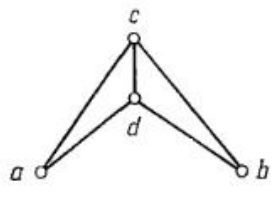


Рис. 13.

а также в алгебрах Поста и их модификациях, связанных с изучением преобразований на абстрактных регистрах (см. § 4.3, 4.4). Подобные конструкции (деревья вывода) применяются при описании синтаксической структуры правильных цепочек формальных языков (см. гл. 5,6).

Частично упорядоченное множество S называется **структурой** или **решеткой,** если для любой пары элементов $a, b \in S$ существуют точная нижняя грань $\inf_S\{a, b\}$ и точная верхняя грань $\sup_S\{a, b\}$ [6].

Пример 6. Множество N всех натуральных чисел, упорядоченное по возрастанию, является структурой. Причем для любой пары чисел $a, b \in N$, где $a \leqslant b$, $\inf_N\{a, b\} = a$, $\sup_N\{a, b\} = b$.

Пример 7. Пусть $C = s_1 \underset{F(\mathfrak{A})}{\subseteq} s_2 \underset{F(\mathfrak{A})}{\subseteq} \cdots \underset{F(\mathfrak{A})}{\subseteq} s_i \underset{F(\mathfrak{A})}{\subseteq} \cdots$ — цепь в частично упорядоченном множестве $F(\mathfrak{A})$ с отношением порядка $\underset{F(\mathfrak{A})}{\subseteq}$ (см. § 2.4, пример 7). Цепь C является структурой, причем для любой пары слов $s, s' \in C$ таких, что $s \underset{F(\mathfrak{A})}{\subseteq} s'$, $\inf_C\{s, s'\} = s$, $\sup_C\{s, s'\} = s'$.

Пример 8. Множество $\mathfrak{M}(A)$ всех подмножеств некоторого непустого множества A, частично упорядоченное по отношению

включения $\subseteq$ (см. § 2.4, пример 8), является структурой. Причем $\inf_{\mathfrak{M}(A)}\{A_1, A_2\} = A_1 \cap A_2$, $\sup_{\mathfrak{M}(A)}\{A_1, A_2\} = A_1 \cup A_2$ для любых $A_1, A_2 \subseteq A$. В частности, структурами являются множество $\mathfrak{M}(A^2)$ всех бинарных отношений на множестве A, а также множество $\mathfrak{M}[F(\mathfrak{A})]$ всех языков в алфавите $\mathfrak{A}$.

Понятие структуры может быть определено также с помощью системы аксиом, которым должны удовлетворять бинарные операции $\vee$, $\wedge$ [66]:

$$a \wedge a = a, \quad a \vee a = a, \tag{2.17}$$

$$a \wedge b = b \wedge a, \quad a \vee b = b \vee a, \tag{2.18}$$

$$(a \wedge b) \wedge c = a \wedge (b \wedge c), \quad (a \vee b) \vee c = a \vee (b \vee c), \tag{2.19}$$

$$a \wedge (a \vee b) = a, \quad a \vee (a \wedge b) = a. \tag{2.20}$$

Для произвольной структуры S точные нижнюю и верхнюю грани условимся обозначать $\inf_S\{a, b\} = a \cap b$, $\sup_S\{a, b\} = a \cup b$ для любых $a, b \in S$. Покажем, что произвольная структура удовлетворяет приведенным выше аксиомам.

Справедливость аксиом (2.17), (2.18) непосредственно следует из определения точных нижней и верхней граней: $\inf_S\{a, a\} = a$, $\sup_S\{a, a\} = a$; $\inf_S\{a, b\} = \inf_S\{b, a\}$, $\sup_S\{a, b\} = \sup_S\{b, a\}$. Докажем справедливость аксиомы (2.19). Пусть $(a \cap b) \cap c = d$, тогда $d \leqslant (a \cap b)$ и $d \leqslant c$, а из $d \leqslant a \cap b$ следует $d \leqslant a$ и $d \leqslant b$. Но из $d \leqslant b$ и $d \leqslant c$ следует, что $d \leqslant b \cap c$. Кроме того, $d \leqslant a$, и поэтому $d \leqslant a \cap (b \cap c)$. Значит, $(a \cap b) \cap c \leqslant a \cap (b \cap c)$. Аналогично нетрудно показать, что $a \cap (b \cap c) \leqslant (a \cap b) \cap c$. Таким образом, $(a \cap b) \cap c = a \cap (b \cap c)$. Справедливость аксиомы $(a \cup b) \cup c = a \cup (b \cup c)$ доказывается двойственным образом.

Остается доказать справедливость аксиомы (2.20). Пусть $a \cap (a \cup b) = q$. Тогда $q \leqslant a$. С другой стороны, вследствие рефлексивности отношения частичного порядка $a \leqslant a$, кроме того, $a \leqslant a \cup b$. Следовательно, a — нижняя грань элементов a и $a \cup b$. Но q — точная нижняя грань и поэтому $a \leqslant q$, таким образом, $a = q$. Справедливость аксиомы $a \cup (a \cap b) = a$ доказывается двойственным образом.

Пусть теперь задано множество S с операциями $\wedge$, $\vee$, которые удовлетворяют аксиомам (2.17)—(2.20). Для произвольных элементов $a, b \in S$ из $a \wedge b = a$ следует $a \vee b = b$ и наоборот, так что соотношения

$$a \wedge b = a, \qquad a \vee b = b \tag{2.21}$$

одновременно или выполняются или не выполняются в множестве S. Действительно, пусть $a \wedge b = a$. Тогда $a \vee b = (a \wedge b) \vee b$, откуда в силу аксиом (2.18), (2.20) $a \vee b = b \vee (b \wedge a) = b$. Наоборот, из $a \vee b = b$ следует, что $a \wedge b = a \wedge (a \vee b) = a$.

Определим на множестве S отношение $\leqslant$ так, что $a \leqslant b$ тогда и только тогда, когда для элементов $a, b \in S$ справедливы соотношения (2.21). Очевидно, отношение $\leqslant$ на множестве является отношением частичного порядка. Действительно, вследствие аксиомы (2.17) отношение $\leqslant$ рефлексивно, $a \leqslant a$. Отношение $\leqslant$ антисимметрично, так как из $a \leqslant b$ и $b \leqslant a$ следует $a \wedge b = a$ и $b \wedge a = b$, но вследствие аксиомы (2.18) $a \wedge b = b \wedge a$ и поэтому $a = b$. Наконец, пусть $a \leqslant b$ и $b \leqslant c$. Тогда $a \wedge b = a$, $b \wedge c = b$ и, значит, $a \wedge c = (a \wedge b) \wedge c$. Применив аксиому (2.19), получим $a \wedge c = (a \wedge b) \wedge c = a \wedge (b \wedge c) = a \wedge b = a$, откуда $a \leqslant c$ и отношение $\leqslant$ транзитивно.

Покажем теперь, что для любых $a, b \in S$ $a \wedge b = a \cap b$ и $a \vee b = a \cup b$, т. е. $a \wedge b$ и $a \vee b$ являются точными соответственно нижней и верхней гранями элементов a, b в множестве S. Действительно, из $(a \wedge b) \wedge a = a \wedge (b \wedge a) = a \wedge (a \wedge b) = (a \wedge a) \wedge b = a \wedge b$ следует, что $a \wedge b \leqslant a$. Аналогично $a \wedge b \leqslant b$. Следовательно, $a \wedge b$ — нижняя грань элементов a, b. Покажем, что $a \wedge b = \inf_S \{a, b\} = a \cap b$. Пусть $c \in S$ — некоторая нижняя грань элементов a, b. Тогда $c \leqslant a$, $c \leqslant b$, т. е. $c \wedge a = c$ и $c \wedge b = c$. Но $c \wedge (a \wedge b) = (c \wedge a) \wedge b = c \wedge b = c$ и, значит, $c \leqslant a \wedge b$. Двойственным образом можно показать, что $a \vee b = \sup_S \{a, b\} = a \cup b$ для любых $a, b \in S$.

Таким образом, справедливо следующее утверждение.

Теорема 2.4. *Множество S с бинарными операциями $\wedge$, $\vee$ тогда и только тогда является структурой, когда операции $\wedge$, $\vee$ удовлетворяют аксиомам (2.17) — (2.20).*

Структуры S и S' называются **изоморфными**, если существует взаимно-однозначное отображение φ структуры S на структуру S' такое, что для любых элементов $a, b \in S$ выполняются равенства

$$(a \cap b)\varphi = (a)\varphi \cap (b)\varphi, \quad (a \cup b)\varphi = (a)\varphi \cup (b)\varphi. \tag{2.22}$$

Отображение φ называется **изоморфным отображением** структуры S на структуру S'.

Теорема 2.5. *Структуры S и S' тогда и только тогда изоморфны, когда они изоморфны как частично упорядоченные множества.*

Доказательство. Пусть структуры S и S' изоморфны в смысле приведенного выше определения, т. е. для некоторого взаимно-однозначного отображения φ структуры S на S' выполняются равенства (2.22). Если $a, b \in S$, причем $a \leqslant b$, то $a \cap b = a$ и $(a)\varphi \cap (b)\varphi = (a)\varphi$, откуда следует $(a)\varphi \underset{S'}{\leqslant} (b)\varphi$. Учитывая взаимную однозначность отображения φ, нетрудно показать, что из $(a)\varphi \underset{S'}{\leqslant} (b)\varphi$ следует $a \underset{S}{\leqslant} b$. Обратно: если структуры S и S' изоморфны как частично упорядоченные множества, то они изоморфны в смысле приведенного выше определения. Покажем, что

для любых $a, b \in S$ $(a \cap b)\varphi = (a)\varphi \cap (b)\varphi$, где φ — изоморфное отображение частично упорядоченного множества S на S' (соотношение $(a \cup b)\varphi = (a)\varphi \cup (b)\varphi$ доказывается двойственным образом). По определению точной нижней грани $(a \cap b) \underset{S}{\leqslant} a$ и $(a \cap b) \underset{S}{\leqslant} b$. Тогда $(a \cap b)\varphi \underset{S'}{\leqslant} (a)\varphi$ и $(a \cap b)\varphi \underset{S'}{\leqslant} (b)\varphi$. Далее, для любого элемента $c' \in S'$ такого, что $c' \underset{S'}{\leqslant} (a)\varphi$ и $c' \underset{S'}{\leqslant} (b)\varphi$, в силу взаимной однозначности отображения φ найдется элемент $c \in S$ такой, что $(c)\varphi = c'$, причем $c \underset{S}{\leqslant} a$ и $c \underset{S}{\leqslant} b$, но тогда $c \underset{S}{\leqslant} a \cap b$ и поэтому $c' \underset{S'}{\leqslant} (a \cap b)\varphi$. Таким образом, $(a \cap b)\varphi = (a)\varphi \cap (b)\varphi$.

Частично упорядоченное множество S называется **полной структурой**, если для любого непустого подмножества $B \subseteq S$ в множестве S существуют точные нижняя и верхняя грани, которые, как и в структурах, обозначаются $\inf_S B = \bigcap\limits_{a \in B} a$, $\sup_S B = \bigcap\limits_{a \in B} a$. Очевидно, любая полная структура является структурой. Так, полными структурами являются множества $\mathfrak{M}(A)$, $\mathfrak{M}(A^2)$, $\mathfrak{M}[F(\mathfrak{A})]$ (см. пример 8). В то же время не любая структура является полной структурой. Так, множество N всех натуральных чисел является структурой (см. пример 6) и не является полной структурой, поскольку любое бесконечное подмножество множества N не имеет точной нижней грани. Точная нижняя грань всего множества S называется **нулем полной структуры**, $0 = \bigcap\limits_{a \in S} a$, а точная верхняя грань множества S — **единицей** данной структуры, $1 = \bigcup\limits_{a \in S} a$. Очевидно, элементы 0 и 1 являются соответственно наименьшим и наибольшим элементами множества S в смысле его частичной упорядоченности. Для 0 и 1 справедливы следующие соотношения: $0 \cap a = 0$, $1 \cap a = a$, $0 \cup a = a$, $1 \cup a = 1$. Разумеется, элементы 0 и 1 (или один из них) могут быть не только в полных структурах (см. пример 2).

Линейно упорядоченное множество $a_0 \leqslant a_1 \leqslant \ldots \leqslant a_n$, являющееся подмножеством некоторого частично упорядоченного множества A с элементами 0 и 1, называется **композиционным рядом**, если $a_0 = 0$, $a_n = 1$ и все интервалы $[a_{i-1}, a_i]$ $(i = 1, \ldots, n)$ простые. Число n называется **длиной** композиционного ряда.

Теорема 2.6. *Частично упорядоченное множество A является полной структурой тогда и только тогда, когда в нем есть единичный элемент и для любого непустого подмножества $B \subseteq A$ существует точная нижняя грань $\inf_A B = \bigcap\limits_{a \in B} a$.*

Достаточно показать, что для любого непустого подмножества $B \subseteq A$ в множестве A существует точная верхняя грань $\sup_A B =$

$= \bigcup_{a \in B} a$. Рассмотрим верхний конус $B^{\triangle}$, состоящий из всех верхних граней множества B. Очевидно, $B^{\triangle}$ непусто, так как в него входит единичный элемент, который является верхней гранью любого подмножества. Но по условию теоремы существует элемент $b = \bigcap_{a \in B^{\triangle}} a$, который и является точной верхней гранью подмножества $B \subseteq A$.

В силу принципа двойственности частично упорядоченное множество A тогда и только тогда является полной структурой, когда в нем есть нулевой элемент и для любого непустого подмножества $B \subseteq A$ существует точная верхняя грань $\sup_A B = \bigcap_{a \in B} a$.

В аксиомах (2.17)—(2.20), определяющих структуру S, операции $\vee$, $\wedge$ связаны лишь аксиомой (2.20). Однако существуют структуры с более тесной связью операций $\wedge$, $\vee$. К последним относятся так называемые **дистрибутивные структуры**, в которых выполняется аксиома дистрибутивности

$$\left.\begin{aligned}(a \wedge (b \vee c)) &= (a \wedge b) \vee (a \wedge c),\\ (a \vee (b \wedge c)) &= (a \vee b) \wedge (a \vee c).\end{aligned}\right\} \qquad (2.23)$$

Например, дистрибутивными являются структуры $\mathfrak{M}(A)$, $\mathfrak{M}(A^2)$, $\mathfrak{M}[F(\mathfrak{A})]$ (см. пример 8).

Дистрибутивная структура S с элементами 0 и 1 называется **булевой структурой** или **булевой алгеброй**, если для любого элемента $a \in S$ существует элемент $\bar{a} \in S$ такой, что $a \wedge \bar{a} = 0$ и $a \vee \bar{a} = 1$; элемент $\bar{a}$ называется **дополнением** элемента a в структуре S. Структура $\mathfrak{M}(A)$ является булевой структурой, так как для любого подмножества $B \subseteq A$ существует теоретико-множественное дополнение $\overline{B} \subseteq A$. В частности, булевыми структурами являются множество $\mathfrak{M}(\overline{A^2})$ всех бинарных отношений на множестве A и множество $\mathfrak{M}[F(\mathfrak{A})]$ всех языков в алфавите $\mathfrak{A}$.

§ 3.1. Понятия моделей и универсальных алгебр

Пусть A — некоторое непустое множество; $\{\xi^{n_1}_1, \xi^{n_2}_2, \ldots, \xi^{n_s}_s, \ldots\}$ — система n-отношений на множестве A (конечная или бесконечная). Каждому n-отношению ξ^n на множестве A можно сопоставить n-местную логическую функцию **(предикат)** $\rho^n: A^n \to \{и, л\}$ так, что $\rho^n(a_{i_1}, a_{i_2}, \ldots, a_{i_n}) =$ и тогда и только тогда, когда выполняется n-отношение $\xi^n(a_{i_1}, a_{i_2}, \ldots, a_{i_n})$ (и, л — логические значения истины и лжи соответственно; $a_{i_1}, a_{i_2}, \ldots, a_{i_n} \in A$).

Моделью $M_A = \langle A; \pi\rangle$ называется система, состоящая из множества A и определенной на данном множестве совокупности предикатов $\pi = \{\rho^{n_s}_s \mid s = 1, 2, \ldots\}$ [77]. Множество A называется **основным** множеством данной модели; предикаты, принадлежащие π, — ее основными или главными предикатами. Последовательность $n_1, n_2, \ldots, n_s, \ldots$ называется **типом** модели M_A, а совокупность $\tau = \{\rho^{n_s}_s \mid s = 1, 2, \ldots\}$ — ее **сигнатурой**. Моделями, например, являются множества с определенными на них предикатами, соответствующими отношениям эквивалентности и частичного порядка (см. § 2.4, 2.5).

Модель $M_{A_1} = \langle A_1; \pi'\rangle$ (A_1 — непустое подмножество множества A; $\overline{\rho}^n \in \pi'$ — предикат на множестве A_1, индуцированный предикатом $\overline{\rho}^n \in \pi$) называется **подмоделью** модели $M_A = \langle A; \pi\rangle$. Множество однотипных моделей, т. е. моделей, типы которых совпадают, образует класс моделей по данному типу.

Однотипные модели $M_A = \langle A; \pi\rangle$ и $M_B = \langle B; \pi'\rangle$ ($\pi = \{\rho^{n_s}_s \mid s = 1, 2, \ldots\}$; $\pi' = \{\sigma^{n_s}_s \mid s = 1, 2, \ldots\}$) называются **изоморфными**, если существует взаимно-однозначное отображение φ множества A на множество B такое, что для любого $s = 1, 2, \ldots$ предикат $\rho^{n_s}_s(a_1, a_2, \ldots, a_{n_s})$ справедлив тогда и только тогда, когда выполняется предикат $\sigma^{n_s}_s((a_1)\varphi, (a_2)\varphi, \ldots, (a_{n_s})\varphi)$ для произвольных $a_1, a_2, \ldots, a_{n_s} \in A$. Отображение φ в этом случае называется **изоморфизмом** модели M_A на M_B.

Понятие изоморфизма модели позволяет изучать абстрактные свойства предикатов и связанных с ними n-отношений, не зависящие от природы элементов основного множества, на котором эти предикаты и отношения определены.

Рассмотрим модель $M_A = \langle A; \pi \rangle$, где каждый n-местный предикат, входящий в π, соответствует функциональному отношению φ^n на множестве A (вообще говоря, частично определенному). С каждым таким отношением φ^n связана некоторая частично определенная $(n-1)$-местная функция $F_{\varphi^n}(x_1, x_2, \ldots, x_{n-1})$. Напомним, что если отношение φ^{n+1} функционально, то для набора элементов $a_{i_1}, a_{i_2}, \ldots, a_{i_n} \in A$ существует не более одного элемента $a_{i_{n+1}}$ такого, что $(a_{i_1}, a_{i_2}, \ldots, a_{i_n}, a_{i_{n+1}}) \in \varphi_A^{n+1}$, причем $F_{\varphi^{n+1}}(a_{i_1}, a_{i_2}, \ldots, a_{i_n}) = a_{i_{n+1}}$ (см. § 2.3).

Частично определенная функция $F_{\varphi^{n+1}}(x_1, \ldots, x_n)$ называется **n-арной частичной операцией** на множестве A. В случае, когда функция $F_{\varphi^{n+1}}(x_1, \ldots, x_n)$ всюду определена, говорят просто об n-арной операции на множестве A (см. § 1.2). Бинарными ($n = 2$), например, являются известные арифметические операции над числами (сложение, умножение и др.), а также теоретико-множественные операции (объединение $\bigcup$, пересечение $\bigcap$, декартово произведение $\times$ и др.). Унарная ($n = 1$) операция $F(x)$ однозначно сопоставляет каждому элементу $a \in A$ некоторый элемент $F(a) \in A$, т. е. является отображением множества A в себя. Унарной, например, является операция логического отрицания $\neg$, а также операция теоретико-множественного дополнения. Часто рассматриваются нульарные ($n = 0$) операции, каждая из которых фиксирует в основном множестве A некоторый элемент $a \in A$, не зависящий от других элементов из множества A или от их систем. Иными словами, функция F_{φ^1}, реализующая нульарную операцию, тождественна константе $F_{\varphi^1} \equiv a$ для некоторого $a \in A$.

Система $U_A = \langle A; \Omega \rangle$, состоящая из основного множества A и определенной на нем совокупности частичных операций $\Omega = \{F_s^{n_s}(x_1, \ldots, x_{n_s}) \mid s = 1, 2, \ldots\}$, называется **частичной универсальной алгеброй** типа $(n_s \mid s = 1, 2, \ldots)$ с сигнатурой операций $\Omega = \{F_s^{n_s} \mid s = 1, 2, \ldots\}$. Если каждая из операций, принадлежащих сигнатуре Ω, всюду определена на множестве A, говорят просто об **универсальной алгебре** $U_A = \langle A; \Omega \rangle$.

Как и в модели, сигнатура операций Ω в частичной универсальной алгебре может быть конечной или бесконечной. Однотипные универсальные алгебры часто рассматриваются как алгебры с одной и той же сигнатурой операций. Пусть $U_A = \langle A; \Omega \rangle$ и $U_B = \langle B; \Omega \rangle$ — однотипные алгебры. Алгебра U_A **изоморфна** алгебре U_B, если существует взаимно-однозначное отображение φ множества A на B такое, что произвольная m-местная операция $F \in \Omega$ удовлетворяет соотношению

$$[F(a_1, a_2, \ldots, a_m)]\varphi = F((a_1)\varphi, (a_2)\varphi, \ldots, (a_m)\varphi)$$

для любого набора элементов $(a_1, a_2, \ldots, a_m) \in A^m$. Отображение φ в этом случае называется **изоморфизмом** алгебры U_A на алгебру U_B.

Понятие изоморфизма универсальных алгебр позволяет изучать абстрактные свойства алгебраических операций, не зависящие от природы элементов основного множества, на котором эти операции определены. Проведенные выше рассуждения показывают, что каждой модели $M_A = \langle A; \pi \rangle$, где $\pi = \{\varphi_s^{n_s} \mid s = 1, 2, \ldots\}$ — сигнатура предикатов, связанных с функциональными отношениями, соответствует (вообще говоря, частичная) универсальная алгебра $U_A = \langle A; \Omega = \{F_s^{n_s} \mid s = 1, 2, \ldots\} \rangle$ и, наоборот, каждой частичной универсальной алгебре $U_A = \langle A; \Omega \rangle$ с сигнатурой $\Omega = \{F_s^{n_s} \mid s = 1, 2, \ldots\}$ соответствует модель $M_A = \langle A; \pi = \{\varphi_s^{n_s+1} \mid s = 1, 2, \ldots\} \rangle$, где $F_s^{n_s}$ — функция, связанная с $(n_s + 1)$-арным функциональным отношением.

Рассмотрим примеры универсальных алгебр.

Пример 1. Система $\langle N; \{+\} \rangle$, состоящая из основного множества N всех натуральных чисел с единственной бинарной операцией $+$ (сложение чисел), образует универсальную алгебру типа 2 с сигнатурой $\{+\}$.

Пример 2. Система $\langle N; \{\times\} \rangle$, состоящая из основного множества N, на котором определена бинарная операция $\times$ (умножение), образует универсальную алгебру типа 2 с сигнатурой $\{\times\}$.

Пример 3. Система $\langle N; \{+, -, \times, :\} \rangle$, состоящая из основного множества N с определенными на нем бинарными арифметическими операциями, образует частичную (операции $-$, $:$ не всюду определены) универсальную алгебру типа (2, 2, 2, 2) с сигнатурой $\{+, -, \times, :\}$.

В примерах 1—3 приведены универсальные алгебры с конечными сигнатурами.

Пример 4. Система $\langle V; \{+, -, \times, :, \uparrow n, \sqrt[n]{\ } \mid n = 2, 3, \ldots\} \rangle$, состоящая из основного множества V всех действительных чисел с определенными на нем бинарными арифметическими операциями $+$, $-$, $\times$, $:$ и бесконечным множеством унарных операций $\uparrow n$ (возвышение в n-ю степень) и $\sqrt[n]{\ }$ (извлечение корня n-й степени), образует частичную (операции: и $\sqrt[2k]{\ }$ не всюду определены) универсальную алгебру типа (2, 2, 2, 2, 1, 1, ...) с бесконечной сигнатурой.

Пусть $F(\mathfrak{A})$ — множество всех цепочек (слов) конечной длины в алфавите $\mathfrak{A} = \{a_1, a_2, \ldots, a_m\}$ (предполагается, что множество $F(\mathfrak{A})$ содержит также пустую цепочку e); $\mathfrak{M}(\mathfrak{A})$ — множество

всех языков над алфавитом $\mathfrak{A}$. Напомним, что под языком L над алфавитом $\mathfrak{A}$ понимается некоторое подмножество $L \subseteq F(\mathfrak{A})$. Рассмотрим некоторые операции над цепочками и языками [19].

Конкатенацией или умножением цепочек $s_1, s_2 \in F(\mathfrak{A})$ называется бинарная операция, которая каждой паре цепочек s_1, s_2 ставит в соответствие новую цепочку $s_1 s_2$, полученную в результате приписывания справа к цепочке s_1 цепочки s_2. Причем пустая цепочка e удовлетворяет соотношению

$$es = se = s \tag{3.1}$$

для любого $s \in F(\mathfrak{A})$. Операция конкатенации ассоциативна, так что

$$(s_1 s_2)\, s_3 = s_1 (s_2 s_3) = s_1 s_2 s_3. \tag{3.2}$$

Пример 5. Система $\langle F(\mathfrak{A}); \{\boxtimes\}\rangle$, состоящая из основного множества $F(\mathfrak{A})$ всех цепочек конечной длины в алфавите $\mathfrak{A}$ с определенной на данном множестве операцией конкатенации $\boxtimes$, образует универсальную алгебру типа 2 с конечной сигнатурой $\{\boxtimes\}$.

Универсальная алгебра с сигнатурой, имеющей одну ассоциативную бинарную операцию, называется **полугруппой** (см. § 3.6).

Каждый язык L представляет собой некоторое подмножество множества $F(\mathfrak{A})$, $L \subseteq F(\mathfrak{A})$. Поэтому можно говорить о бинарных операциях $\cup$ (объединение языков), $\cap$ (пересечение языков) и унарной операции $\neg$ (дополнение языка). Как показывалось в § 2.5, множество $\mathfrak{M}(\mathfrak{A})$ является булевой структурой и может быть представлено как универсальная алгебра типа (1, 2, 2) с сигнатурой $\{\neg, \cup, \cap\}$.

Пусть $q_1, q_2, \ldots, q_n$ — выделенные символы из алфавита $\mathfrak{A} = \{a_1, a_2, \ldots, a_m\}$ $(n \leqslant m)$. Сопоставим каждому символу q_i язык $L_i \subseteq F(\mathfrak{A})$ $(i = 1, 2, \ldots, n)$. Пусть, далее, $L \in \mathfrak{M}(\mathfrak{A})$ — некоторый произвольный язык.

Суперпозицией $S^{n+1}(q_1, \ldots, q_n)$ называется $(n+1)$-арная операция над языками $L_1, \ldots, L_n$ и L, порождающая новый язык $L' = S^{n+1}\left(\begin{matrix} q_1, & \ldots, & q_n \\ L_1, & \ldots, & L_n \end{matrix} L\right)$ такой, что $s \in L'$ тогда и только тогда, когда в языке L найдется цепочка $\tilde{s} \in L$, из которой можно получить s путем замены каждого символа q_i, входящего в $\tilde{s}$, некоторой цепочкой, принадлежащей соответствующему языку L_i. Причем q_i в разных местах в цепочке $\tilde{s}$ можно заменять различными цепочками из L_i [23, 70]. Кратко суперпозиция $S^{n+1}\left(\begin{matrix} q_1, & \ldots, & q_n \\ L_1, & \ldots, & L_n \end{matrix} L\right)$ обозначается $S^{n+1}\left(\begin{matrix} q_i \\ L_i \end{matrix} L\right)$ или S^{n+1} (если ясно, о каких q_i, L_i и L

идет речь). Для операций суперпозиции выполняются следующие соотношения:

$$S^{n+1}\left(\begin{matrix} q_i \\ L_i \end{matrix} e\right) = e \quad (e \text{ — пустое слово}), \tag{3.3}$$

$$S^{n+1}\left(\begin{matrix} q_i \\ L_i \end{matrix} \varepsilon\right) = \varepsilon \quad (\varepsilon = \varnothing \text{ — пустой язык}), \tag{3.4}$$

$$S^{n+1}\left(\begin{matrix} q_i \\ \{q_i\} \end{matrix} L\right) = L \quad (i = 1, 2, \ldots, n;\ n = m), \tag{3.5}$$

$$S^{n+1}\left(\begin{matrix} q_1, & \ldots, & q_n \\ L_1, & \ldots, & L_n \end{matrix} L\right) = S^{m+1}\left(\begin{matrix} q_1, & \ldots, & q_n, & q_{n+1}, & \ldots, & q_m \\ L_1, & \ldots, & L_n, & \{q_{n+1}\}, & \ldots, & \{q_m\} \end{matrix} L\right), \tag{3.6}$$

$$S^{m+1}\left(\begin{matrix} q_i \\ \{e\} \end{matrix} L\right) = e \quad (L \neq \varepsilon;\ i = 1, 2, \ldots, m), \tag{3.7}$$

$$S^{n+1}\left(\begin{matrix} q_1, & \ldots, & q_n \\ L_1, & \ldots, & L_n \end{matrix} L\right) = L \,(L \subseteq F(\mathfrak{A} \setminus \{q_1, \ldots, q_n\})). \tag{3.8}$$

Пусть $S = \left\{ S^{n+1}\left(\begin{matrix} q_i \\ L_i \end{matrix} L\right) \middle| q_i \in \mathfrak{A};\ i = 1, 2, \ldots, n \right\}$ — совокупность всевозможных суперпозиций на множестве $\mathfrak{M}(\mathfrak{A})$ всех языков над алфавитом $\mathfrak{A}$.

Пример 6. Система $\langle \mathfrak{M}(\mathfrak{A}); S\rangle$ образует универсальную алгебру с основным множеством $\mathfrak{M}(\mathfrak{A})$ и конечной сигнатурой S.

Пример 7. Универсальной алгеброй является система $\langle \mathfrak{M}_h(\mathfrak{A}); S\rangle$ с основным множеством $\mathfrak{M}_h(\mathfrak{A}) \subset \mathfrak{M}(\mathfrak{A})$ всех конечных языков над алфавитом $\mathfrak{A}$. Напомним, что язык $L \subset F(\mathfrak{A})$ называется конечным, если он состоит из конечного множества цепочек. (Универсальные алгебры $\langle \mathfrak{M}(\mathfrak{A}); S\rangle$ и $\langle \mathfrak{M}_h(\mathfrak{A}); S\rangle$, а также некоторые их модификации, связанные с расширением сигнатуры операций этих алгебр, более подробно изучены в § 5.5. См. также § 3.2, 3.3, примеры.)

Рассмотрим частичную универсальную алгебру $U_A = \langle A; \Omega\rangle$. Из основных операций сигнатуры $\Omega = \{F_s^{n_s} \mid s = 1, 2, \ldots\}$ можно получить другие операции над множеством A, используя суперпозицию $\tilde{S}^{n+1}(f_1, f_2, \ldots, f_n; f^n)$ функций $f_1, f_2, \ldots, f_n$; f^n, смысл которой, как обычно, состоит в подстановке функций $f_1, f_2, \ldots, f_n$ вместо соответствующих аргументов в функцию $f^n(x_1, x_2, \ldots, x_n)$. Например, суперпозиция $\tilde{S}^4(f_1, f_2, f_3)$ функций $f_1(x_1) = x_1$, $f_2(x_2, x_3) = x_2 x_3$, $f_3(x_1, x_2) = x_1 + x_2$ порождает функцию $f(x_1, x_2, x_3) = f_3(f_1(x_1), f_2(x_2, x_3)) = x_1^m + x_2 x_3$. Если $f^n(x_1, \ldots, x_n)$ — некоторая функция, определенная на множестве A, то с помощью суперпозиции $\tilde{S}^{n+1}(y_1, y_2, \ldots, y_n; f^n) = f^n(y_1, y_2, \ldots, y_n)$ аргументы функции f^n можно переименовать, в частности отождествить или переставить их.

Функции, соответствующие суперпозициям операций алгебры $U_A = \langle A; \Omega\rangle$, называются **производными операциями** данной

алгебры. Пусть $\tilde{S} = \{\tilde{S}^{n+1}(f_1, f_2, \ldots, f_n; f^n) \mid n = 1, 2, \ldots\}$ — совокупность всевозможных операций суперпозиции. Рассмотрим универсальные алгебры $U_A = \langle A; \Omega\rangle$ и $U'_A = \langle A; \Omega'\rangle$ с основным множеством A и сигнатурами операций соответственно Ω и Ω'.

Алгебра U'_A называется **производной** от алгебры U_A, если каждая операция $f^n \in \Omega'$ является производной в алгебре $U_A = \langle A; \Omega\rangle$. Иными словами, алгебра U'_A производна от алгебры U_A, если $\Omega' \subseteq [\Omega]_{\tilde{S}}$, где $[R]_{\tilde{S}}$ — множество всех функций, порожденных функциями, принадлежащими совокупности R, с помощью операций суперпозиции (множество $[R]_{\tilde{S}}$ называется **замыканием** множества R по операциям суперпозиции).

Пример 8. Система $\langle \Phi_A, \tilde{S}\rangle$ является универсальной алгеброй с основным множеством Φ_A всех операций, определенных на A, и бесконечной сигнатурой $\tilde{S} = \{\tilde{S}^{n+1} \mid n = 1, 2, \ldots\}$. Алгебра $\langle \Phi_A; \tilde{S}\rangle$, очевидно, является частичной универсальной алгеброй, так как операции суперпозиции $\tilde{S}^{n+1}(f_1, \ldots, f_n; f)$ не всюду определены. Действительно, суперпозиция $\tilde{S}^{n+1}(f_1, f_2, \ldots, f_n; f)$ определена лишь в случае, когда функция f n-арная.

Пример 9. Частичную универсальную алгебру $\langle \Phi_A; \tilde{S}\rangle$ можно свести [78] к универсальной алгебре $\langle \Phi_A; \{\zeta, \tau, \Delta, \nabla, \curlywedge\}\rangle$ с всюду определенными сигнатурными операциями ζ, τ, Δ, ∇, $\curlywedge$, которые определяются следующими тождествами:

$$
\begin{aligned}
(\zeta f)(x_1, \ldots, x_m) &= f(x_2, \ldots, x_m, x_1),\\
(\tau f)(x_1, x_2, x_3, \ldots, x_m) &= f(x_2, x_1, x_3, \ldots, x_m),\\
(\Delta f)(x_1, x_2, x_3, \ldots, x_{m-1}) &= f(x_1, x_1, x_2, \ldots, x_{m-1}),\\
(\nabla f)(x_1, x_2, x_3, \ldots, x_{m+1}) &= f(x_2, x_3, \ldots, x_{m+1}),\\
(f \curlywedge g)(x_1, x_2, \ldots, x_{k+m-1}) &= f(g(x_1, \ldots, x_k), x_{k+1}, \ldots, x_{k+m-1}),
\end{aligned}
$$

где f, $g \in \Phi_A$ — произвольные m- и k-местные функции. Если функция f одноместная, то можно положить $\zeta f = \tau f = \Delta f = f$. Операции ζ, τ, Δ позволяют производить всевозможные отождествления и перестановки переменных функций f, а операция ∇ — присоединять к функции f любое число фиктивных аргументов.

Любая функция $f \in \Phi_A$ тогда и только тогда представима в виде суперпозиции $f = S(f_1, \ldots, f_n; g^n)$, где $f_1, \ldots, f_n, g^n \in \Phi_A$, когда функция f может быть получена из функций $f_1, f_2, \ldots, f_n$, g^n с помощью операций ζ, τ, Δ, ∇, $\curlywedge$. Например,

$$
\begin{aligned}
f(g_1(x), g_2(x), \ldots, g_m(x)) = \Delta^{m-1}(\zeta \ldots \zeta(\zeta(f \curlywedge g_1) \curlywedge g_2) \curlywedge\\
\curlywedge \cdots \curlywedge g_m(x)).
\end{aligned}
$$

Пусть множество A совпадает с множеством $E_k = \{0, 1, \ldots, k-1\}$, состоящим из k чисел ($k \geqslant 2$). Функции $f^n(x_1, x_2, \ldots, x_n)$, определенные на множестве E_k и принимающие значения из этого множества, называются **функциями k-значной логики**. При $k = 2$ функции f^n называются **булевыми или функциями алгебры логики**.

Пример 10. Система $\langle \Phi_{E_k}; \{\zeta, \tau, \Delta, \nabla, \curlywedge\}\rangle$, состоящая из основного множества Φ_{E_k} всех функций k-значной логики с определенной на данном множестве сигнатурой операций $\{\zeta, \tau, \Delta, \nabla, \curlywedge\}$, позволяющих осуществить любую суперпозицию функций из Φ_{E_k}, является универсальной алгеброй k-значных функций.

Универсальная алгебра k-значных функций называется **k-значной алгеброй Поста** [78, 133]. k-Значные алгебры Поста, а также их модификации, связанные с видоизменением сигнатуры операций, находят самые разнообразные приложения в различных областях кибернетики, в частности в абстрактной теории конечных и бесконечных автоматов, теории алгоритмов и программировании (см. гл. 4).

Пусть $\mathfrak{L} = \{L\}$ — множество языков и $T = \left\{\overset{m}{L'}, \overset{m}{L''}, L\right\}$ — множество трансляторов, где транслятор $\overset{m}{L'}, \overset{m}{L''}, L$ представляет собой программу в языке L и переводит программу с входного языка L' на выходной язык L''. Транслятор $\overset{m}{L'}, \overset{m}{L''}, L$ можно рассматривать как унарную операцию, областью определения которой является язык L', а областью значений — язык L''. Так что $m(x) = y$, где x — программа в языке L'; y — программа в языке L'', представляющая собой перевод программы x. В частности, переработав транслятор $\overset{m}{L'}, \overset{m}{L''}, L_1$ транслятором $\overset{m}{L_1}, \overset{m}{L_2}, L$, получим

$$
\overset{m}{L_1}, \overset{m}{L_2}, L\left(\overset{m}{L'}, \overset{m}{L''}, L_1\right) = \overset{m}{L'}, \overset{m}{L''}, L_2.
$$

Иными словами, если транслятор m^1 является программой в языке L_1, который служит входным языком транслятора m^2, то, применив m^2 к m^1, получим запись m^1 в выходном языке L_2 транслятора m^2. Такое преобразование транслятора называется **операцией перекодировки**.

Над трансляторами $\overset{m}{L_1}, \overset{m}{L_2}, L$ и $\overset{m}{L_2}, \overset{m}{L_3}, L$ естественно определяется **операция композиции** (последовательного применения трансляторов), так что

$$
\overset{m}{L_1}, \overset{m}{L_2}, L \curlywedge \overset{m}{L_2}, \overset{m}{L_3}, L = \overset{m}{L_1}, \overset{m}{L_3}, L.
$$

В терминах указанных операций могут быть формализованы процессы, часто используемые в программировании.

Пример 11. Пусть L_1, L_2 — языки машин соответственно M_1, M_2. Причем на машине M_1 транслятор $\overset{m}{L}, L_1, L_1$ осуществляет перевод с языка высокого уровня L на язык этой машины L_1, транслятор $\overset{m}{L}, L_2, L$ осуществляет перевод с языка L на язык L_2 и описан на языке высокого уровня L. Тогда, дважды применяя операцию перекодировки, можно получить транслятор с языка L на L_2, работающий на машине M_2. Так,

$$\overset{m}{L}, L_1, L_1\left(\overset{m}{L}, L_2, L\right) = \overset{m}{L}, L_2, L_1, \quad \overset{m}{L}, L_2, L_1\left(\overset{m}{L}, L_2, L\right) = \overset{m}{L}, L_2, L_2.$$

Смысл приведенного примера состоит в том, что программу трансляции или отдельные ее блоки удобнее писать на языке высокого уровня с последующей перекодировкой на язык рабочей машины. Особенно важное значение это имеет при разработке трансляторов с одного и того же языка на языки разных машин. У таких трансляторов обычно достаточно большая общая часть, реализующая синтаксический анализ входного языка и некоторые другие блоки. Подобные преобразования использовались при конструировании транслятора для машины «Киев», переводящего с адресного языка на язык машины «Днепр» [103].

Пример 12. Пусть L_0 — язык некоторой машины M; L_0, L_1, L_2, L_3 — упорядоченная последовательность языков, для которых существуют трансляторы $\overset{m}{L_3}, L_2, L_2$, $\overset{m}{L_2}, L_1, L_1$, $\overset{m}{L_1}, L_0, L_0$. Тогда, используя операции перекодировки и композиции для любой пары языков L_i, L_j $(i > j)$, можно получить транслятор для машины M, переводящий с языка L_i более высокого уровня на язык L_j более низкого уровня. Так, для того чтобы получить транслятор $\overset{m}{L_2}, L_0, L_0$, необходимо осуществить перекодировку

$$\overset{m}{L_1}, L_0, L_0\left(\overset{m}{L_2}, L_1, L_1\right) = \overset{m}{L_2}, L_1, L_0.$$

Затем, выполнив композицию, получим

$$\overset{m}{L_2}, L_1, L_0 \times \overset{m}{L_1}, L_0, L_0 = \overset{m}{L_2}, L_0, L_0.$$

Аналогично

$$\overset{m}{L_2}, L_0, L_0\left(\overset{m}{L_3}, L_2, L_2\right) = \overset{m}{L_3}, L_2, L_0,$$

$$\overset{m}{L_3}, L_2, L_0 \times \overset{m}{L_2}, L_0, L_0 = \overset{m}{L_3}, L_0, L_0$$

и т. д.

Приведенный пример свидетельствует о том, что технология конструирования транслятора с развитого языка программирования, как правило, состоит в выделении иерархии промежуточных уровней языка и разработке фаз трансляции для смежных уровней с последующим сопряжением этих фаз.

Пример 13. Пусть заданы языки высокого уровня $\{L_i \mid i = 1, 2, \ldots, r\}$ и машины $\{M_j \mid j = 1, 2, \ldots, s\}$ и каждая машина M_j имеет свой язык R_j. Тогда, для того чтобы обеспечить перевод с любого языка L_i на язык R_j каждой машины, необходимо построить $r \cdot s$ трансляторов. Используя промежуточный язык L, для которого конструируются трансляторы $\left\{\overset{m^i}{L_i}, L, L \,\middle|\, i = 1, 2, \ldots, r\right\}$ с заданных языков на язык L и трансляторы $\left\{\overset{m^j}{L}, R_j, R_j \,\middle|\, j = 1, 2, \ldots, s\right\}$ с языка L на языки машин, и применяя затем операции перекодировки и композиции для каждой машины M_j, можно получить транслятор $\overset{m}{L_i}, R_j, R_j$ с произвольного языка L_i на язык R_j данной машины.

Введение промежуточного языка позволяет сократить число необходимых трансляторов до $r + s$. В программировании известны два подхода к выделению промежуточного языка при решении проблемы трансляции. 1. Промежуточный язык может быть ориентирован на класс машин. Такой подход применен при создании языка АЛМО [52]. 2. Промежуточный язык может быть ориентирован на класс входных языков. В этом случае он представляет собой параметрическую систему, которая включает в себя ядро, отражающее общие черты класса входных языков, и параметры, отражающие специфику отдельного входного языка из заданного класса (см. гл. 6).

Пусть $\{M\}$ — множество машин, каждая из которых характеризуется лишь своим языком L_M так, что машины M_1 и M_2 совпадают, если $L_{M_1} = L_{M_2}$. Тогда существуют программы, преобразующие одни машины в другие. Пусть, далее, R_M — язык машины M. Тогда **интерпретатор** $\overset{u}{L}, R$ с языка высокого уровня L, реализованный на машине M, позволяет выполнять на данной машине программы на языке L. Тем самым интерпретатор $\overset{u}{L}, R$ преобразует машину M в новую машину M' с языком $L_{M'} = L$. Такого рода программами являются также **эммуляторы**, моделирующие математическое обеспечение (МО) одной машины на другой. В частности, для автоматизации процесса разработки современных ЭВМ часто используют рабочую машину M_1, на которой моделируют систему МО проектируемой машины M_2. Машина M_1 в таком

случае называется **инструментальной**, а система, моделирующая математическое обеспечение машины M_2, — **иммитатором**. Интерпретаторы, эммуляторы и иммитаторы можно рассматривать как операции, определенные на множестве машин и принимающие значения в этом же множестве.

Применение аппарата универсальных алгебр для формализации процессов программирования позволяет, абстрагируясь от структуры рассматриваемых объектов (машин, программ, трансляторов и др.), изучать их свойства, соотношения и взаимодействия.

§ 3.2. Подалгебры. Системы образующих

Пусть $U_A = \langle A; \Omega\rangle$ — универсальная алгебра (вообще говоря, частичная) и $A_1 \subseteq A$ — некоторое непустое подмножество A. Множество $[A_1]_F \subseteq A$ называется **замыканием** множества A_1 в алгебре U_A по n-местной операции $F \in \Omega$, если а) $A_1 \subseteq [A_1]_F$; б) для любых элементов $q_1, q_2, \ldots, q_n \in [A_1]_F$ таких, что значение $F(q_1, q_2, \ldots, q_n)$ определено, $F(q_1, q_2, \ldots, q_n) \in [A_1]_F$, при этом любой элемент из $[A_1]_F$ порождается в конечном счете из элементов A_1 посредством операции F. Если $A_1 \subseteq A_2$ (для некоторых подмножеств $A_1, A_2 \subseteq A$), то $[A_1]_F \subseteq [A_2]_F$, $[[A_1]_F]_F = [A_1]_F$ (для любых $F \in \Omega$).

Множество A_1 называется **замкнутым по n-местной операции $F \in \Omega$ в алгебре U_A**, если $A_1 = [A_1]_F$. Ясно, что множество A_1 замкнуто по операции $F \in \Omega$ в алгебре U_A тогда и только тогда, когда $F(a_1, a_2, \ldots, a_n) \in A_1$ для любой последовательности элементов $a_1, \ldots, a_n \in A_1$ такой, что значение $F(a_1, \ldots, a_n)$ определено.

Множество $[A_1] \subseteq A$ называется **замыканием множества A_1 в алгебре U_A**, если $[A_1]$ состоит из всех элементов, которые порождаются из элементов, принадлежащих A_1, с помощью основных и производных операций алгебры U_A (см. § 3.1). Замыкание $[A_1]$ множества A_1 в алгебре U_A может быть представлено следующим образом: $[A_1] = A_1^0 \cup A_1^1 \cup \cdots \cup A_1^m \cup \cdots$, где $A_1^0 = A_1$, $A_1^{i+1} = \bigcup\limits_F [A_1^i]_F$ (F пробегает множество Ω всех основных операций алгебры U_A, $i = 1, 2, \ldots$). Если $A_1 \subseteq A_2$ (для некоторых подмножеств $A_1, A_2 \subseteq A$), то $[A_1] \subseteq [A_2]$. Очевидно также, что $[[A_1]] = [A_1]$.

Множество A_1 называется **замкнутым в алгебре U_A**, если $A_1 = [A_1]$. Множество A_1 замкнуто в алгебре U_A тогда и только тогда, когда $F(a_1, a_2, \ldots, a_n) \in A_1$ для любой n-местной операции $F \in \Omega$ и любой последовательности элементов $a_1, \ldots, a_n \in A_1$, для которой значение $F(a_1, \ldots, a_n)$ определено.

Пусть множество $A_1 \subseteq A$ замкнуто в алгебре U_A. Тогда систему $U_{A_1} = \langle A_1; \Omega\rangle$ можно рассматривать как универсальную алгебру. Алгебра $U_{A_1} = \langle A_1; \Omega\rangle$ называется **подалгеброй** универсальной алгебры $U_A = \langle A; \Omega\rangle$. Примером несобственной подалгебры алгебры $U_A = \langle A; \Omega\rangle$ является сама эта алгебра. Заметим, что любая подалгебра $U_{A_1} = \langle A_1; \Omega\rangle$ алгебры U_A однозначно определяется соответствующим замкнутым множеством A_1. Поэтому в дальнейшем вместо «подалгебра $U_{A_1} = \langle A_1; \Omega\rangle$» будем писать «подалгебра A_1».

Приведем примеры подалгебр универсальных алгебр, рассмотренных в § 3.1.

Пример 1. Система $\langle N(r); \{+\}\rangle$, где $N(r) = \{n \mid n \geqslant r\} \subseteq N$, является подалгеброй алгебры $\langle N; \{+\}\rangle$.

Пример 2. Система $\langle N_p; \{\times\}\rangle$, где $N_p = \{n \mid n : p\} \subseteq N$ — множество всех чисел, кратных p (p — некоторое простое число), является подалгеброй алгебры $\langle N; \{\times\}\rangle$.

Пример 3. Тривиальной подалгеброй алгебры $\langle F(\mathfrak{A}); \{\boxtimes\}\rangle$ является система $\langle \{e\}; \{\boxtimes\}\rangle$, где e — пустая цепочка над алфавитом $\mathfrak{A}$.

Пример 4. Система $\langle \mathfrak{M}_k(\mathfrak{A}); S\rangle$ является подалгеброй алгебры $\langle \mathfrak{M}(\mathfrak{A}); S\rangle$.

Пример 5. Система $\langle P_0; \{\zeta, \tau, \Delta, \nabla, \divideontimes\}\rangle$, где $P_0 \subset \Phi_A$ — множество функций, существенно зависящих лишь от одной переменной, является подалгеброй алгебры $\langle \Phi_A; \{\zeta, \tau, \Delta, \nabla, \divideontimes\}\rangle$. В то же время система $\langle P_1; \{\zeta, \tau, \Delta, \nabla, \divideontimes\}\rangle$, где P_1 — множество всех k-значных функций, тождественных константам из E_k, является подалгеброй k-значной алгебры Поста $\langle \Phi_{E_k}; \{\zeta, \tau, \Delta, \nabla, \divideontimes\}\rangle$.

Теорема 3.1. *Пусть $\{U_{A_\alpha} \mid \alpha \in I\}$ — совокупность подалгебр алгебры $U_A = \langle A; \Omega\rangle$, где $A_\alpha \subseteq A$ — замкнутые множества в алгебре U_A. Непустое пересечение совокупности $\{U_{A_\alpha} \mid \alpha \in I\}$ подалгебр алгебры U_A также образует подалгебру данной алгебры.*

Действительно, достаточно показать, что непустое пересечение $D = \bigcap\limits_{\alpha \in I} A_\alpha$ является замкнутым множеством в алгебре U_A. Для любой n-местной операции $F \in \Omega$ и произвольной совокупности элементов $q_1, \ldots, q_n \in D$ такой, что значение $F(q_1, \ldots, q_n)$ определено, ввиду замкнутости множества A_α в алгебре U_A (для любого $\alpha \in I$) $F(q_1, \ldots, q_n) \in A_\alpha$. Следовательно, $F(q_1, \ldots, q_n) \in D$ для любой n-местной операции F, принадлежащей Ω, и произвольной совокупности элементов $q_1, \ldots, q_n \in D$, для которой значение $F(q_1, \ldots, q_n)$ определено. Таким образом, установлена замкнутость множества D, откуда следует, что система $U_D = \langle D; \Omega\rangle$ является подалгеброй алгебры U_A.

Если под объединением подалгебр U_{A_1} и U_{A_2} понимать подалгебру $U_{A_1 \cup A_2} = \langle [A_1 \cup A_2]; \Omega\rangle$, то совокупность всех алгебр алгебры U_A является структурой (см. § 2.5). Изучение структуры

подалгебр в универсальных алгебрах тесно связано с понятием системы образующих элементов.

Пусть $U_{A_1} = \langle A_1; \Omega\rangle$ — некоторая подалгебра универсальной алгебры $U_A = \langle A; \Omega\rangle$. Система элементов $\Sigma \subseteq A_1$ называется **системой образующих или полной системой подалгебры** U_{A_1}, если $[\Sigma] = A_1$. Если $[\Sigma] = A$, система Σ называется **системой образующих алгебры** U_A.

Теорема 3.2. *Пусть $\Sigma \subseteq A$ — некоторое непустое подмножество множества A. Тогда подалгебра $U_{[\Sigma]}$ является пересечением совокупности*

$$\{U_{A_\alpha} \mid \alpha \in I\} \tag{3.9}$$

всех подалгебр алгебры U_A таких, что $\Sigma \subseteq A_\alpha$ для каждого $\alpha \in I$.

Действительно, для любого $\Sigma \subseteq A$ совокупность подалгебр (3.9), очевидно, непуста, так как, в частности, $U_A \in \{U_{A_\alpha} \mid \alpha \in I\}$. Ввиду $\Sigma \subseteq A_\alpha$ для любого $\alpha \in I$ пересечение $D = \bigcap\limits_{\alpha \in I} A_\alpha$ непусто и $\Sigma \subseteq D$. Очевидно, вследствие $\Sigma \subseteq [\Sigma]$ подалгебра $U_{[\Sigma]}$ принадлежит совокупности (3.9), и поэтому $D \subseteq [\Sigma]$. Вместе с тем из замкнутости множества D (теорема 3.1) следует включение $[\Sigma] \subseteq D$, таким образом, $U_{[\Sigma]} = U_D$.

Универсальная алгебра $U_A = \langle A; \Omega\rangle$ (в частности, любая ее подалгебра) может иметь конечную систему образующих, тогда алгебра U_A называется **конечно-порожденной**, и может не иметь таких систем, тогда алгебра U_A называется **бесконечно порожденной.**

Пример 6. Конечно-порожденной является подалгебра $\langle N(r); \{+\}\rangle$ (см. пример 1), системой образующих этой подалгебры — $\Sigma = \{r, r+1, \ldots, 2r-1\}$. В качестве системы образующих всей алгебры $\langle N; \{+\}\rangle$ может быть любая система $\Sigma \subseteq N$ такая, что $1 \in \Sigma$.

Пример 7. Алгебра $\langle N; \{\times\}\rangle$ не имеет конечной системы образующих и бесконечно порождена. Действительно, предположим противное. Пусть Σ — некоторая конечная система образующих данной алгебры. Тогда для любых $p, q \in \Sigma$ число $p \times q$, очевидно, не принадлежит множеству $P \subseteq N$ всех простых чисел (как известно, бесконечному) и, следовательно, $[\Sigma] \not\supseteq P$, что противоречит полноте системы Σ.

Пример 8. Системой образующих алгебры $\langle F(\mathfrak{A}); \{\square\}\rangle$ может служить всякая система Σ такая, что $E \subseteq \Sigma$, где $E = \{e, a_i \mid a_i \in \mathfrak{A}, i = 1, 2, \ldots, m\}$ — множество всех однобуквенных слов (включая пустое) над алфавитом $\mathfrak{A}$.

Пример 9. Системой образующих подалгебры $\langle \mathfrak{M}_k(\mathfrak{A}); S\rangle$ (см. пример 4) является система $\Sigma = E \cup \{L_1, L_2\}$, где $L_1 = \{a_1a_2\}$ — язык над алфавитом $\mathfrak{A}$, состоящий из одного слова a_1a_2; $L_2 = \{a_1, a_2\}$ — язык над алфавитом $\mathfrak{A}$, являющийся объединением двух различных символов из $\mathfrak{A}$.

Пример 10. В двузначной алгебре Поста $\langle \Phi_{E_2}; \{\zeta, \tau, \Delta, \nabla, \not\times\}\rangle$ системами образующих являются известные полные наборы функций $\Sigma_1 = \{\bar{x}, x \vee y, x \wedge y\}$, $\Sigma_2 = \{1, x \oplus y, x \wedge y\}$ и др.

Теорема 3.3. *Пусть $U_A = \langle A; \Omega\rangle$ — конечно-порожденная алгебра. Тогда в любой системе образующих Σ алгебры U_A можно выделить конечную подсистему $\Sigma' \subseteq \Sigma$, которая также порождает алгебру U_A.*

Действительно, алгебра U_A конечно-порожденная. Это означает, что существует конечная система элементов $\Sigma_0 = \{q_1, \ldots, q_r\} \subseteq A$ такая, что $[\Sigma_0] = A$. Пусть $\Sigma = \{t_1, t_2, \ldots\} \subseteq A$ — произвольная система образующих алгебры U_A. Так как $[\Sigma] = A$, для каждого элемента $a_i \in A$ можно указать суперпозицию S_i основных операций сигнатуры Ω такую, что $a_i = S_i(t_{i_1}, t_{i_2}, \ldots, t_{i_r})$, где $t_{i_j} \in \Sigma$ при любом $j = 1, 2, \ldots, k$. Следовательно, существуют суперпозиции $S_1, S_2, \ldots, S_r$ операций сигнатуры Ω такие, что

$$\left.\begin{aligned} q_1 &= S_1(t_{11}, t_{12}, \ldots, t_{1k_1}),\\ q_2 &= S_2(t_{21}, t_{22}, \ldots, t_{2k_2}),\\ &\ldots\ldots\ldots\ldots\\ q_r &= S_r(t_{r1}, t_{r2}, \ldots, t_{rk_r}), \end{aligned}\right\} \tag{3.10}$$

где $t, j \in \Sigma$ $(j = 1, 2, \ldots, k_i;\ i = 1, 2, \ldots, r)$. Объединим элементы системы Σ, к которым применяются суперпозиции $S_1, \ldots, S_r$ (см. (3.10)): $\Sigma' = (\bigcup\limits_{i=1}^{r} \{t_{i1}, t_{i2}, \ldots, t_{ik_i}\}) \subseteq \Sigma$. Вследствие полноты системы $\Sigma_0 = \{q_1, \ldots, q_r\}$ и справедливости соотношений (3.10) полученная конечная подсистема $\Sigma' \subseteq \Sigma$ также является системой образующих алгебры U_A. Теорема доказана.

При изучении системы образующих универсальных алгебр и их подалгебр важное значение имеет установление эффективных критериев, позволяющих определить, является ли некоторая фиксированная система элементов системой образующих данной алгебры (соответственно подалгебры). Сформулированная проблема называется **проблемой полноты** для данной алгебры (соответственно подалгебры), а ее решение может быть связано с изучением так называемых максимальных подалгебр.

Пусть $U_A = \langle A; \Omega\rangle$ — произвольная универсальная алгебра (в частности, это может быть подалгебра некоторой универсальной алгебры). Собственная подалгебра $A^M \subset A$ называется **максимальной** подалгеброй относительно алгебры U_A, если не существует собственной подалгебры $A' \subset A$, для которой выполнялось бы строгое включение $A^M \subset A'$. Очевидно, подалгебра A^M тогда и только тогда максимальна относительно U_A, когда присоединение

любого элемента $q \in A \setminus A^M$ к подалгебре A^M приводит к соотношению $[A^M \cup q] = A$. Примером максимальной подалгебры относительно алгебры $\langle N; \{+\}\rangle$ является замкнутое множество $N(2) = \{2, 3, \ldots\}$. Подалгебра $\langle \mathfrak{M}_i; \{S^{n+1} \mid n = 1, 2, \ldots, m\}\rangle$, где $\mathfrak{M}_i = \mathfrak{M}_k(\mathfrak{A}) \setminus \{a_i\}$, $a_i \in \mathfrak{A}$, максимальна относительно подалгебры $\langle \mathfrak{M}_k(\mathfrak{A}); \{S^{n+1} \mid n = 1, 2, \ldots, m\}\rangle$ (см. пример 9).

Подалгебра $A' \subset A$ универсальной алгебры $U_A = \langle A; \Omega\rangle$ может быть расширена до подалгебры $A^M \subset A$, максимальной относительно U_A, если $A' \subseteq A^M$. Для конечно-порожденных алгебр справедлива следующая теорема Неймана [61].

Теорема 3.4. *Всякая подалгебра $A' \subset A$ конечно-порожденной алгебры U_A может быть расширена до некоторой подалгебры A^M, максимальной относительно U_A.*

Доказательство. Пусть $\Sigma = \{q_1, q_2, \ldots, q_r\}$ — система образующих алгебры U_A. Рассмотрим множество S всех собственных подалгебр алгебры U_A, включающих A'. Очевидно, $S \neq \varnothing$, поскольку $A' \in S$. Выберем в множестве S, частично упорядоченном по отношению включения, произвольную цепь C: $A_1 \subset A_2 \subset \cdots \subset A_s \subset \cdots$ Пусть $\tilde{A} = \bigcup_{s=1}^{\infty} A_s$. Если $\tilde{A} = A$, то $\Sigma \subset \tilde{A}$. Тогда для каждого элемента $q_i \in \Sigma$ в цепи C найдется подалгебра A_i такая, что $q_i \in A_i$. Ввиду конечности системы Σ в цепи C можно указать такой номер r_0, что $A_i \subseteq A_{r_0}$, и, следовательно, $q_i \in A_{r_0}$ при любом $i = 1, 2, \ldots, r$, т. е. $\Sigma \subset A_{r_0}$ и $A_{r_0} = A$. Но это противоречит утверждению, что $A_{r_0} \in S$ — собственная подалгебра алгебры U_A. Таким образом, $\tilde{A} \subset A$, так что A — верхняя грань произвольной цепи C из множества S и вследствие леммы Цорна (см. § 2.5) множество S имеет подалгебру A^M, которая является максимальной подалгеброй относительно алгебры U_A. Теорема доказана.

Подобно теоремам Поста [133, 134, 155], установленным для конечно-порожденных замкнутых классов в многозначных логиках, для конечно-порожденных алгебр справедлива следующая теорема.

Теорема 3.5 (критерий Поста). *Пусть M — множество всех подалгебр, максимальных относительно конечно-порожденной алгебры U_A. Система $\Sigma \subset A$ тогда и только тогда является системой образующих алгебры U_A, когда для каждой подалгебры $A^M \in M$ в системе Σ найдется по крайней мере один элемент, не принадлежащий данной подалгебре.*

Необходимость следует из замкнутости множеств $A^M \in M$, а также из строгого включения $A^M \subset A$.

Достаточность. Пусть $\Sigma \subset A$ — система элементов такая, что для каждой максимальной подалгебры $A^M \in M$ в системе Σ существует $q_k \overline{\in} A^M$. Покажем, что Σ — система образующих алгебры A. Предположим, что система Σ неполная. Тогда $[\Sigma] \neq A$ и вследствие

конечной порожденности алгебры A по доказанной выше теореме 3.4 подалгебру $[\Sigma] \subset A$ можно расширить до некоторой подалгебры $A_s^M \in M$, максимальной относительно алгебры A. В то же время по условию система Σ содержит элемент q_s такой, что $q_s \overline{\in} A_s^M$, а это противоречит включению $\Sigma \subset A_s^M$. Следовательно, предположение о том, что $[\Sigma] \neq A$, неверно и система Σ порождает алгебру A. Теорема доказана.

При проектировании автоматов (в частности, ЭВМ) систему исходных элементов необходимо выбрать так, чтобы с их помощью можно было реализовать любой автомат данного класса (предполагается, естественно, что любой элемент исходной системы имеется в неограниченном количестве экземпляров). Точно так же при создании алгоритмического языка, ориентированного на некоторый класс алгоритмов, необходимо учитывать, что с помощью основных операторов данного языка должен записываться любой из алгоритмов данного класса.

§ 3.3. Структура подалгебр универсальной алгебры

Одной из важных проблем универсальных алгебр является изучение структуры их подалгебр. Особый интерес представляют вопросы, связанные с построением структурного графа подалгебр данной универсальной алгебры (см. § 2.5). Структурный граф подалгебр универсальной алгебры представляет собой граф, вершинами которого являются подалгебры; вершина A_i соединяется с вершиной A_j дугой или отрезком, если подалгебра A_j максимальна относительно подалгебры A_i; при этом вершина A_i располагается выше вершины A_j. Примером подобного исследования служит работа [155] (см. также [134]), посвященная построению структурного графа подалгебр (диаграммы включений функционально замкнутых классов) в двузначной алгебре Поста — алгебре логики. Е. Посту удалось полностью построить структурный граф подалгебр алгебры логики. В частности, им показано, что каждая подалгебра алгебры логики имеет конечную систему образующих и множество всех подалгебр счетно. Множество всех подалгебр k-значных алгебр Поста ($k \geqslant 3$) является множеством мощности континуума. Алгебры, множества подалгебр которых являются множествами мощности не менее континуума, называются **алгебрами континуального типа**. С алгебрами континуального типа часто связано существование бесконечно порожденных подалгебр, имеющих счетный базис [139].

Система образующих $\Sigma \subseteq A$ универсальной алгебры $U_A = \langle A; \Omega\rangle$ называется **базисом** данной алгебры, если $[\Sigma \setminus a_i] \neq A$ для каждого элемента $a_i \in \Sigma$. Иными словами, система элементов Σ

составляет базис алгебры U_A, если $|\Sigma| = A$ и для любого элемента $a_i \in \Sigma$ справедливо $[\Sigma \diagup a_i] \overline{\ni} a_i$.

Пример 1. Каждая система Σ такая, что $\Sigma \ni 1$, является системой образующих алгебры $\langle N; \{+\}\rangle$. В то же время алгебра $\langle N; \{+\}\rangle$ имеет единственный базис $\Sigma = \{1\}$.

Пример 2. Система $\Sigma = \{a_i, e \mid a_i \in \mathfrak{A}\}$ является базисом в алгебре $\langle F(\mathfrak{A}); \{\boxtimes\}\rangle$.

Пример 3. Система функций $\Sigma = \{\overline{x}, x \vee y\}$ и подобные ей являются базисом в алгебре логики.

Приведенные примеры относятся к алгебрам с конечным базисом. Можно также привести примеры алгебр с бесконечным базисом.

Пример 4. Алгебра $\langle N; \{\times\}\rangle$ бесконечно порождена (см. § 3.2, пример 7). Нетрудно видеть, что множество P всех простых чисел представляет собой бесконечный базис данной алгебры. Действительно, любое число $n \in N$ представимо в виде $n = p_1^{a_1} p_2^{a_2} \dots p_k^{a_k}$, где $p_1, p_2, \dots, p_k \in P$ и, следовательно, P — система образующих алгебры $\langle N; \{\times\}\rangle$. В то же время ни одно из простых чисел нельзя представить как произведение степеней других простых чисел. Таким образом, множество P образует бесконечный базис алгебры $\langle N; \{\times\}\rangle$.

Из теоремы 3.3 (§ 3.2) вытекает непосредственно следующая теорема.

Теорема 3.6. *Всякая конечно-порожденная алгебра имеет конечный базис.*

Действительно, пусть $U_A = \langle A; \Omega\rangle$ — конечно-порожденная универсальная алгебра. Тогда в произвольной системе образующих Σ алгебры U_A вследствие теоремы 3.3 можно выделить конечную подсистему $\Sigma' \subset \Sigma$, которая также является системой образующих данной алгебры. Если система Σ' не содержит ни одного элемента $q_{i_1} \in \Sigma'$ такого, что $q_{i_1} \in [\Sigma' \setminus q_{i_1}]$, то Σ' образует базис алгебры U_A. В противном случае система $\Sigma'' = \Sigma' \setminus q_{i_1}$ является системой образующих алгебры U_A. Далее, если система Σ'' не содержит ни одного элемента $q_{i_2} \in \Sigma''$ такого, что $q_{i_2} \in [\Sigma'' \setminus q_{i_2}]$, то Σ'' — базис алгебры U_A. В противном случае перейдем к системе $\Sigma''' = \Sigma'' \setminus q_{i_2}$. Ввиду конечности системы Σ' этот процесс завершится построением базиса алгебры U_A. Теорема доказана.

Можно привести примеры универсальных алгебр, и в частности подалгебр, не имеющих базиса. В любую систему образующих алгебры без базиса входят элементы, исключение которых не нарушает полноты данной системы.

Пример 5. Не имеет базиса алгебра $\langle N; \{\odot\}\rangle$ с основным множеством N всех натуральных чисел и определенной на данном множестве унарной операцией $\odot$ такой, что $1^{\odot} = 1$ и для любого $n \in N \setminus 1$ $n^{\odot} = n - 1$. Очевидно, что любой начальный отрезок $\{1, 2, \dots, n\}$ образует подалгебру данной алгебры. В то же время

любое бесконечное подмножество $N' \subset N$ является системой образующих данной алгебры. Причем в любой такой системе всегда найдется по крайней мере один (а следовательно, и бесконечно много) элемент $q \in N'$ такой, что $[N' \setminus q] \ni q$. Таким образом, алгебра $\langle N; \{\odot\}\rangle$ не имеет базиса.

Следствие. Любая алгебра, не имеющая базиса, бесконечно порождена.

Пусть, далее, цепь подалгебр алгебры A

$$C: A_1 \subset \dots \subset A_i \subset \dots \tag{3.11}$$

такая, что $A = \bigcup_i A_i$; C удовлетворяет **условию плотности,** если для любой подалгебры $A' \subset A$ найдется номер n, для которого $A' \subseteq A_n$.

Теорема 3.7. *Если цепь C плотна и для каждого i найдется конечная система $\Sigma \subseteq A_{i+1} \setminus A_i$ такая, что $A_i \subset [\Sigma_i]$, то алгебра A не имеет базиса.*

Очевидно, что A не имеет конечного базиса Σ, так как в противном случае вследствие монотонного возрастания последовательности (3.11) найдется n такое, что $\Sigma \subset A_n$, и поэтому $A_n = A$, что противоречит бесконечности последовательности (3.11). Пусть Σ — бесконечный базис алгебры A. Выделим в Σ некоторую бесконечную подсистему $\Sigma' \subset \Sigma$. Очевидно, что $[\Sigma'] \neq A$, так как Σ' — собственное подмножество базиса Σ. Вследствие плотности цепи C можно указать такое n, что $[\Sigma'] \subset A_n$. По условию для A_n найдется конечная система элементов $\Sigma_n \subset A_{n+1}$ такая, что $[\Sigma_n] \supseteq A_n$. Но Σ — полная система и, следовательно, в Σ можно выделить конечную подсистему $\Sigma'' \subset \Sigma$ такую, что $[\Sigma''] \supset \Sigma_n$. Тогда из $[\Sigma_n] \supset A_n$, $[\Sigma''] \supset \Sigma'$ и из бесконечности множества Σ' следует, что найдется по крайней мере один элемент $q_i \in \Sigma$ такой, что $q_i \overline{\in} \Sigma''$, причем $q_i \in [\Sigma'']$, а это противоречит предположению, что Σ — базис алгебры. Теорема доказана.

Как показывает следующая теорема, универсальная алгебра является алгеброй континуального типа, если она имеет хотя бы одну подалгебру с бесконечным базисом.

Теорема 3.8. *Пусть алгебра $U_A = \langle A; \Omega\rangle$ имеет подалгебру U_{A_1} с бесконечным счетным базисом. Тогда мощность множества всех подалгебр алгебры U_A не меньше континуальной.*

Действительно, пусть $\Sigma = \{q_1, q_2, \dots, q_s, \dots\}$ — бесконечный счетный базис подалгебры U_{A_1} алгебры U_A. Пусть, далее, $\Sigma' \subset \subset \Sigma$ — произвольное подмножество элементов базиса. Рассмотрим подалгебру $[\Sigma']$, порожденную системой Σ'. По определению базиса для любого элемента $a_i \in (\Sigma \setminus \Sigma')$ справедливо $a_i \overline{\in} [\Sigma']$. Таким образом, различные подсистемы Σ', $\Sigma'' \subset \Sigma$ определяют различные подалгебры $[\Sigma']$ и $[\Sigma'']$. Следовательно, мощность множества всех

подалгебр алгебры U_A не меньше мощности множества всех подмножеств счетной совокупности Σ. Но множество всех подмножеств счетного множества континуально (см. § 1.4). Теорема доказана.

Таким образом, всякая алгебра, имеющая хотя бы одну подалгебру со счетным базисом, является алгеброй континуального типа. К последним, в частности, относятся k-значные алгебры Поста ($k \geqslant 3$), а также некоторые их модификации, связанные с преобразованием на регистрах (см. § 4.3).

Следствие. Алгебрами континуального типа являются алгебра $\langle \mathfrak{M}_k(\mathfrak{A}); \{S^{n+1} \mid n = 1, 2, \ldots, m\}\rangle$ всех конечных языков и алгебра $\langle \mathfrak{M}(\mathfrak{A}); \{S^{n+1} \mid n = 1, 2, \ldots, m\}\rangle$ всех языков над алфавитом $\mathfrak{A} = \{a_1, a_2, \ldots, a_n\}$.

Действительно, выделим в алфавите $\mathfrak{A}$ элемент $a_i \in \mathfrak{A}$ и рассмотрим множество $\{L_p \mid p \in P\}$ всех одноэлементных языков $L_p = \{s = a_i a_i \ldots a_i\}$, где $|s| = p$; $p \in P$ — некоторое простое число. Очевидно, что замыкание совокупности $\Sigma = \{L_p \mid p \in P\}$ образует подалгебру, изоморфную алгебре $\langle N; \{\times\}\rangle$, которая, как было показано выше (см. пример 4), имеет бесконечный базис. Следовательно, алгебра $\mathfrak{M}_k(\mathfrak{A})$, а значит, и алгебра $\mathfrak{M}(\mathfrak{A})$ являются алгебрами континуального типа.

При изучении структуры подалгебр важно установить критерии бесконечной порожденности, а также существования базиса универсальных алгебр. Эти вопросы для некоторых классов универсальных алгебр решены в работе [119]. Универсальная алгебра $U_A = \langle A; \Omega\rangle$ называется **плотной**, если любая собственная подалгебра $A' \subset A$ может быть расширена до некоторой подалгебры, максимальной относительно U_A, либо если при $A = \bigcup_i A_i$ последовательность $A_1 \subset \cdots \subset A_i \subset \cdots$ образует цепь, удовлетворяющую условию плотности (т. е. для всякой подалгебры $A' \subset A$ найдется номер n такой, что $A' \subseteq A_n$).

Теорема 3.9. *Пусть плотная алгебра A' имеет базис. Тогда любая ее собственная подалгебра U_A может быть расширена до некоторой подалгебры, максимальной относительно U_A.*

Из данной теоремы следует, в частности, теорема 3.4 (§ 3.2). Кроме того, для плотной алгебры U_A справедливы следствия.

Следствие 1. Алгебра U_A не имеет базиса, если существует по крайней мере одна собственная подалгебра $A' \subset A$, не допускающая расширения до подалгебры, максимальной относительно U_A.

Следствие 2. Алгебра U_A с бесконечным базисом имеет бесконечную совокупность максимальных подалгебр.

Следствие вытекает из того, что для любого $q \in \Sigma$, где Σ — бесконечный базис алгебры U_A, множество $[\Sigma \setminus q]$ может быть расширено до некоторой подалгебры, максимальной относительно U_A.

Для плотных алгебр с базисом справедлив критерий Поста (теорема 3.5).

Следствие 3. Система $\Sigma \subset A$ тогда и только тогда является системой образующих алгебры U_A, имеющей базис, когда для каждой подалгебры $A^M \in M$ в системе Σ найдется по крайней мере один элемент, не принадлежащий подалгебре A^M (M — множество всех максимальных подалгебр алгебры U_A).

Для плотных алгебр с базисом справедлив следующий критерий конечной порожденности.

Теорема 3.10. *Пусть U_A — плотная алгебра с базисом. Алгебра U_A имеет конечный базис тогда и только тогда, когда для множества $M = \{A^M_i \mid i \in I\}$ всех подалгебр, максимальных относительно U_A, существует конечное разбиение $R(I) = (I_1, I_2, \ldots, I_k)$ такое, что для любого $r = 1, 2, \ldots, k$*

$$\bigcap_{i_r \in I_r} \bar{A}^M_{i_r} \neq \varnothing, \tag{3.12}$$

где $\bar{A}^M = A \setminus A^M$.

Необходимость. Пусть U_A — алгебра с конечным базисом $\Sigma = \{q_1, q_2, \ldots, q_k\}$. Покажем, что тогда для множества M существует конечное разбиение $R(I)$, для которого выполняется соотношение (3.12). Для каждого элемента $q_r \in \Sigma$ рассмотрим множество $M(q_r) = \{A^M_{S_r} \mid S_r = 1, 2, \ldots\}$ всех подалгебр, максимальных относительно U_A и таких, что $q_r \bar{\in} A^M_{S_r}$ ($r = 1, 2, \ldots, k$). Так как Σ — система образующих алгебры U_A, то $\bigcup_{r=1} M(q_r) = M$. Кроме того, $M(q_1) \not\supseteq M(q_2)$, в противном случае множество $\Sigma \setminus q_2$ является системой образующих алгебры U_A, а это противоречит предположению, что Σ — базис данной алгебры. На этом же основании $(M(q_1) \cup M(q_2)) \not\supseteq M(q_3)$, $\ldots$, $(\bigcup_{s=1}^{k=1} M(q_s)) \not\supseteq M(q_k)$. Построим следующие классы подалгебр, максимальных относительно U_A:

$$M_1 = \{A^M_{i_1} \mid i_1 \in I_1\} = M(q_1),$$
$$M_2 = \{A^M_{i_2} \mid i_2 \in I_2\} = M(q_2) \setminus M(q_1),$$
$$M_3 = \{A^M_{i_3} \mid i_3 \in I_3\} = M(q_3) \setminus (M(q_1) \cup M(q_2)),$$
$$\ldots\ldots\ldots\ldots\ldots\ldots\ldots$$
$$M_k = \{A^M_{i_k} \mid i_k \in I_k\} = M(q_k) \setminus (\bigcup_{s=1}^{k-1} M(q_s)).$$

Из этого построения вытекает, что $M_i \cap M_j = \varnothing$ при $i \neq j$ (при любых $i, j = 1, 2, \ldots, k$). Кроме того, $M = \bigcup_{r=1}^{k} M_r$. Следовательно, существует конечное разбиение $R(I) = \{I_1, I_2, \ldots, I_k\}$, для которого выполняется соотношение (3.12).

Достаточность. Пусть для множества M существует конечное разбиение $R(I) = \{I_1, I_2, \ldots, I_k\}$, для которого выполняется соотношение (3.12). Тогда алгебра U_A имеет конечный базис.

Действительно, для совокупности $M_r = \{A^M_{i_r} | i_r \in I_r\}$ существует по крайней мере один элемент $q_r \in A^M_{i_r}$ при любом $i_r \in I_r$ ($r = 1, 2, \ldots, k$). В силу соотношения $M = \bigcup_{r=1}^{k} M_r$, а также теоремы 3.9 (следствие 3) система $\Sigma = \{q_1, q_2, \ldots, q_k\}$ является системой образующих алгебры U_A. Теорема доказана.

Обратная теорема устанавливает критерий бесконечной порожденности для плотных алгебр.

Изучение структуры подалгебр в алгебрах континуального типа связано с серьезными трудностями, первым этапом в преодолении которых является построение верхнего и нижнего фрагментов структурного графа подалгебр — поверхности алгебры и ее основания соответственно.

Подалгебра A алгебры $U_B = \langle B; \Omega \rangle$ называется **достижимой сверху** в структурном графе подалгебр алгебры U_B, если существует по крайней мере одна монотонно убывающая последовательность конечно-порожденных подалгебр $A_1 \supset A_2 \supset \cdots$ такая, что $A = \bigcap_i A_i$, где $A_1 = B$ (совпадает со всей алгеброй), и при каждом $i = 1, 2, \ldots$ подалгебра A_i максимальна относительно A_{i-1}, либо $A_i = \bigcap_{\alpha < i} A_\alpha$. Множество всех достижимых сверху конечно-порожденных подалгебр алгебры U_B называется **поверхностью** этой алгебры.

Двойственным образом определяются подалгебра, достижимая снизу, и **основание** алгебры U_B. Подалгебра A называется **достижимой снизу** в структурном графе подалгебр алгебры U_B, если существует монотонно возрастающая последовательность конечно-порожденных подалгебр $A_1 \subset A_2 \subset \cdots$ такая, что $A = \bigcup_i A_i$, где A_1 — один из минимальных элементов структуры подалгебр алгебры U_B (см. § 2.5), и при каждом $i = 1, 2, \ldots$ подалгебра A_i максимальна относительно A_{i+1}, либо $A_i = \bigcup_{\alpha < i} A_\alpha$. Множество всех достижимых снизу конечно-порожденных подалгебр алгебры U_B называется **основанием** данной алгебры.

Если каждая подалгебра U_B конечно-порожденная, то поверхность данной алгебры совпадает с ее основанием и полностью представляет структурный граф данной алгебры. Так, поверхность (основание) двузначной алгебры Поста совпадает со всем структурным графом (диаграммой включений) подалгебр данной алгебры. Если алгебра U_B имеет бесконечно порожденные подалгебры, построение поверхности (основания) данной алгебры

позволяет изучить ограничивающие эту поверхность (основание) бесконечно порожденные подалгебры.

Пример 6. Рассмотрим структуру подалгебр, входящих в подалгебру O_9 — всех одноместных функций и констант двузначной алгебры Поста. Поверхность подалгебры O_9 является частью всей поверхности рассматриваемой алгебры и содержит конечное число (9) подалгебр:

$$O_9 = \{0, 1, x, \bar{x}\}, \quad O_8 = \{0, 1, x\}, \quad O_7 = \{0, 1\},$$
$$O_6 = \{0, x\}, \quad O_5 = \{1, x\}, \quad O_4 = \{x, \bar{x}\},$$
$$O_3 = \{0\}, \quad O_2 = \{1\}, \quad O_1 = \{x\}.$$

Каждая из перечисленных подалгебр, очевидно, конечно-порожденная. Поэтому поверхность подалгебры O_9 совпадает с ее основанием и полностью представляет структурный граф подалгебр, входящих в O_9 (рис. 14).

Пример 7. Алгебра $\langle N; \{\odot\} \rangle$ бесконечно порождена (см. пример 5) и, следовательно, не имеет поверхности. Основание этой алгебры представляет собой структурный граф, показанный на рис. 15.

Структура подалгебры данной алгебры имеет единственный минимальный элемент — подалгебру $N_1 = \{1\}$. Каждый начальный отрезок $N_m = \{1, 2, \ldots, m\}$ образует подалгебру алгебры $\langle N; \{\odot\} \rangle$, причем подалгебра N_m максимальна относительно подалгебры N_{m+1} при любом $m = 1, 2, \ldots$ Вся алгебра $\langle N; \{\odot\} \rangle = \bigcup_{i=1}^{\infty} N_i$ достижима снизу и является единственной бесконечно порожденной несобственной подалгеброй.

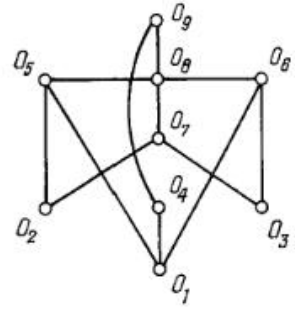


Рис. 14.

Рис. 15.

§ 3.4. Функции алгебры логики

В настоящем параграфе рассмотрены элементы теории булевых функций или **функций алгебры логики**, используемые при проектировании ЭВМ и программировании (см. гл. 4).

Рассмотрим всюду определенные булевы функции, т. е. функции, принимающие значение 1 или 0 на каждом из своих двоичных наборов (см. § 2.3). Поскольку каждая переменная может принимать лишь два значения, любая булева функция имеет конечную область определения.

Область определения n-местной булевой функции состоит из 2^n наборов значений переменных. Конечность области определения и области значения произвольной булевой функции позволяет задавать функцию с помощью таблиц истинности. Двоичные наборы значений переменных записываются как некоторые целые числа в двоичной системе счисления. Набор $\tilde{\alpha} = (a_1, a_2, \ldots, a_n)$ отождествляется с записью числа

$$a_1 \cdot 2^{n-1} + a_2 \cdot 2^{n-2} + \cdots + a_{n-1} \cdot 2 + a_n.$$

Это число называется **номером соответствующего набора.** Так, для четырехместной булевой функции номером набора 0101 является число

$$0 \cdot 2^3 + 1 \cdot 2^2 + 0 \cdot 2 + 1 = 5.$$

Номера наборов n-местной булевой функции изменяются от нуля до $2^n - 1$. Расположив наборы в столбец в порядке роста их номеров и указав значения функции на каждом наборе, получим **таблицу истинности** булевой функции (табл. 1).

Таблица 1

$x_1x_2 \ldots x_{n-1}x_n$	$f(x_1, x_2, \ldots, x_{n-1}, x_n)$
00 ... 00	$f(0, 0, \ldots, 0, 0)$
00 ... 01	$f(0, 0, \ldots, 0, 1)$
00 ... 10	$f(0, 0, \ldots, 1, 0)$
00 ... 11	$f(0, 0, \ldots, 1, 1)$
...	...
11 ... 11	$f(1, 1, \ldots, 1, 1)$

Булевыми функциями являются логические операции: дизъюнкция $\vee$, конъюнкция $\wedge$ (или $\odot$), отрицание — (см. § 1.2). К другим известным булевым функциям относятся импликация $x_1 \to x_2$, эквивалентность $x_1 \equiv x_2$, сложение (по $\bmod 2$) $x_1 \oplus x_2$, обратная импликация $x_1 \leftarrow x_2$, прямая антиимпликация $x_1 \not\to x_2$, обратная антиимпликация $x_1 \not\leftarrow x_2$, штрих Шеффера $x_1 \mid x_2$, стрелка Пирса $x_1 \downarrow x_2$. Эти функции определены в табл. 2.

Таблица 2

x	$\bar{x}$	x_1	x_2	$x_1 \cdot x_2$	$x_1 \vee x_2$	$x_1 + x_2$	$x_1 \equiv x_2$	$x_1 \oplus x_2$	$x_1 \leftarrow x_2$	$x_1 \not\to x_2$	$x_1 \not\leftarrow x_2$	$x_1 \mid x_2$	$x_1 \downarrow x_2$
0	1	0	0	0	0	1	1	0	1	0	0	1	1
1	0	0	1	0	1	1	0	1	0	0	1	1	0
		1	0	0	1	0	0	1	1	1	0	1	0
		1	1	1	1	1	1	0	1	0	0	0	0

В связи с тем что на каждом наборе булева функция может принимать одно из двух значений (1 или 0), независимо от значений, которые она принимает на остальных наборах, существуют 2^{2^n} различных n-местных булевых функций.

Заметим, что к n-местным булевым функциям принадлежат функции $f(x_1, x_2, \ldots, x_n)$, существенно не зависящие от некоторых переменных. Следовательно, всякая функция $f(x_1, x_2, \ldots, x_k)$ включается в число функций $f(x_1, x_2, \ldots, x_n)$, где $n > k$. Так, константы, которые естественно считать нульместными булевыми функциями, можно рассматривать как функции одноместные, двуместные и т. д. С ростом числа переменных быстро увеличивается число зависящих от них булевых функций.

Таблицы истинности булевых функций с ростом числа аргументов становятся громоздкими и неудобными. Более удобный аналитический способ задания булевых функций основан на рассмотрении алгебры Φ_{E_2} с операцией суперпозиции над множеством булевых функций — двузначной алгебры Поста (см. § 3.1, пример 10). Зафиксировав в качестве системы образующих алгебры Φ_{E_2} булевый набор операций $(\vee, \cdot, -)$, произвольную булеву функцию можно представить как суперпозицию этих булевых операций. Нетрудно убедиться в том, что операции $\vee$, $\cdot$, — удовлетворяют всем аксиомам булевой алгебры (см. § 2.5, а также § 1.2).

Одной из интересных систем образующих алгебры Φ_{E_2} является набор Жегалкина $(\oplus, \cdot, 1)$. Операции $\oplus$, $\cdot$ ассоциативны и коммутативны, кроме того, для $\oplus$ выполняются следующие соотношения:

$$x \cdot (y \oplus z) = x \cdot y \oplus x \cdot z, \tag{3.13}$$

$$x \oplus x = 0, \tag{3.14}$$

$$x \oplus 0 = x. \tag{3.15}$$

Для указанных наборов операций разработаны канонические формы представлений булевых функций. Зафиксируем перечень переменных $X = \{x_1, x_2, \ldots, x_n\}$. Конъюнкция переменных из набора X или их отрицаний, т. е. конъюнкция вида $\tilde{x}_{i_1} \cdot \tilde{x}_{i_2} \cdot \ldots \cdot \tilde{x}_{i_k}$, где $\tilde{x}_{i_j} = \begin{cases} \bar{x}_{i_j} \\ x_{i_j} \end{cases} (j = 1, 2, \ldots, k)$, называется **элементарной,** если в ней каждая буква встречается не более одного раза. К элементарным конъюнкциям относятся также выражения, состоящие из одной буквы (с отрицанием или без отрицания), — одноместные элементарные конъюнкции. Константу 1 можно рассматривать как нульместную элементарную конъюнкцию. Так, элементарными конъюнкциями являются 1, $\bar{x}_1$, $x_1 \cdot x_2$, $\bar{x}_1 \cdot \bar{x}_2 \cdot x_3$, а выражения $\overline{x_1 \cdot x_2}$, $x_1 \cdot x_2 \cdot x_1$, $x_1 \cdot \bar{x}_1$ по определению ими не являются.

n-Местная элементарная конъюнкция, включающая все переменные из набора X, называется **конституэнтой 1.** Нетрудно видеть, что число всех (различных) конституэнт 1 для фиксированного набора из n переменных равно 2^n.

Дизъюнкция $\mathfrak{K}$-элементарных конъюнкций U_i,

$$\mathfrak{K} = U_1 \vee U_2 \vee \cdots \vee U_m,$$

называется **дизъюнктивной нормальной формой** (д. н. ф.). Д. н. ф., каждый дизъюнктивный член которой является конституэнтой 1, называется **совершенной дизъюнктивной нормальной формой** (с. д. н. ф). При этом не исключается случай одноместной дизъюнкции (д. н. ф. совпадает с некоторой U_i). Нульместная дизъюнкция принимается тождественно равной нулю.

Двойственным образом (см. § 2.5), заменяя в приведенных определениях нули единицами и наоборот, дизъюнкции — конъюнкциями и наоборот, определяем понятия элементарной дизъюнкции, конституэнты 0, **конъюнктивной нормальной формы** (к. н. ф.), **совершенной конъюнктивной нормальной формы** (с. к. н. ф.).

Теорема 3.11. *Всякая булева функция $f(x_1, x_2, \ldots, x_n) \not\equiv 0$ может быть однозначно представима в совершенной дизъюнктивной нормальной форме.*

Доказательство. Пусть задана некоторая функция $f(x_1, x_2, \ldots, x_n) \not\equiv 0$. Каждому двоичному набору $\tilde{\alpha} = (a_1, a_2, \ldots, a_n)$ соответствует единственная конституэнта 1 $\tilde{x}_1 \cdot \tilde{x}_2 \cdot \ldots \cdot \tilde{x}_n$, обращающаяся на этом наборе в единицу. Действительно, такая конституэнта определяется следующим образом:

$$\tilde{x}_i = \begin{cases} x_i, & \text{если } a_i = 1, \\ \bar{x}_i, & \text{если } a_i = 0 \end{cases} \quad (i = 1, 2, \ldots, n).$$

Все остальные конституэнты 1 на данном наборе обращаются в нуль. Например, для функции, зависящей лишь от переменных x_1, x_2, x_3, набору (0, 1, 0) соответствует конституэнта 1 — $\bar{x}_1 x_2 \bar{x}_3$, а набору (1, 1, 0) — конституэнта 1 — $x_1 x_2 \bar{x}_3$.

Образуя дизъюнкцию конституэнт 1, соответствующих всем наборам, на которых функция f обращается в единицу, получаем с. д. н. ф., равную данной функции. Причем функция f однозначно определяет соответствующую с. д. н. ф., которую вследствие этой однозначности принято называть **с. д. н. функции f**. Теорема доказана.

Ввиду того что $x \cdot \bar{x} = 0$, справедливо такое следствие.

Следствие. Любая функция может быть представлена в виде суперпозиции булевых операций.

Пример. Построим с. д. н. ф. функции $f(x, y, z)$, заданной таблицей истинности (табл. 3). Конституэнты 1, соответствующие наборам, на которых данная функция обращается в единицу, имеют вид $\bar{x}yz$, $x\bar{y}z$, $xy\bar{z}$, xyz. Таким образом,

$$f(x, y, z) = xyz \vee xy\bar{z} \vee x\bar{y}z \vee \bar{x}yz. \qquad (3.16)$$

Заметим, что произвольная функция может быть представлена в классе д. н. ф. различными способами. Так, для приведенного

выше примера наряду с равенством (3.16) выполняется также равенство

$$f(x, y, z) = xy \vee xz \vee yz. \qquad (3.17)$$

Таблица 3

x	y	z	$f(x, y, z)$
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

Поэтому в результате применения теории булевых функций при синтезе комбинационных схем возникла проблема нахождения минимальной по числу букв д. н. ф. для произвольной булевой функции (**проблема минимизации**) [24, 49, 112, 133].

В силу принципа двойственности все результаты, относящиеся к представлению булевых функций в классе д. н. ф., переносятся на к. н. ф.

Пусть функция $f \not\equiv 0$ представлена с. д. н. ф.

$$f(x_1, \ldots, x_n) = K_1 \vee K_2 \vee \ldots \vee K_r, \qquad (3.18)$$

где K_i — конституэнты 1, соответствующие наборам, на которых f принимает значение 1 ($i = 1, 2, \ldots, r$). Используя соотношения

$$\bar{x} = x \oplus 1, \quad x \vee y = xy \oplus x \oplus y, \qquad (3.19)$$

можно устранить в (3.18) все отрицания и дизъюнкции. Раскрывая затем с помощью (3.13), (3.14) скобки и выполняя приведение подобных членов, получаем представление функции f в базисе Жегалкина с помощью полинома Жегалкина. Таким образом, справедлива следующая теорема.

Теорема 3.12. *Каждая булева функция может быть представлена в форме полинома Жегалкина.*

Более подробно различные формы представления булевых функций рассматриваются в работе [24].

Системы образующих алгебры Φ_{E_2} называются **функционально полными системами** [133, 134]. В частности, функционально полными являются рассмотренные выше булева система операций и базис Жегалкина.

Заметим, что система $\{\vee, \cdot, -\}$ не является базисом в алгебре Φ_{E_2}. Действительно, ввиду того что

$$x \vee y = \overline{\bar{x}\bar{y}}, \qquad (3.20)$$

$$xy = \overline{\bar{x} \vee \bar{y}}, \qquad (3.21)$$

из булевой системы операций можно исключить дизъюнкцию или конъюнкцию так, что системы $\{\vee, -\}$ и $\{\cdot, -\}$ также будут функционально полными.

Таким образом, алгебра Φ_{E_2} конечно-порожденная и на основании теоремы 3.5 (§ 3.2) для нее справедлив критерий Поста,

обеспечивающий (в терминах максимальных подалгебр) необходимые и достаточные условия, при которых произвольная система $\Sigma \subset \Phi_{E_2}$ является системой образующих данной алгебры.

Множество всех максимальных подалгебр (предполных замкнутых классов) алгебры логики впервые было описано Е. Постом [155] (см. также [134]). Рассмотрим эти подалгебры и основную идею доказательства теоремы Поста о полноте.

Булевой функцией, сохраняющей константу 0, называется функция $f(x_1, x_2, \ldots, x_n)$ такая, что $f(0, 0, \ldots, 0) = 0$. Например, сохраняют константу 0 дизъюнкция и конъюнкция, а отрицание и импликация не сохраняют ее (см. табл. 2).

Булевой функцией, сохраняющей константу 1, называется функция $f(x_1, x_2, \ldots, x_n)$ такая, что $f(1, 1, \ldots, 1) = 1$. Сохраняют константу 1, например, конъюнкция, дизъюнкция, тогда как отрицание и сумма (по mod 2) не сохраняют ее (см. табл. 2).

Функция $f(x_1, x_2, \ldots, x_n)$ называется **самодвойственной,** если $f(x_1, x_2, \ldots, x_n) = \overline{f}(\bar{x}_1, \bar{x}_2, \ldots, \bar{x}_n)$. Иными словами, самодвойственная функция f на противоположных наборах (не совпадающих по каждой компоненте) принимает противоположные значения. Примерами самодвойственной функции являются отрицание (все остальные функции, заданные табл. 2, не самодвойственны) и функция $f(x, y, z)$ (см. табл. 3).

Введем отношение частичного порядка в множестве всех наборов, на которых определена n-местная функция $f(x_1, x_2, \ldots, x_n)$. Положим $0 \prec 1$. Тогда для наборов $\tilde{\alpha} = (a_1, a_2, \ldots, a_n)$ и $\tilde{\beta} = (b_1, b_2, \ldots, b_n)$ $\tilde{\alpha} \preccurlyeq \tilde{\beta}$, если $a_i \preccurlyeq b_i$ при любом $i = 1, 2, \ldots, n$. Например, $(1, 0, 0) \preccurlyeq (1, 0, 1)$.

Монотонной булевой функцией называется функция $f(x_1, x_2, \ldots, x_n)$ такая, что для любых наборов значений переменных $\tilde{\alpha}$ и $\tilde{\beta}$, где $\tilde{\alpha} \preccurlyeq \tilde{\beta}$, $f(\tilde{\alpha}) \preccurlyeq f(\tilde{\beta})$ (под $f(\tilde{\alpha})$ понимается $f(a_1, a_2, \ldots, a_n)$). Монотонны, например, дизъюнкция, конъюнкция, тогда как отрицание и сумма (по mod 2) немонотонны.

Линейной булевой функцией называется функция $f(x_1, x_2, \ldots, x_n)$, представимая полиномом Жегалкина первой степени, т. е. принимающая такой вид:

$$f(x_1, x_2, \ldots, x_n) = a_0 \oplus a_1 x_1 \oplus a_2 x_2 \oplus \cdots \oplus a_n x_n,$$

где a_i — коэффициенты, равные нулю или единице ($i = 0, 1, \ldots, n$). Например, отрицание и сумма (по mod 2) линейны, а конъюнкция и дизъюнкция (см. (3.19)) нелинейны.

Нетрудно убедиться, что указанные классы булевых функций замкнуты относительно суперпозиции и, следовательно, образуют подалгебры алгебры Φ_{E_2}, максимальные относительно Φ_{E_2}, и алгебра логики не имеет других максимальных подалгебр. Справедливость этого утверждения следует из теоремы Поста [133].

Теорема 3.13 (Критерий функциональной полноты). *Система функционально полна тогда и только тогда, когда содержит*

1) хотя бы одну функцию, не сохраняющую константу 0,
2) хотя бы одну функцию, не сохраняющую константу 1,
3) хотя бы одну несамодвойственную функцию,
4) хотя бы одну немонотонную функцию,
5) хотя бы одну нелинейную функцию.

Необходимость следует из того, что каждый из классов в пяти перечисленных условиях образует собственную подалгебру алгебры Φ_{E_2}.

Доказательство *достаточности* состоит в сведении системы, удовлетворяющей условиям теоремы, к некоторой функционально полной системе (например, $\{\vee, -\}$ или $\{\cdot, -\}$).

Использовав условия 1—3 теоремы, построим из выбранных функций константы 1 и 0. Пусть $f^{(0)}(x_1, x_2, \ldots, x_n) \in \Sigma$ — функция, не сохраняющая константу 1. Отождествив все переменные функции $f^{(0)}$, получим

$$h(x) = f^{(0)}(x, \ldots, x) = \begin{cases} \bar{x}, & \text{если } f^{(0)}(0, 0, \ldots, 0) = 1, \\ 0, & \text{если } f^{(0)}(0, 0, \ldots, 0) = 0. \end{cases}$$

При $h(x) = 0$, использовав функцию $f^{(1)} \in \Sigma$, не сохраняющую константу 0, получим константу 1. Если $h(x) = \bar{x}$, выберем несамодвойственную функцию $f^{(\tilde{c})}(x_1, x_2, \ldots, x_m) \in \Sigma$. Тогда найдется пара противоположных наборов $\tilde{\alpha} = (a_1, a_2, \ldots, a_m)$ и $\tilde{\alpha}' = (\bar{a}_1, \bar{a}_2, \ldots, \bar{a}_m)$ таких, что $f^{(\tilde{c})}(\tilde{\alpha}) = f^{(\tilde{c})}(\tilde{\alpha}')$.

Разобьем переменные функции $f^{(\tilde{c})}$ на две группы: к 1-й отнесем все x_i, для которых значения a_i в наборе $\tilde{\alpha}$ равны нулю, ко 2-й — все остальные переменные. Подставив в $f^{(\tilde{c})}$ вместо каждой переменной 1-й группы функцию $h(x) = \bar{x}$, а вместо переменной 2-й группы — переменную x, получим одну из констант $f^{(\tilde{c})}(\bar{x}, \ldots, \bar{x}) = t$, где $t \in \{0, 1\}$. Другую константу можно получить, используя функцию, не сохраняющую t.

Далее, выбрав немонотонную функцию $f^{(\tilde{m})}(x_1, x_2, \ldots, x_r) \in \Sigma$, можно построить отрицание. Действительно, в силу немонотонности для функции $f^{(\tilde{m})}$ найдется пара наборов $\tilde{\alpha} = (a_1, a_2, \ldots, a_r)$ и $\tilde{\beta} = (b_1, b_2, \ldots, b_r)$ таких, что $\tilde{\alpha} \preccurlyeq \tilde{\beta}$, причем $f(\alpha) \succ f(\beta)$. Проверяя значения функции f на промежуточных наборах γ (их конечное число), где $\tilde{\alpha} \preccurlyeq \tilde{\gamma} \preccurlyeq \tilde{\beta}$, можно выделить пару соседних наборов $\tilde{\alpha}_1 = (a_1^1, a_2^1, \ldots, a_{(i-1)}^1, 0, a_{(i+1)}^1, \ldots, a_r^1)$ и $\tilde{\alpha}_2 = (a_1^1, a_2^1, \ldots, a_{(i-1)}^1, 1, a_{(i+1)}^1, \ldots, a_r^1)$ таких, что $\tilde{\alpha} \preccurlyeq \tilde{\alpha}_1 \prec \tilde{\alpha}_2 \preccurlyeq \tilde{\beta}$, причем $f^{(\tilde{m})}(\tilde{\alpha}_1) \succ f^{(\tilde{m})}(\tilde{\alpha}_2)$. Подставив в функцию $f^{(\tilde{m})}$ вместо переменных x_j, для которых $a_j^1 = 0$, константу 0, а вместо переменных x_j, для которых $a_j^1 = 1$, — константу 1, при любых $j \neq i$, и, отождествив

переменную $x_j \equiv x$, получим $f^{(\overline{m})}(a_1^1, a_2^1, \ldots, a_{j-1}^1, x, a_{j+1}^1, \ldots, a_r^1) = \bar{x}$. Наконец, выберем нелинейную функцию $f^{(\overline{n})}(x_1, \ldots, x_s) \in \Sigma$, представленную полиномом Жегалкина, и выделим в ней нелинейный одночлен P, наименьший по длине. Подставив в функцию $f^{(\overline{n})}$ вместо всех переменных, не входящих в P, константу 0 и отождествив часть переменных одночлена из P с переменной x, а остальные — с переменной y, получим функцию $g(x, y) = xy \oplus ax \oplus by \oplus c$. Ввиду наличия отрицания можно считать, что $c = 0$. Если $a = b$, то на основании (3.19)

$$g(x, y) = \begin{cases} x \vee y, & \text{если } a = b = 1, \\ xy, & \text{если } a = b = 0. \end{cases}$$

С помощью построенной ранее функции $\bar{x}$ исходную систему Σ удалось свести к функционально полной системе $\{\vee, -\}$ или $\{\cdot, -\}$.

Рассмотрим случай $a \neq b$. Пусть для определенности $a = 0$, $b = 1$. Тогда $g(x, y) = xy \oplus y$ и, перейдя к суперпозиции $g(\bar{x}, y) = \bar{x}y \oplus y = (x \oplus 1)y \oplus y = xy \oplus y \oplus y = xy$, сведем систему Σ также к функционально полной системе $\{\cdot, -\}$. Теорема доказана.

Следствие (ослабленная теорема о функциональной полноте) [24]. Для полноты системы булевых функций, включающей константы 0 и 1, необходимо и достаточно, чтобы эта система содержала:

1) хотя бы одну немонотонную функцию,
2) хотя бы одну нелинейную функцию.

В частности, из доказанной теоремы следует, что алгебра Φ_{E_2} имеет базисы, состоящие из одной функции. Примерами таких базисов служат штрих Шеффера и стрелка Пирса (см. табл. 2).

Как подчеркивалось в § 3.3, удалось не только описать множество всех максимальных подалгебр алгебры логики, но и построить поверхность, которая включает все подалгебры данной алгебры [134].

Результаты этого параграфа используются в гл. 4 при изучении тождественных преобразований в системе алгоритмических алгебр (§ 4.2), а также при установлении критериев полноты для модифицированных алгебр Поста, связанных с многорегистровыми операторами (§ 4.3, 4.4).

§ 3.5. Изолированные множества. Конгруэнции

Пусть $U_A = \langle A; \Omega \rangle$ — некоторая универсальная алгебра. Предполагается, что любая операция $w \in \Omega$ существенна по всем своим аргументам. Функция $f(x_1, x_2, \ldots, x_n)$ существенна по переменной x_i, если найдется пара наборов, $\tilde{a} = (q_1, \ldots, q_{i-1}, q_i, q_{i+1}, \ldots, q_n)$, $\tilde{a}' = (q_1, \ldots, q_{i-1}, q_i', q_{i+1}, \ldots, q_n)$, для которых $f(\tilde{a}) \neq f(\tilde{a}')$.

Множество B называется изолированным относительно множества C в алгебре U_A $(B, C \subset A)$, если $B \subset C$, причем для любых n-местной операции $F \in \Omega$ и системы элементов $q_1, q_2, \ldots, q_n \in C$ $F(q_1, q_2, \ldots, q_n) \in B$ лишь тогда, когда $q_1, q_2, \ldots, q_n \in B$. Иными словами, для любой системы элементов $q_1, q_2, \ldots, q_n \in C$, среди которых найдется по крайней мере один $q_i \overline{\in} B$, произвольная n-местная операция $F \in \Omega$ удовлетворяет соотношению $F(q_1, q_2, \ldots, q_n) \overline{\in} B$ [8, 36]. При этом множество B, вообще говоря, незамкнуто, т. е. результат операции, примененной к некоторым элементам, входящим в B, может не принадлежать данному множеству. Изолированность множества B относительно C в алгебре U_A обозначим через $B \underset{(U_A)}{\lhd} C$. Множество $\varnothing$ будем считать изолированным относительно любого множества во всякой алгебре. Условимся также, что $B \underset{(U_A)}{\lhd} B$ для любого $B \subseteq A$. Если $B \underset{(U_A)}{\lhd} A$, т. е. множество B изолировано относительно основного множества A в алгебре U_A, то множество B называется **изолированным** в алгебре U_A.

Пример 1. В алгебре $\langle N; \{+\}\rangle$ любой начальный отрезок $N_m = \{1, 2, \ldots, m\}$ изолирован относительно отрезка $N_n = \{1, 2, \ldots, n\}$ при $m \leqslant n$. Аналогичное утверждение справедливо также в алгебре $\langle N; \{\times\}\rangle$.

Пример 2. В алгебре $F(\mathfrak{A}) = \langle F(\mathfrak{A}); \{\boxtimes\}\rangle$ (см. § 3.1, пример 5) множество $\{e\}$ изолировано относительно любого множества $M \subseteq F(\mathfrak{A})$. Кроме того, $M_k \underset{(F(\mathfrak{A}))}{\lhd} M_l$ при $(k \leqslant l)$, где M_r — множество всех слов над алфавитом $\mathfrak{A}$, длина которых не больше r, $M_r = \{s \mid s \in F(\mathfrak{A}) \text{ и } |s| \leqslant r\}$.

Операцией квазисуперпозиции, $L^n\begin{pmatrix} q_1, & q_2, & \ldots, & q_n \\ L_1, & L_2, & \ldots, & L_n \end{pmatrix}$, называется n-арная суперпозиция $S^n\begin{pmatrix} q_1, & q_2, & \ldots, & q_n & L \\ L_1, & L_2, & \ldots, & L_n & \end{pmatrix}$, у которой язык L фиксирован (см. § 3.1). Квазисуперпозиция $L(q_1, \ldots, q_n)$ существенна по всем своим аргументам тогда и только тогда, когда для любого символа q_i в языке L найдется по крайней мере одно слово $x \in L$ такое, что $x = x' q_i x''$, где $x', x'' \in F(\mathfrak{A})$ $(i = 1, 2, \ldots, n)$.

Пример 3. Пусть $\hat{S} = \left\{ L^n\begin{pmatrix} q_1, & \ldots, & q_n \\ L_1, & \ldots, & L_n \end{pmatrix} \middle| L \subseteq F(\mathfrak{A}), q_i \in \mathfrak{A} \right\}$ — совокупность (вообще говоря, бесконечная) всевозможных существенных квазисуперпозиций. Рассмотрим универсальную алгебру $\langle \tilde{\mathfrak{M}}(\mathfrak{A}); \hat{S}\rangle$ с основным множеством $\tilde{\mathfrak{M}}(\mathfrak{A}) = \mathfrak{M}(\mathfrak{A}) \setminus \{\{e\}\, \varnothing\}$ и сигнатурой операций $\hat{S}$. Множество $\mathfrak{M}_k(\mathfrak{A}) \subset \tilde{\mathfrak{M}}(\mathfrak{A})$ всех конечных языков изолировано в данной алгебре. Действительно, любая

квазисуперпозиция $L\begin{pmatrix} q_1, \ldots, q_n \\ L_1, \ldots, L_n \end{pmatrix} \in \hat{S}$, примененная к языкам L_1, $L_2, \ldots, L_n \in \tilde{\mathfrak{M}}(\mathfrak{A})$, среди которых есть по крайней мере один бесконечный язык, снова порождает бесконечный язык.

Из определения изолированного множества[1] следует, что если $B \underset{(U_A)}{\lhd} A$, то дополнение $\neg B$ до основного множества A образует подалгебру алгебры U_A. Относительная изолированность удовлетворяет свойству транзитивности: если $B \underset{(U_A)}{\lhd} C$ и $C \underset{(U_A)}{\lhd} D$, то $B \underset{(U_A)}{\lhd} D$, и антисимметричности: если $B \underset{(U_A)}{\lhd} C$ и $C \underset{(U_A)}{\lhd} B$, то $C = B$. Таким образом, бинарное отношение $\underset{(U_A)}{\lhd}$ является отношением частичного порядка, а множество $\mathfrak{M}(A)$ всех подмножеств множества A — частично упорядоченным множеством по отношению $\underset{(U_A)}{\lhd}$ (см. § 2.4).

Пусть $B, C, D \subset A$, причем $B \underset{(U_A)}{\lhd} C$. Тогда

$$(B \cap D) \underset{(U_A)}{\lhd} (C \cap D). \tag{3.22}$$

Пусть $U'_A = \langle A; \Omega' \rangle$ — универсальная алгебра с основным множеством A и сигнатурой операций $\Omega' \subset \Omega$. Тогда из $B \underset{(U_A)}{\lhd} C$ следует $B \underset{(U'_A)}{\lhd} C$.

Для изолированных множеств справедливо следующее утверждение.

Теорема 3.14. *Пусть $\{A_\alpha\}$ — совокупность множеств таких, что $A_\alpha \underset{(U_A)}{\lhd} B$. Тогда пересечение $D = \bigcap_\alpha A_\alpha$ также изолировано относительно множества B в алгебре U_A.*

Действительно, допустим, что D не изолировано относительно B в алгебре U_A. Это означает, что для некоторой n-арной операции $F \in \Omega$ найдутся элементы $q_1, q_2, \ldots, q_n \in B$ такие, что $F(q_1, q_2, \ldots, q_n) \in D$, причем (по крайней мере один) $q_i \overline{\in} D$. Тогда для элемента q_i можно отыскать множество $A_r \in \{A_\alpha\}$ такое, что $q_i \overline{\in} A_r$. Но $F(q_1, q_2, \ldots, q_n) \in A_r$, что противоречит изолированности множества A_r относительно B в алгебре U_A.

Если множество A_1 является подалгеброй алгебры U_A, причем $A_1 \underset{(U_A)}{\lhd} B$, то множество A_1 называется **подалгеброй, изолированной относительно B в алгебре U_A**. Иными словами, множество A_1 является подалгеброй, изолированной относительно B в алгебре U_A, если $A_1 \subset B$, причем для любых n-местной операции $F \in \Omega$ и системы элементов $q_1, q_2, \ldots, q_n \in B$ $F(q_1, q_2, \ldots, q_n) \in A_1$ тогда и только тогда, когда $(q_1, q_2, \ldots, q_n) \in A_1$.

Следствие. Пусть $\{A_\alpha\}$ — совокупность подалгебр алгебры U_A таких, что $A_\alpha \underset{(U_A)}{\lhd} B$. Тогда непустое пересечение $D = \bigcap_\alpha A_\alpha$ образует подалгебру алгебры U_A, причем $D \underset{(U_A)}{\lhd} B$.

Понятие изолированного множества используется при изучении некоторых свойств алгебры U_A и структуры ее подалгебр.

Теорема 3.15. *Пусть U'_A — некоторая подалгебра универсальной алгебры U_A (в частности, в роли U_{A_1} может выступать и вся алгебра U_A), B — множество, изолированное относительно U_{A_1} в алгебре U_A, $B \underset{(U_A)}{\lhd} A_1$. Тогда любая система образующих Σ подалгебры U_{A_1}, содержит подсистему $\Sigma' = \Sigma \cap B$ такую, что $[\Sigma'] \supseteq B$.*

Действительно, пусть $[\Sigma'] \not\supseteq B$. Тогда найдется элемент $q \in B$ такой, что $q \overline{\in} [\Sigma']$. Но вследствие изолированности множества B относительно подалгебры U_{A_1} элемент q не порождается с помощью элементов подмножества $(A_1 \setminus B) \subset A_1$. Значит, $[\Sigma] \overline{\ni} q$, что противоречит полноте системы Σ. Теорема доказана.

Следствие 1. Любая универсальная алгебра U_A, имеющая изолированную бесконечно порожденную подалгебру U_{A_1}, бесконечно порождена.

Действительно, по доказанной теореме любая система образующих Σ алгебры U_A содержит бесконечную подсистему $\Sigma' = \Sigma \cap A_1$ такую, что $[\Sigma'] = A_1$.

Следствие 2. Алгебра U_A, имеющая бесконечное множество изолированных подалгебр, бесконечно порождена.

Пусть $\{A_\alpha \mid \alpha \in I\}$ — бесконечная совокупность изолированных подалгебр алгебры U_A, $A_\alpha \underset{(U_A)}{\lhd} A$ (при любом $\alpha \in I$). Тогда произвольная система образующих Σ алгебры U_A для каждой подалгебры A_α ($\alpha \in I$) содержит подсистему $\Sigma_\alpha = \Sigma \cap A_\alpha$ такую, что $[\Sigma_\alpha] = A_\alpha$. Отсюда следует бесконечность системы Σ, так как конечная система не может иметь бесконечной совокупности подсистем.

Пусть U_A — некоторая универсальная алгебра, B — подалгебра, изолированная в U_A, $B \underset{(U_A)}{\lhd} A$, $C = \neg B$.

Теорема 3.16. *Алгебра U_A может быть представлена в виде прямой суммы подалгебр B и C, где $C = \neg B$ — дополнение множества B до основного множества A, так что $A = B \cup C$ и $B \cap C = \varnothing$.*

2. Множество $M^c = C \cup M$, где $M \subset B$, тогда и только тогда образует подалгебру алгебры U_A, когда таковой является множество M.

[1] Связь между понятиями «изолированное множество» и «идеал» установлена в работе [10].

Пусть M_1^c, M_2^c — *подалгебры алгебры* U_A, *причем* $M_1^c \subset M_2^c$ ($M_1, M_2 \subset B$).

3. Подалгебра M_1^c *максимальна относительно подалгебры* M_2^c *тогда и только тогда, когда* M_1 *максимальна относительно* M_2.

Справедливость условий этой теоремы непосредственно следует из определения изолированности.

Множество всех подалгебр алгебры U_A можно разбить на три класса.

1. Класс P_α — все подалгебры, каждая из которых включается в подалгебру B.

2. Класс P_β — все подалгебры, каждая из которых включается в подалгебру C.

3. Класс P_γ — все подалгебры такие, что $P_i \in P_\gamma$ тогда и только тогда, когда $B \cap P_i = B'$, $C \cap P_i = C'$, где $B' \in P_\alpha$, $C' \in P_\beta$. Следовательно, подалгебра P_i является прямой суммой подалгебр C' и B', причем подалгебра B' изолирована относительно P_i (см. § 1.2).

Таким образом, чтобы изучить структуру подалгебр U_A, в частности структуру подалгебр из класса P_γ, необходимо выяснить структуру подалгебр, входящих в классы P_α и P_β.

Проиллюстрируем изложенный выше метод на примере построения поверхности подалгебры O_9 (см. § 3.3, пример 6). Существует одна подалгебра O_4, изолированная относительно O_9 в алгебре Φ_{E_2}, $O_4 \underset{(\Phi_{E_2})}{\lhd} O_9$. Поэтому подалгебры, входящие в O_9, разделяются на три группы:

$P_\alpha = \{O_1, O_4\}$ — подалгебры, входящие в изолированную подалгебру O_4;

$P_\beta = \{O_2, O_3, O_7\}$ — подалгебры, входящие в $\urcorner O_4$;

$P_\gamma = \{O_5, O_6, O_8\}$ — подалгебры смешанного типа, каждая из которых является суммой подалгебр из P_α и P_β.

Подалгебра O_4 имеет лишь одну тривиальную максимальную подалгебру $O_1 \subset O_4$. Отсюда следует, что подалгебра $O_8 = O_7 \cup O_1$ максимальна для O_9. Другой подалгеброй, максимальной для O_9, является подалгебра O_4. Подалгебра $O_7 = \urcorner O_4$ имеет две тривиальные максимальные подалгебры: O_2, O_3. Следовательно, для O_8 максимальны подалгебры O_5, O_6, O_7; для O_5 — подалгебры O_1, O_2; для O_6 — подалгебры O_1, O_3.

Перейдем к рассмотрению понятий гомоморфизма и конгруэнции в универсальных алгебрах. Эти понятия играют важную роль как в алгебре, так и в теории формальных языков.

Пусть задано отображение φ множества A в B, где A, B — основные множества в однотипных универсальных алгебрах $U_A = \langle A; \Omega \rangle$, $U_B = \langle B; \Omega \rangle$ соответственно. Отображение φ называется **гомоморфным отображением** алгебры U_A в алгебру U_B, если для

любых элементов $a_1, a_2, \ldots, a_n \in A$ и произвольной n-арной операции $F \in \Omega$ выполняется соотношение

$$[F(a_1, a_2, \ldots, a_n)]\varphi = F[(a_1)\varphi, (a_2)\varphi, \ldots, (a_n)\varphi], \quad (3.23)$$

где $(a_i)\varphi = b_i \in B$ ($i = 1, 2, \ldots, n$). Если при этом отображение φ взаимно-однозначно, то φ — **изоморфное отображение** алгебры U_A в алгебру U_B; в случае, когда φ — отображение множества A на множество B (т. е. для каждого $b \in B$ найдется хотя бы один элемент $a \in A$, для которого $(a)\varphi = b$), говорят о **гомоморфизме** (соответственно **изоморфизме**) алгебры U_A на алгебру U_B. Алгебра U_B называется **гомоморфным (изоморфным)** образом алгебры U_A при отображении φ.

Если φ — гомоморфное отображение алгебры U_A в алгебру U_B и $B' \subset B$ — образ множества A при отображении φ, $B' = \{b \mid b = (a)\varphi\}$ для любого $a \in A$, то $U_{B'}$ образует подалгебру алгебры U_B и является гомоморфным образом алгебры U_A.

Пусть φ — гомоморфизм алгебры U_A в алгебру U_B, а ψ — гомоморфизм алгебры U_B в алгебру U_C. Тогда композиция $\varphi\psi$ отображений φ и ψ является гомоморфизмом алгебры U_A в алгебру U_C. Действительно, композиция $\varphi\psi$ является отображением множества A в множество C (см. § 2.3). Использовав соотношение (3.23), получим

$$[F(a_1, a_2, \ldots, a_n)]\varphi\psi = ([F(a_1, a_2, \ldots, a_n)]\varphi)\psi =$$
$$= [F((a_1)\varphi, (a_2)\varphi, \ldots, (a_n)\varphi)]\psi = F[[(a_1)\varphi]\psi, [(a_2)\varphi]\psi, \ldots$$
$$\ldots, [(a_n)\varphi]\psi) = F[(a_1)\varphi\psi, (a_2)\varphi\psi, \ldots, (a_n)\varphi\psi].$$

Пусть на основном множестве A алгебры $U_A = \langle A; \Omega \rangle$ определено n-отношение ρ^n. Отношение ρ^n называется **стабильным** в алгебре U_A, если для произвольной m-местной операции $F \in \Omega$ и любой последовательности m-мерных наборов (не обязательно различных)

$$\tilde{a}_1 = (a_{11}, a_{12}, \ldots, a_{1m}),$$
$$\tilde{a}_2 = (a_{21}, a_{22}, \ldots, a_{2m}),$$
$$\ldots\ldots\ldots\ldots$$
$$\tilde{a}_n = (a_{n1}, a_{n2}, \ldots, a_{nm})$$

такой, что $F(\tilde{a}_i) = b_i$ и $\rho^n(a_{1j}a_{2j}, \ldots, a_{nj})$ (где $a_{ij}, b_i \in A$ для любых $j = 1, 2, \ldots, m$; $i = 1, 2, \ldots, n$), выполняется соотношение $\rho^n(b_1, b_2, \ldots, b_n)$.

Бинарное отношение ρ называется конгруэнцией в алгебре $\langle A; \Omega \rangle$, если ρ стабильно в данной алгебре и является эквивалентностью на основном множестве A (см. § 2.4).

Пример 4. Конгруэнцией в алгебре $\langle N; \{+\} \rangle$ и $\langle N; \{>\} \rangle$ является отношение равенства $=$. Действительно, $=$ — отношение эквивалентности на множестве N (см. § 2.4, пример 1). Если

$a = a', b = b'$, где $a, a', b, b' \in N$, то $a + b = a' + b'$, $a \times b = a' \times b'$, т. е. отношение стабильно в рассматриваемых алгебрах.

Пример 5. Конгруэнциями в алгебре $\langle F(\mathfrak{A}); \{\boxtimes\}\rangle$ являются отношения эквивалентности $=, \sim$ (см. § 2.4, примеры 2, 3).

Пример 6. Пусть $U_A = \langle A; \Omega\rangle$ — универсальная алгебра, имеющая изолированную подалгебру $B \underset{(U_A)}{\lhd} A$. Тогда множество $C = \complement B$ также является подалгеброй алгебры U_A. Определим на множестве A бинарное отношение Σ так, что $a \Sigma b$ тогда и только тогда, когда $a, b \in B$, либо $a, b \in C$. С отношением Σ связано разбиение $R(A) = \{B, C\}$, и поэтому Σ — отношение эквивалентности на множестве A (см. § 2.4). Кроме того, для произвольной m-местной операции $F \in \Omega$ и любых наборов элементов $\tilde{a} = (a_1, a_2, \ldots, a_m)$, $\tilde{a}' = (a_1', a_2', \ldots, a_m')$ таких, что $a_i \Sigma a_i'$ $(i = 1, 2, \ldots, m)$, $F(\tilde{a})\ \Sigma F(\tilde{a}')$. Действительно, если $a_i \in B$ для любого $i = 1, 2, \ldots, m$, то по определению отношения Σ из $a_i \Sigma a_i'$ следует, что $a_i' \in B$ для каждого $i = 1, 2, \ldots, m$. Но B — подалгебра алгебры U_A, и поэтому $F(\tilde{a}), F(\tilde{a}') \in B$. Следовательно, $F(\tilde{a})\, \Sigma\, F(\tilde{a}')$.

Пусть теперь в наборе $\tilde{a} = (a_1, a_2, \ldots, a_m)$ найдется хотя бы один элемент $a_k \in C$. Тогда из $a_k \Sigma a_k'$ следует $a_k' \in C$. На основании изолированности подалгебры B $F(\tilde{a}), F(\tilde{a}') \in C$. Таким образом, и в этом случае $F(\tilde{a})\, \Sigma\, (\tilde{a}')$. Следовательно, отношение эквивалентности Σ стабильно в алгебре U_A и является конгруэнцией в данной алгебре.

Существует тесная связь между конгруэнциями и гомоморфизмами в универсальных алгебрах [66].

Пусть ρ — некоторая конгруэнция в универсальной алгебре $U_A = \langle A; \Omega\rangle$. Рассмотрим фактор-множество $A/\rho = \{S_a \mid a \in A\}$ и естественное отображение φ, сопоставляющее каждому элементу $a \in A$ соответствующий ему смежный класс $S_a(\rho)$ (см. § 2.4). Каждой m-местной операции $F \in \Omega$ поставим в соответствие m-местную операцию $\tilde{F}$, определенную на фактор-множестве A/ρ следующим образом: $\tilde{F}(S_{a_1}, S_{a_2}, \ldots, S_{a_m}) = S_a$, где $a = F(a_1, a_2, \ldots, a_m)$ для любых $a_1, a_2, \ldots, a_m \in A$. Вследствие стабильности отношения ρ операция $\tilde{F}$ определена однозначно.

Ввиду произвольности выбора операций $F \in \Omega$ можно рассматривать новую универсальную алгебру $U_{A/\rho} = \langle A/\rho; \tilde{\Omega}\rangle$, где $\tilde{\Omega} = \{\tilde{F} \mid F \in \Omega\}$, однотипную алгебре U_A. Алгебра $U_{A/\rho}$ называется **фактор-алгеброй** универсальной алгебры U_A по конгруэнции ρ. При этом отображение φ является естественным гомоморфизмом алгебры U_A на фактор-алгебру $U_{A/\rho}$. Например, конгруэнции Σ в алгебре U_A (см. пример 6) соответствует однотипная фактор-алгебра $U_A/\Sigma = \langle A/\Sigma; \Omega'\rangle$ с основным множеством $A/\Sigma = \{B, C\}$ и сигнатурой операций $\Omega' = \{\tilde{F} \mid F \in \Omega\}$, где двузначная m-местная операция $\tilde{F}$ определяется на m-мерных наборах смежных классов B, C так, что $\tilde{F}(B, B, \ldots, B) = B$ и на любом наборе, содержащем хотя бы один элемент C, операция $\tilde{F}$ принимает значение C.

Проведенные рассуждения показывают, что любой конгруэнции ρ в алгебре U_A соответствует естественный гомоморфизм φ алгебры U_A на фактор-алгебру $U_{A/\rho}$. С другой стороны, произвольному гомоморфизму ψ алгебры U_A на однотипную алгебру U_B соответствует конгруэнция ρ в алгебре U_A, для которой существует изоморфизм τ алгебры U_B на фактор-алгебру $U_{A/\rho}$ такой, что $\psi\tau = \varphi$ (φ — естественный гомоморфизм U_A на $U_{A/\rho}$ (рис. 16)). Это утверждение известно под названием **теоремы о гомоморфизмах** ([66], гл. III, § 1.8).

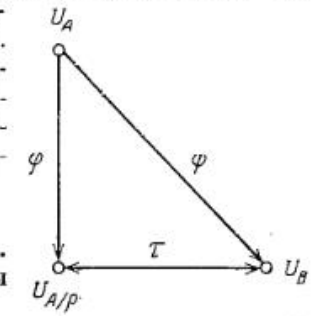


Рис. 16.

Конструкции, связанные с гомоморфизмами, часто используются при описании синтаксиса и семантики формальных языков [71], в частности языков программирования (см. § 6.7).

§ 3.6. Полугруппы. Многоосновные алгебры

Рассмотрим универсальную алгебру $U_A = \langle A; \Omega\rangle$. Пусть S_1 и S_2 — суперпозиции операций системы Ω, $S_1, S_2 \in [\Omega]_S$ (см. § 3.1), причем $x_1, x_2, \ldots, x_n$ — переменные, каждая из которых входит хотя бы в одну из суперпозиций S_1, S_2.

Выражение $S_1 = S_2$ называется тождеством алгебры U_A, если суперпозиции S_1 и S_2 принимают одно и то же значение для любого набора $\tilde{a} = a_1, a_2, \ldots, a_n$ значений переменных $x_1, x_2, \ldots, x_n$ в основном множестве A. В этом случае говорят, что тождество $S_1 = S_2$ выполняется в алгебре U_A. В гл. 4 (§. 4.2) подробно изучаются тождества, которые выполняются в системах алгоритмических алгебр, составляющих основу прикладной теории алгоритмов.

Пусть

$$T = \{S_1^i = S_2^i \mid i = 1, 2, \ldots, m\} \tag{3.24}$$

является системой тождеств, левые и правые части которых — суперпозиции операций сигнатуры Ω. Примитивным классом или многообразием [77] $K(T) = \{\langle A_\alpha; \Omega\rangle \mid \alpha \in I\}$ называется множество всех однотипных универсальных алгебр $U_{A_\alpha} = \langle A_\alpha; \Omega\rangle$, в каждой из которых выполняются тождества, принадлежащие системе T (3.24).

Теорема (3.17). *Любой примитивный класс $K(T)$ вместе с любой своей алгеброй $U_{A_\alpha} \in K(T)$ содержит все подалгебры и гомоморфные образы данной алгебры* [66].

Действительно, если тождество, принадлежащее системе T, выполняется в алгебре $U_{A_\alpha} \in K(T)$, то оно выполняется, в частности, и для элементов любой подалгебры данной алгебры. Пусть, далее, φ — произвольный гомоморфизм алгебры $U_A \in K(T)$ на однотипную алгебру $U_B = \langle B; \Omega\rangle$. Рассмотрим произвольный набор $\tilde{\beta} = (b_1, b_2, \ldots, b_{n_k})$ (где $b_i \in B$, $i = 1, 2, \ldots, n_k$) значений переменных $x_1, x_2, \ldots, x_{n_k}$, входящих в некоторое тождество $S_1^k = S_2^k$ системы T $(1 \leqslant k \leqslant m)$. Алгебра U_B является гомоморфным образом алгебры U_A при гомоморфизме φ, следовательно, найдется набор $\tilde{\alpha} = (a_1, a_2, \ldots, a_{n_k})$ значений переменных $x_1, x_2, \ldots, x_{n_k}$ такой, что $(a_i)\varphi = b_i$, $a_i \in A$ $(i = 1, 2, \ldots, n_k)$. Но $U_A \in K(T)$, следовательно, $S_1^k(\tilde{\alpha}) = S_2^k(\tilde{\alpha})$, где $S_j^k(\tilde{\alpha})$ — значение суперпозиции S_j^k на наборе $\tilde{\alpha}$ $(j = 1, 2)$. Отсюда по определению гомоморфизма $S_1^k(\tilde{\beta}) = [S_1^k(\tilde{\alpha})]\varphi = [S_2^k(\tilde{\alpha})]\varphi = S_2^k(\tilde{\beta})$. Таким образом, ввиду произвольности набора $\tilde{\beta}$ тождество $S_1^k = S_2^k$ выполняется в алгебре U_B при любом $k = 1, 2, \ldots, m$ и $U_B \in K(T)$. Теорема доказана.

Множество всех однотипных универсальных алгебр можно рассматривать как примитивный класс с пустой системой тождеств, в частности как класс $K(\varnothing) = \{\langle A_\alpha; \{\circ\}\rangle\}$ всех универсальных алгебр, сигнатура операций которых состоит из единственной бинарной операции $x \circ y$. Каждая алгебра, входящая в класс $K(\varnothing)$, называется **группоидом**. Группоид, основной операцией которого является **умножение**, называется **мультипликативным**, в отличие от **аддитивного** группоида, в котором основная операция сложение, $x + y$. Подклассами класса $K(\varnothing)$ являются примитивные классы $K(T_1)$, $K(T_2)$, $K(T_3)$ с системами тождеств

$$T_1 = \{x \circ x = x\},$$
$$T_2 = \{x \circ y = y \circ x\},$$
$$T_3 = \{(x \circ y) \circ z = x \circ (y \circ z)\}.$$

Группоиды, принадлежащие классу $K(T_1)$, называются **идемпотентными**, группоиды класса $K(T_2)$ — **коммутативными или абелевыми**, а группоиды класса $K(T_3)$ — **полугруппами**.

Примитивными являются класс $K(T_1 \cup T_3) = K(T_1) \cap K(T_3)$, содержащий все идемпотентные полугруппы, и класс $K(T_2 \cup T_3) = K(T_2) \cap K(T_3)$, состоящий из всех коммутативных полугрупп.

Пусть $\Pi_A = \langle A; \{\circ\}\rangle$ — мультипликативная полугруппа. Элемент $e \in A$ называется **правой** единицей полугруппы Π_A, если $ae = a$ для любого $a \in A$, **левой** единицей, если $ea = a$ для любого $a \in A$, и **двусторонней** единицей (или просто **единицей**), если $ea = ae = a$ для любого $a \in A$. В случае аддитивной полугруппы аналогично определяется левый, правый и двусторонний нули. Если полугруппа $\Pi_A = \langle A; \{\circ\}\rangle$ имеет хотя бы одну левую единицу e_1 и хотя бы одну правую единицу e_2, то она имеет единственную двустороннюю единицу e, причем $e_1 = e_2 = e$. Действительно, $e_1 e_2 = e_1$ и $e_1 e_2 = e_2$, следовательно, $e_1 = e_2$ — единственная двусторонняя единица полугруппы Π_A. Подалгебры некоторой полугруппы Π_A называются **подполугруппами** данной полугруппы.

Часто рассматриваются полугруппы с **выделенной**, или **главной**, единицей e. Примитивный класс таких полугрупп $K'(T_3) = \{\langle A_\alpha; \Omega\rangle \mid \alpha \in I\}$ определяется как совокупность всех полугрупп с сигнатурой операций $\Omega = \{\circ, e\}$, где e — унарная операция, фиксирующая единицу полугруппы. Полугруппы, принадлежащие классу $K'(T_3)$, называются **моноидами**. Очевидно, моноид и любая его подалгебра являются полугруппами с единицей. В то же время полугруппа с единицей может иметь полугруппы, не содержащие единиц.

Пример 1. Алгебра $\langle N; \{\times\}\rangle$ является мультипликативной полугруппой с единицей. Очевидно, рассматриваемая полугруппа имеет подполугруппы без единиц, например $N_r = \{n \mid n > r\}$ для некоторого натурального r. Аналогично аддитивной полугруппой с нулем является алгебра $\langle\{0, 1, 2, \ldots\}; \{+\}\rangle$.

Пример 2. Алгебра $\langle F(\mathfrak{A}); \{\boxtimes\}\rangle$ с основным множеством $F(\mathfrak{A})$ всех слов в алфавите $\mathfrak{A}$ и операцией конкатенации $\boxtimes$ (см. § 3.1, пример 5) образует полугруппу с единицей e (e — пустое слово).

Полугруппа $F(\mathfrak{A})$ называется **свободной полугруппой над алфавитом $\mathfrak{A}$ или (кратко) свободной полугруппой.**

Пример 3. Алгебра $\langle \mathfrak{M}(A); \{\cap\}\rangle$, где $\mathfrak{M}(A)$ — множество всех подмножеств непустого множества A, образует полугруппу, единицей в которой является множество A. Алгебра $\langle \mathfrak{M}(A); \{\cap\}\rangle$ относится к коммутативным полугруппам, причем полугруппа $\langle \mathfrak{M}(A); \{\cap\}\rangle$ идемпотентна.

Пример 4. Некоммутативной полугруппой с единицей является алгебра $R_A = \langle R(A); \{\boxtimes\}\rangle$ с основным множеством $R(A)$ всех бинарных отношений на множестве A и операцией композиции отношений $\times$. Единицей этой полугруппы служит диагональное отношение $\triangle_A$ (см. § 2.2).

Одной из важных подполугрупп полугруппы R_A является полугруппа $S_A = \langle S(A); \{\times\}\rangle$ с основным множеством $S(A)$ всех отображений множества A в себя. Отображения множества A в себя часто называют **преобразованиями** или **операторами** на множестве A (см. гл. 4). Любой оператор $P \in S(A)$ на множестве $A = \{a_1, a_2, \ldots, a_n, \ldots\}$ может быть задан в виде подстановки

$$P = \begin{pmatrix} a_1, & a_2, & \ldots, & a_n, & \ldots \\ (a_1)P, & (a_2)P, & \ldots, & (a_n)P, & \ldots \end{pmatrix},$$

где $(a_n)P$ — результат применения оператора P к элементу $a_n \in A$ $(n = 1, 2, \ldots)$. Поэтому полугруппа S_A называется **симметри-**

ческой полугруппой подстановок. Эта полугруппа играет важную роль в теории полугрупп. В частности, справедливо следующее утверждение.

Теорема 3.18. *Произвольная полугруппа* $\Pi_A = \langle A; \{\circ\}\rangle$ *изоморфна некоторой подполугруппе симметрической полугруппы* $\langle S(\overline{A}); \{\times\}\rangle$. *Иначе говоря,* Π_A *изоморфно вложима в полугруппу* $S(\overline{A})$, *причем*

$$\overline{A} = \begin{cases} A, & \text{если } \Pi_A \text{ — полугруппа с единицей,} \\ A \cup e & \text{в противном случае,} \end{cases}$$

где $e \notin A$, $e \circ e = e$ *и* $a \circ e = e \circ a = a$ *для любого* $a \in A$.

Доказательство. Допустим вначале, что Π_A — полугруппа с единицей e. Каждому элементу $a \in A$ поставим в соответствие оператор $P_a \in S(A)$, заданный подстановкой

$$P_a = \begin{pmatrix} e, & a_1, & a_2, & \dots, & a_n, & \dots \\ e \circ a, & a_1 \circ a, & a_2 \circ a, & \dots, & a_n \circ a, & \dots \end{pmatrix},$$

где $\{e, a_1, a_2, \dots, a_n, \dots\} = A$. Для различных элементов a, $a' \in A$ $a = e \circ a \neq e \circ a' = a'$, поэтому указанное соответствие взаимно-однозначно. Более того, $(x) P_{a \circ a'} = x \circ (a \circ a') = (x \circ a) \circ a' = ((x) P_a) P_{a'}$ для любого $x \in A$. Таким образом, $P_{a \circ a'} = P_a \times P_{a'}$, следовательно, в рассматриваемом случае полугруппа Π_A изоморфно вложима в симметрическую полугруппу S_A. Если Π_A — полугруппа без единицы, то, присоединяя к множеству A единицу e, нетрудно перейти к полугруппе $\Pi_{\overline{A}} = \langle \overline{A}; \{\circ\}\rangle$ с единицей e, для которой Π_A — подполугруппа. Но согласно доказанному выше полугруппа $\Pi_{\overline{A}}$ изоморфно вложима в симметрическую полугруппу $S(\overline{A})$. Отсюда следует, что подполугруппа Π_A также изоморфно вложима в $S(\overline{A})$. Теорема доказана.

Пусть $K(T) = \{\langle A_\alpha; \tilde{\Omega}\rangle \mid \alpha \in I\}$ — примитивный класс универсальных алгебр с сигнатурой операций $\tilde{\Omega} = \{\circ, -1\}$ (где $x \circ y$ — бинарная полугрупповая операция, x^{-1} — унарная операция взятия обратного элемента) и системой тождеств

$$T = \begin{cases} x \circ (y \circ z) = (x \circ y) \circ z, & (3.25) \\ y^{-1} \circ (y \circ x) = x = (x \circ y) \circ y^{-1}. & (3.26) \end{cases}$$

Каждая универсальная алгебра, входящая в примитивный класс $K_\tau(T)$, называется **группой**. В каждой группе существует однозначно определенный единичный элемент e, для которого справедливо тождество $xe = ex = x$.

Таким образом, всякая группа является полугруппой с единицей. Элемент y^{-1} называется **обратным** элементу y, а тождество (3.26) — **законом обращения.** Заметим, что в группе каждое из

уравнений $ax = b$, $ya = b$ имеет единственное решение $x = a^{-1}b$ и $y = ba^{-1}$ соответственно.

Пример 5. Группой является аддитивная полугруппа $\langle Z; \{+, -1\}\rangle$ с нулем, где Z — множество всех целых чисел $a^{-1} = -a$ для любого $a \in Z$.

Пример 6. Группой является полугруппа $\tilde{S}(A) \subset S(A)$ всех взаимно-однозначных отображений множества A в себя (см. § 2.3), называемых также **обратимыми операторами.**

Не останавливаясь более подробно на теории групп, которой посвящена фундаментальная работа А. Г. Курош [67], заметим лишь, что для групп справедлив аналог теоремы 3.18, состоящий в том, что каждая группа изоморфна некоторой группе обратимых операторов. Связь теории групп с обратимыми операторами определяет важность этой теории, а также ее более раннее развитие по сравнению с теорией полугрупп. В последнее время наблюдается все возрастающий интерес к общей теории полугрупп [74], в частности в связи с использованием ее в теории формальных языков.

Пусть $\Pi_A = \langle A; \{\circ\}\rangle$ — некоторая мультипликативная полугруппа. Элемент $a \in A$ называется **неразложимым** в полугруппе Π_A, если не существует элементов $b, c \in A$ (отличных от единицы e, когда Π_A — полугруппа с единицей) таких, что $a = bc$. Очевидно, любая система образующих Σ полугруппы Π_A содержит все неразложимые элементы данной полугруппы. Если при этом найдется система образующих $\tilde{\Sigma}$ полугруппы Π_A, состоящая лишь из неразложимых элементов, то $\tilde{\Sigma}$ — единственный базис данной полугруппы.

Пример 7. Система $\{a_i \cup e \mid a_i \in \mathfrak{A}\}$, состоящая из всех неразложимых элементов свободной полугруппы $F(\mathfrak{A})$, является единственным базисом данной полугруппы (см. § 3.3, пример 2).

Пример 8. Множество P всех простых чисел является единственным базисом, состоящим из всех неразложимых элементов полугруппы $\langle N; \{\cdot\}\rangle$ (см. § 3.3, пример 4).

В теории полугрупп, как и в теории формальных языков, важную роль играют свободные полугруппы (см. пример 2). В качестве алфавита свободной полугруппы выберем некоторую систему образующих $\Sigma \subseteq A$ полугруппы Π_A. Каждому слову $q_1 q_2 \dots q_r \in F(\Sigma)$ соответствует элемент $a \in A$ такой, что $q_1 \cdot q_2 \cdot \dots \cdot q_r = a$. Элемент a называется **значением** слова $q_1 q_2 \dots q_r$ в полугруппе Π_A. Для некоторого элемента $a \in A$ в свободной полугруппе $F(\Sigma)$ обычно существует несколько слов:

$$s_1 = q_{11} q_{12} \dots q_{1r_1},$$
$$s_2 = q_{21} q_{22} \dots q_{2r_2},$$
$$\dots \dots \dots \dots \dots$$
$$s_r = q_{r1} q_{r2} \dots q_{rr_r}.$$

($q_{ij} \in \Sigma$), значением каждого из которых является данный элемент. Это означает, что в полугруппе Π_A выполняется равенство произведений

$$q_{e1} \cdot q_{e2} \cdot \ldots \cdot q_{en_e} = q_{t1} \cdot q_{t2} \cdot \ldots \cdot q_{tn_t} \ (1 \leqslant e,\ t \leqslant r), \quad (3.27)$$

называемое **соотношением в полугруппе** Π_A относительно системы Σ или просто **соотношением в** Π_A, если ясно, о какой Σ идет речь. Таким образом, соотношением в полугруппе Π_A является пара слов свободной полугруппы $F(\Sigma)$, имеющих в Π_A одно и то же значение.

Рассмотрим такое бинарное отношение π в свободной полугруппе $F(\Sigma)$, что $s \pi s'$ тогда и только тогда, когда слова $s,\ s' \in F(\Sigma)$ принимают в полугруппе Π_A одно и то же значение, т. е. справедливо соотношение $s = s'$. Отношение π рефлексивно, симметрично и транзитивно; это означает, что π — отношение эквивалентности на множестве $F(\Sigma)$. Более того, π — конгруэнция и можно рассматривать фактор-полугруппу $\langle F(\Sigma)/\pi;\ \{\cdot\}\rangle$ с основным множеством

$$F(\Sigma)/\pi = \{B_a \mid a \in A\}, \quad (3.28)$$

где $B_a \subset F(\Sigma)$ — смежный класс по конгруэнции π, состоящий из всех слов, принимающих в полугруппе Π_A значение a, и операцией умножения классов $B_a \cdot B_b = B_{a \cdot b}$. Из каждого класса $B_a \in F(\Sigma)/\pi$ выберем по одному слову $u_a \in B_a$ — представителю данного класса. Совокупность $\{u_a \mid a \in A\}$ называется **множеством канонических форм** элементов полугруппы Π_A. Очевидно, каждый элемент $a \in A$ имеет единственную каноническую форму u_a. Если при этом по каноническим формам u_a, u_b легко построить каноническую форму $u_{a \cdot b}$ (т. е. можно говорить об умножении канонических форм), то полугруппа Π_A полностью представляется связанной с ней полугруппой канонических форм.

Пример 9. Рассмотрим мультипликативную полугруппу $\langle N;\ \{\times\}\rangle$, базисом которой является множество P всех простых чисел (см. пример 8, а также § 3.2, пример 7, и § 3.3, пример 4). Построим фактор-полугруппу $\langle F(P)/\pi;\ \cdot\rangle$ свободной полугруппы $F(P)$ по конгруэнции π, где $F(P)/\pi = \{B_n \mid n \in N\}$ — фактор-множество, каждый элемент класса B_n в котором содержит все слова свободной полугруппы $F(P)$, являющиеся разложениями числа n на простые множители. Канонической формой $u_n \in B_n$ любого числа $n \in N$ будем считать слово вида

$$\underbrace{p_1 \ldots p_1}_{\alpha_1 \text{ раз}} \underbrace{p_2 \ldots p_2}_{\alpha_2 \text{ раз}} \ldots \underbrace{p_k \ldots p_k}_{\alpha_k \text{ раз}}$$

такое, что $p_1^{\alpha_1} \cdot p_2^{\alpha_2} \cdot \ldots \cdot p_k^{\alpha_k} = n$, где $1 \leqslant p_1 < p_2 < \cdots < p_k$ ($p_1, p_2, \ldots, p_k \in P$). Хорошо известен удобный способ умножения чисел, заданных разложениями на простые множители, который определяет операцию умножения на множестве $\{u_n \mid n \in N\}$ всех канонических форм. Таким образом, полугруппу $\langle N;\ \{\times\}\rangle$ можно представить с помощью построенной выше полугруппы канонических форм.

С каноническими формами элементов полугруппы тесно связано понятие совокупности определяющих соотношений. Пусть $C = \{v_\alpha = w_\alpha \mid \alpha \in \Gamma\}$ — совокупность соотношений в полугруппе Π_A относительно системы образующих Σ. Слово w выводимо из слова v с помощью системы C, $v \underset{(C)}{\dashv} w$, где $v,\ w \in F(\Sigma)$, если существует цепочка

$$v = s_0,\ s_1,\ \ldots,\ s_k = w \quad (3.29)$$

слов $s_i \in F(\Sigma)$ ($i = 0, 1, \ldots, k$) (при $k = 0$ слово w графически совпадает с v, $w \equiv v$), в которой слово s_i можно представить в виде $s_i = r_1^i z r_2^i$, где $r_1^i,\ r_2^i \in F(\Sigma)$, так что в системе C найдется соотношение $v_\alpha = w_\alpha$, для которого либо $z = v_\alpha$ — и тогда $s_{i+1} = r_1^i w_\alpha r_2^i$, либо $z = w_\alpha$ — и тогда $s_{i+1} = r_1^i v_\alpha r_2^i$ для любого $i = 0, 1, \ldots, k-1$. Иными словами, для того чтобы из слова s_i получить слово s_{i+1}, достаточно в s_i выделить подслово z, которое является левой или правой частью некоторого соотношения из C, и заменить его другой частью этого соотношения. Цепочка (3.29) называется **выводом** слова w из v.

Отношение $\underset{(C)}{\dashv}$ на множестве $F(\Sigma)$ рефлексивно. Кроме того, если $v \underset{(C)}{\dashv} w$, то, произведя вывод в обратном порядке, получим $w \underset{(C)}{\dashv} v$, следовательно, отношение $\underset{(C)}{\dashv}$ симметрично. Отношение $\underset{(C)}{\dashv}$ транзитивно: если $v \underset{(C)}{\dashv} w$ и $w \underset{(C)}{\dashv} u$, то $v \underset{(C)}{\dashv} u$. Таким образом, отношение $\underset{(C)}{\dashv}$ является отношением эквивалентности на множестве $F(\Sigma)$. Очевидно, если $v \underset{(C)}{\dashv} w$, то слова $v,\ w \in F(\Sigma)$ принимают в полугруппе Π_A одно и то же значение. Значит, $v = w$ является следствием соотношений системы C в полугруппе Π_A и фактор-множество $F(\Sigma)/\underset{(C)}{\dashv}$ образует подразбиение разбиения $F(\Sigma)/\pi$.

Совокупность C соотношений относительно Σ в полугруппе Π_A называется **определяющей совокупностью**, если любое соотношение $v = w$ относительно Σ является следствием из C. Определяющей совокупностью соотношений относительно Σ в полугруппе Π_A является, например, множество всех соотношений относительно Σ. Если в качестве системы образующих Σ выбрано основное множество A полугруппы Π_A, то определяющая совокупность соот-

ношений относительно A может быть построена по таблице умножения в данной полугруппе.

Пусть C — совокупность соотношений относительно Σ в полугруппе Π_A и для элементов $a \in A$ данной полугруппы определены канонические формы $u_a \in B_a$, где B_a — смежный класс из фактор-множества $F(\Sigma)/\pi$.

Теорема 3.19. *Совокупность C является определяющей, если следствием из C будет любое соотношение $v = u_a$ для любого $v \in B_a (a \in A)$. Иначе говоря, C — определяющая совокупность соотношений, если с ее помощью всякое слово $w \in F(\Sigma)$ можно привести к канонической форме.*

Действительно, выберем произвольное соотношение $v = w$ относительно Σ в полугруппе Π_A. По определению слова $v \in F(\Sigma)$, $w \in F(\Sigma)$ принимают в Π_A одно и то же значение a, следовательно, $v, w \in B_a$ и им соответствует одна каноническая форма $u_a \in B_a$. По предположению соотношения $v = u_a$ и $w = u_a$ являются следствиями совокупности C, т. е. $v \underset{(C)}{\dashv} u_a$ и $w \underset{(C)}{\dashv} u_a$. Но в силу симметричности и транзитивности отношения $\underset{(C)}{\dashv}$ из $w \underset{(C)}{\dashv} u_a$ следует $u_a \underset{(C)}{\dashv} w$, а из $v \underset{(C)}{\dashv} u_a$ и $u_a \underset{(C)}{\dashv} w$ следует $v \underset{(C)}{\dashv} w$. Таким образом, соотношение $v = w$ является следствием совокупности соотношений C. Теорема доказана.

Заметим, что с помощью совокупности определяющих соотношений можно задавать полугруппу Π_A с точностью до изоморфизма. Однако при таком способе задания могут возникать трудности, связанные с алгоритмической неразрешимостью проблемы тождеств в некоторых ассоциативных исчислениях [74]. К теории ассоциативных исчислений относятся порождающие грамматики формальных языков, которым посвящена гл. 5.

Важными естественными обобщениями понятия универсальной алгебры являются многоосновные алгебры или алгебры со схемами операторов, предложенные П. Хиггинсом (см. [71]).

Многоосновная алгебра представляет собой систему $\langle \mathfrak{M}; \Omega \rangle$, состоящую из семейства основных множеств $\mathfrak{M} = \{A_\alpha \mid \alpha \in I\}$ и сигнатуры операций Ω, определенных на семействе $\mathfrak{M}$ следующим образом: каждой n-местной операции $F_{\langle \alpha_1, \ldots, \alpha_n; \alpha_r \rangle} \in \Omega$ однозначно сопоставлен кортеж $\langle \alpha_1, \ldots, \alpha_n; \alpha_r \rangle$ — схема данной операции, так что $F_{\langle \alpha_1, \ldots, \alpha_n; \alpha_r \rangle}$ является отображением декартова произведения $A_{\alpha_1} \times A_{\alpha_2} \times \cdots \times A_{\alpha_n}$ в множество A_{α_r}, где $\alpha_j, \alpha_r \in I$ $(j = 1, 2, \ldots, n)$. Таким образом, в многоосновных алгебрах вместо одного основного множества используется семейство $\mathfrak{M}$ основных множеств, а операции $F_{\langle \alpha_1, \ldots, \alpha_n; \alpha_r \rangle} \in \Omega$ являются n-местными функциями $F(x_1, \ldots, x_n)$, аргументы которых x_q $(q = 1, 2, \ldots, n)$ определены на множествах $A_{\alpha_q} \in \mathfrak{M}$ соответственно, сама же функция F принимает значения из множества $A_{\alpha_r} \in \mathfrak{M}$. Причем каждая нульарная операция $F \in \Omega$ фиксирует в множестве $A_{\alpha_r} \in \mathfrak{M}$ некоторый элемент (константу) $a \in A_{\alpha_r}$. Элементами многоосновной алгебры $\langle \mathfrak{M}; \Omega \rangle$ являются элементы ее основных множеств A_α $(\alpha \in I)$. Если семейство $\mathfrak{M}$ состоит из одной компоненты $\mathfrak{M} = \{A\}$, то алгебра универсальна, $\langle A; \Omega \rangle$.

На многоосновные алгебры распространяются рассмотренные в данной главе определения и конструкции теории универсальных алгебр.

Пусть $\mathfrak{M} = \{A_\alpha \mid \alpha \in I\}$ и $\mathfrak{M}' = \{B_\alpha \mid \alpha \in I\}$ — семейства множеств. Семейство $\mathfrak{M}'$ включается в семейство $\mathfrak{M}$ $(\mathfrak{M}' \subset \mathfrak{M})$, если $B_\alpha \subset A_\alpha$ для любого $\alpha \in I$. При включении $\mathfrak{M}' \subset \mathfrak{M}$ семейство $\mathfrak{M}'$ называется **подсемейством** семейства $\mathfrak{M}$. Можно говорить также о теоретико-множественных операциях над семействами, которые сводятся к одноименным покомпонентным операциям. Например, $\mathfrak{M} \cap \mathfrak{M}' = \{A_\alpha \cap B_\alpha \mid \alpha \in I\}$.

Подсемейство $\mathfrak{M}' \subset \mathfrak{M}$ называется **подалгеброй алгебры** $\langle \mathfrak{M}; \Omega \rangle$, если $\mathfrak{M}'$ замкнуто относительно всех операций данной алгебры. Подсемейство $\mathfrak{M}' \subset \mathfrak{M}$, для которого $[\mathfrak{M}'] = \mathfrak{M}$, называется **системой образующих алгебры** $\langle \mathfrak{M}; \Omega \rangle$. **Однотипными** называются многоосновные алгебры $\langle \mathfrak{M}; \Omega \rangle$ и $\langle \mathfrak{M}'; \Omega \rangle$ с одной и той же сигнатурой операций Ω и семействами основных множеств $\mathfrak{M} = \{A_\alpha \mid \alpha \in I\}$ и $\mathfrak{M}' = \{B_\alpha \mid \alpha \in I\}$ соответственно.

Отображением алгебры $\langle \mathfrak{M}; \Omega \rangle$ в однотипную алгебру $\langle \mathfrak{M}'; \Omega \rangle$ называется семейство отображений $\varphi = \{\varphi_\alpha \mid \alpha \in I\}$ таких, что φ_α есть отображение множества $A_\alpha \in \mathfrak{M}$ в множество $B_\alpha \in \mathfrak{M}'$. Отображение φ алгебры $\langle \mathfrak{M}; \Omega \rangle$ в однотипную алгебру $\langle \mathfrak{M}'; \Omega \rangle$ называется **гомоморфизмом**, если любая n-местная операция $F_{\langle \alpha_1, \ldots, \alpha_n; \alpha_r \rangle} \in \Omega$ для любых элементов $a_1 \in A_{\alpha_1}, \ldots, a_n \in A_{\alpha_n}$ удовлетворяет равенству

$$[F_{\langle \alpha_1, \ldots, \alpha_n; \alpha_r \rangle}(a_1, \ldots, a_n)]\varphi = F((a)_1\varphi, (a_2)\varphi, \ldots, (a_n)\varphi).$$

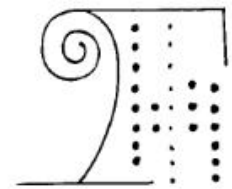
Здесь $(a)\varphi = (a)\varphi_\alpha$ при $a \in A_\alpha$ для любого $\alpha \in I$. Если каждое отображение $\varphi_\alpha (\alpha \in I)$ взаимно-однозначно, то φ — **изоморфизм** алгебры $\langle \mathfrak{M}; \Omega \rangle$ в $\langle \mathfrak{M}'; \Omega \rangle$.

Пусть $X = \{X_\alpha \mid \alpha \in I\}$ — некоторое семейство множеств. **Свободной многоосновной алгеброй** $S = \langle \{S_\alpha \mid \alpha \in I\}; \Omega \rangle$, порожденной семейством X, называется алгебра, элементы которой — слова в алфавите $\{F/F \in \Omega \cup X\}$, причем по индукции: а) однобуквенными словами алгебры S являются элементы множеств $X_\alpha \subset S_\alpha$ (для любого $\alpha \in I$) и только они, б) если $s_1 \in S_{\alpha_1}$, $s_2 \in S_{\alpha_2}$, $\ldots$, $s_n \in S_{\alpha_n}$ —

элементы алгебры S, то для любой n-местной операции $F \in \Omega$ со схемой $\langle \alpha_1, \ldots, \alpha_n; \alpha_r \rangle$ слово $Fs_1s_2 \ldots s_n \in S_{\alpha_r}$ — также элемент алгебры S. Семейство X называется **системой свободных образующих алгебры** S.

Многоосновная алгебра $\langle \mathfrak{M}; \Omega \rangle$ с выделенной основной компонентой $A_0 \in \mathfrak{M}$ называется **порождающей**. Важным примером порождающей многоосновной алгебры является система алгоритмических алгебр [26] (см. гл. 4). Порождающие многоосновные алгебры находят приложения при описании синтаксиса и семантики языков программирования [71] (см. § 6.7).

Часть вторая ЭЛЕМЕНТЫ ТЕОРИИ ПРОГРАММИРОВАНИЯ



ГЛАВА 4

СИСТЕМЫ АЛГОРИТМИЧЕСКИХ АЛГЕБР

§ 4.1. Понятие системы алгоритмических алгебр

Современные ЭВМ представляют собой сложные комплексы устройств, снабженные развитыми системами математического обеспечения, которые включают языки программирования и трансляторы с этих языков, операционные системы, диспетчеры, мониторы, различного рода обслуживающие программы, а также системы стандартных и типовых подпрограмм. Все эти средства могут быть реализованы как программным, так и схемным способом. Процесс разработки систем математического обеспечения, как и проектирования самих ЭВМ, чрезвычайно сложный и трудоемкий. Состав и способы разработки систем математического обеспечения существенно связаны со структурой данной машины и должны быть готовы к моменту ее выпуска. Поэтому систему математического обеспечения ЭВМ следует разрабатывать одновременно с ее проектированием. В связи с этим актуальной проблемой современной вычислительной техники является автоматизация разработки систем математического обеспечения и проектирования ЭВМ как единого процесса. Ее решение требует развития новых теоретических направлений в кибернетике, в частности прикладной теории алгоритмов. Одним из основных источников задач прикладной теории алгоритмов является проблема оптимального перевода с одного языка на другой, которая может быть сформулирована следующим образом: существуют два алгоритмических языка и некоторый алгоритм, написанный на одном из них; требуется найти оптимальную по заданным критериям реализацию этого алгоритма на другом языке. В программировании обычно первым является некоторый язык программирования, ориентированный на тот или иной круг задач, а вторым — внутренний язык машины, на которой решаются данные задачи. Таким образом, речь идет о трансляции с языка программирования на машинный язык с одновременной оптимизацией выходной программы. В то же время исходным может быть алгоритм работы некоторого устройства ЭВМ, запи-

санный на предназначенном для этой цели алгоритмическом языке, а язык, на который транслируется данный алгоритм, — это язык схем. Тогда задача состоит в получении оптимальной схемы, реализующей алгоритм работы данного устройства или некоторой его части. Процесс решения таких задач на практике делится на промежуточные этапы, на каждом из которых выполняется некоторая частичная оптимизация исходного алгоритма. Каждому из этих этапов соответствует свой промежуточный язык, причем перевод с одного промежуточного языка на другой должен осуществляться достаточно просто. Тогда оптимизацию можно проводить с помощью эквивалентных преобразований алгоритма, полученного на данном этапе с учетом его последующей трансляции на язык очередного этапа. Такой подход был выработан при решении задач автоматизации проектирования ЭВМ и может быть перенесен на задачи, связанные с реализацией в схемном или программном виде средств математического обеспечения ЭВМ [32, 34]. Для выполнения тонких и глубоких эквивалентных преобразований алгоритмов необходимо построить алгебру алгоритмов, которая позволила бы производить эквивалентные преобразования столь же простым и естественным способом, каким они выполняются в обычной алгебре или анализе. Подобный алгебраический аппарат, предназначенный для решения задач, связанных с автоматизацией проектирования ЭВМ и программирования, был предложен В. М. Глушковым [26].

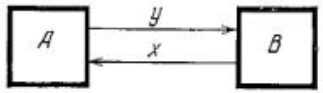


Рис. 17.

Рассматриваемые нами построения эквивалентных преобразований микропрограмм носят общий характер и могут быть распространены на случай произвольных программ и алгоритмов. Приемы формальных преобразований микропрограмм делятся на два класса. Первый класс составляют преобразования, изменяющие лишь форму записи микропрограмм, последовательность выдачи микроопераций в операционное устройство при этом не изменяется. Ко второму классу относятся более глубокие преобразования, изменяющие как форму записи микропрограмм, так и выдаваемые этими микропрограммами последовательности микроопераций.

К основным понятиям рассматриваемой теории относится **абстрактная модель ЭВМ,** которая представляет собой композицию двух автоматов: **операционного** и **управляющего.** Поэтому функционирование каждой ЭВМ в промежутках между обращениями за новой командой может быть описано в виде взаимодействия этих автоматов (рис. 17).

Управляющий автомат A получает от операционного автомата B сигналы x, представляющие собой кортежи $(\alpha_1, \alpha_2, \ldots, \alpha_m)$ значений различных элементарных логических условий, определенных на операционном устройстве. Выходные сигналы y управляющего автомата A отождествляются с микрооперациями: выходной сигнал y осуществляет микрооперацию y в операционном устройстве B. Состояния управляющего автомата отождествляются с командами реализуемых этим автоматом микропрограмм.

Управляющий автомат A обычно представляет собой конечный автомат Мили [24] с относительно небольшим числом состояний (от нескольких сотен до нескольких десятков тысяч). Операционный автомат B является автоматом Мура, состояния которого для микропрограмм интерпретируются как содержимое регистров (или ячеек оперативной памяти, если управляющий автомат реализует программу ЭВМ).

Множество M состояний автомата B называют **информационным.** В отличие от автомата A операционный автомат B имеет огромное число состояний (один 40-разрядный регистр имеет свыше триллиона состояний). Поэтому автомат B естественно считать бесконечным, причем состояния его задаются с помощью бесконечных (в одну или в обе стороны) **абстрактных регистров** (преобразования на бесконечных абстрактных регистрах рассматриваются более подробно в § 4.3, 4.4).

Представив систему микропрограмм в виде автомата, можно применить к ней различные преобразования, используемые в абстрактной теории автоматов [24]. Весьма полезным преобразованием является, в частности, минимизация автомата, позволяющая в ряде случаев существенно уменьшить объем памяти, необходимый для хранения микропрограмм.

Рассмотрим, далее, простейшие преобразования, относящиеся ко второму классу преобразований микропрограмм. Соотношения микроопераций $p_1, \ldots, p_n, r_1, \ldots, r_m$ вида $p_1 p_2 \ldots p_n = r_1 r_2 \ldots r_m$ позволяют заменять одни последовательности микроопераций другими на так называемых линейных участках микропрограмм (при преобразовании программ на соответствующих участках одна последовательность команд заменяется другой). **Линейным** называется такой участок микропрограмм, в котором каждая микрооперация передает управление следующей (переходов, как условных, так и безусловных, нет). При этом в ряде случаев можно вводить несколько уровней управления или изменять разбиение на уровни, произведенное ранее.

Как пример рассмотрим преобразование линейного участка микропрограммы операционного устройства, состоящего из одного бесконечного в обе стороны абстрактного двоичного регистра. Будем считать, что все разряды кода, установленного на регистре, начиная с нулевого и выше, представляют целую часть соответствующего (двоичного) числа, а отрицательные разряды — его дробную часть. Обозначим через p микрооперацию прибавления единицы к содержимому регистра, а через r — микрооперацию

сдвига на один разряд в сторону младших разрядов. Из определения указанных операций легко выводится следующее соотношение:

$$p^2 r = rp. \tag{4.1}$$

Действительно, оператор p^2 в левой части этого соотношения преобразует код x, установленный на регистре, в код $x+2$. Последующее применение операции сдвига r переводит этот код в код $\frac{1}{2}(x+2)$. Аналогичные преобразования операций в правой части соотношения (4.1) переводят код x в код $\frac{1}{2}x+1$. Так как эти коды равны, то справедливость соотношения (4.1) доказана.

Для рассмотрения более общего случая необходим такой аппарат, который позволил бы представить микропрограмму (или программу) в виде формулы в некоторой алгебре и применить обычные алгебраические методы преобразования этих формул.

Для достижения указанной цели каждому операционному автомату B поставим в соответствие две алгебры: **алгебру операторов** $\mathfrak{A}_B$ и **алгебру условий** $\mathfrak{B}_B$. Операторами из $\mathfrak{A}_B$ будут любые однозначные (в общем случае лишь частично определенные) преобразования информационного множества M (состояний автомата B). К операторам относятся, в частности, все микрооперации, определенные на операционном автомате B. Условимся обозначать микрооперации строчными, а операторы (если они не являются микрооперациями рассматриваемого операционного автомата) — прописными буквами латинского алфавита.

Условиями из $\mathfrak{B}_B$ будем называть отображения (в общем случае частично определенные) множества M в двухэлементное множество $\{1, 0\}$. К условиям из $\mathfrak{B}_B$ относятся все элементарные логические условия, значения которых являются выходными сигналами операционного автомата B. Элементарные и неэлементарные условия будем обозначать строчными греческими буквами.

Для операторов из $\mathfrak{A}_B$ помимо обычной операции умножения (последовательного применения операторов) введем еще два типа операций: так называемые α-дизъюнкцию и α-итерацию операторов. Эти операции определяются для каждого условия $\alpha \in \mathfrak{B}_B$.

α-Дизъюнкция операторов P и Q — будем обозначать ее через $(P \overset{\alpha}{\vee} Q)$ — представляет собой новый оператор R, определяемый следующим образом. Для произвольного состояния $m \in M$ $R(m) = P(m)$, если условие α истинно на состоянии m $(\alpha(m) = 1)$, и $R(m) = Q(m)$, если условие $\alpha(m) = 0$. Если $\alpha(m)$ не определено, то неопределенным считается также и $R(m)$. Более наглядно: α-дизъюнкция $(P \overset{\alpha}{\vee} Q)$ представляет собой кусочно-заданный оператор, равный P там, где условие α истинно, и Q — там, где оно ложно.

Результат **α-итерации** оператора P — будем обозначать его через $\{P\}^{\alpha}$ — представляет собой новый оператор S, определяемый следующим образом. Для любого состояния $m \in M$ оператор $S(m)$ равняется первому из элементов ряда $m, P(m), P^2(m), P^3(m), \ldots$, для которого выполняется условие α в предположении, что для всех предшествующих элементов это условие определено. Если такого элемента нет, то значение $S(m)$ не определено. Выполнение оператора α-итерации $\{P\}^{\alpha}$ заключается в проверке условия α и применении оператора P до тех пор, пока это условие остается ложным. Если в результате очередной проверки условия α получено значение 1, то определение значения этого оператора заканчивается. Если условие после очередной проверки окажется неопределенным, то неопределенным будет и значение оператора.

Для условий из $\mathfrak{B}_B$ вводятся обычные операции дизъюнкции, конъюнкции и отрицания, при этом, однако, условия могут быть не определенными. Конъюнкция $\alpha \wedge \beta$ двух условий считается истинной, если оба условия α и β истинны, ложной — если хотя бы одно из них ложно, и не определенной — во всех остальных случаях. Дизъюнкция $\alpha \vee \beta$ считается истинной, если хотя бы одно из условий α или β истинно, ложной — если оба они ложны, и не определенной — во всех остальных случаях. Условие $\bar{\alpha}$ естественно считать неопределенным тогда и только тогда, когда условие α не определено.

Кроме этих операций определяется операция **левого умножения условий на операторы.** Если α — условие, а P — оператор, то произведение $P\alpha$ представляет собой новое условие β, проверка которого эквивалентна проверке условия α после выполнения оператора P. Для любого состояния $m \in M$ $\beta(m) = \alpha(P(m))$. Искомую пару алгебр $\langle \mathfrak{A}_B; \mathfrak{B}_B \rangle$ можно построить теперь следующим образом. Прежде всего в состав алгебры $\mathfrak{A}_B$ включаются (в качестве образующих элементов) все микрооперации автомата B, а в состав алгебры $\mathfrak{B}_B$ — все определенные на множестве M элементарные логические условия. К построенным таким образом множествам $\mathfrak{A}_B^{(0)}$ и $\mathfrak{B}_B^{(0)}$ операторов и условий применяются введенные выше операции. При этом возникают, вообще говоря, новые операторы и условия, которые вместе с уже имеющимися составляют множества $\mathfrak{A}_B^{(1)}$ и $\mathfrak{B}_B^{(1)}$. К этим множествам снова применяется указанный способ образования новых операторов и условий.

Объединив все построенные множества $\mathfrak{A}_B^{(i)}$ и $\mathfrak{B}_B^{(i)}$, получим искомые алгебры $\mathfrak{A}_B$ и $\mathfrak{B}_B$.

Для любого $\alpha \in \mathfrak{B}_B$ α-дизъюнкции и α-итерации операторов из $\mathfrak{A}_B$ содержатся в $\mathfrak{A}_B$. Аналогично все произведения P_α для любых операторов $P \in \mathfrak{A}_B$ содержатся в $\mathfrak{B}_B$. Легко видеть также, что любой оператор $S \in \mathfrak{A}_B$ может быть выражен с помощью введен-

ных выше операций в алгебрах $\mathfrak{A}_B$ и $\mathfrak{B}_B$ через образующие элементы этих алгебр, которыми являются микрооперации и элементарные логические условия автомата B.

Система $\langle \mathfrak{A}_B; \mathfrak{B}_B \rangle$, состоящая из пары алгебр: алгебры операторов $\mathfrak{A}_B$ и алгебры условий $\mathfrak{B}_B$, называется **системой алгоритмических (микропрограммных) алгебр.** Из определения порождающей многоосновной алгебры (см. § 3.6) вытекает следующее утверждение.

Теорема 4.1. *Система алгоритмических алгебр $\langle \mathfrak{A}_B; \mathfrak{B}_B \rangle$ является порождающей многоосновной алгеброй с выделенной в качестве основной компоненты алгеброй операторов $\mathfrak{A}_B$.*

Представление любого оператора из алгебры $\mathfrak{A}_B$ через образующие элементы системы $\langle \mathfrak{A}_B; \mathfrak{B}_B \rangle$ называется **регулярной микропрограммой** этого оператора.

Нетрудно понять, что в общем случае оператор может быть представлен не одной, а многими регулярными микропрограммами, поскольку в алгебре $\mathfrak{A}_B$, как и в алгебре $\mathfrak{B}_B$, существуют различные соотношения ее элементов, например соотношение (4.1). Используя ту или иную систему определяющих соотношений алгебр $\mathfrak{A}_B$ и $\mathfrak{B}_B$, можно осуществлять формальные преобразования регулярных микропрограмм в принципе таким же способом, каким преобразуются выражения в обычной алгебре (см. с. 123—125).

Остановимся теперь на вопросе о степени общности регулярных микропрограмм. Зафиксировав какую-нибудь систему Σ микроопераций и элементарных логических условий автомата B, рассмотрим как регулярные, так и обычные микропрограммы, построенные с помощью системы Σ.

На первый взгляд обычные микропрограммы отличаются большей степенью общности по сравнению с регулярными. В самом деле, в обычных микропрограммах допускаются произвольные условные переходы, а в регулярных все условные переходы задаются операциями α-дизъюнкции и α-итерации. Как следует из определения этих операций, условный переход в представляемых ими выражениях осуществляется лишь в одном направлении — вперед. Скачки назад возможны только в одном случае: от конца итерационных скобок к их началу. Причем все такие скачки выполняются вполне однозначно в зависимости от условия, по которому производится итерация. Тем не менее, несмотря на меньшую общность записи регулярных микропрограмм, оказывается, что они представляют тот же класс операторов, что и обычные микропрограммы. Справедливо следующее утверждение.

Теорема 4.2. *Любая микропрограмма может быть представлена в регулярной форме. Существует алгоритм для преобразования произвольных микропрограмм в регулярную форму.*

Для доказательства этого утверждения рассмотрим произвольную микропрограмму в системе Σ микроопераций и элементарных

логических условий. Как отмечалось выше, эту микропрограмму можно представить в виде конечного автомата. Вводя в случае необходимости дополнительные состояния (в которых не совершаются никакие преобразования над множеством M), все переходы в этом автомате можно выполнять после проверки не более одного условия в каждом состоянии.

Пусть, например, заданы два условия: α и β. В состоянии i проводится проверка этих условий, и в зависимости от их значений осуществляется переход в четыре разных состояния. Комбинации (0, 0), (0, 1), (1, 0) и (1, 1) переводят автомат соответственно в состояния i_1, i_2, i_3, i_4, выдавая при этом микрокоманды p_1, p_2, p_3, p_4. Вводятся дополнительные состояния i' и i''. В состоянии i проверяется лишь условие α, и если оно ложно (равно нулю), то осуществляется переход в состояние i', а если истинно — в состояние i''. Во время этих переходов выдается микрокоманда e тождественного преобразования. В состояниях i' и i'' проверяется условие β и осуществляются переходы в состояния i_1 и i_2 (из состояния i') или в состояния i_3 и i_4 (из состояния i''). Во время этих переходов выдаются соответствующие микрокоманды p_1, p_2, p_3, p_4.

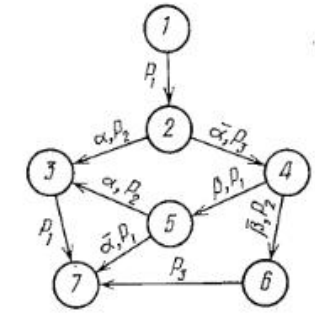


Рис. 18.

Предположим, что рассматриваемый нами автомат не имеет циклов. В таком случае микропрограмма записывается в регулярной форме. Все разветвления в микропрограмме представляются при этом α-дизъюнкциями. Пусть, например, автомат имеет вид, представленный на рис. 18. Состояние 1 соответствует первой команде микропрограммы, а состояние 7 — последней. Эта микропрограмма записывается в регулярной форме

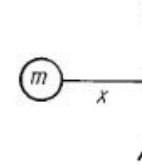


Рис. 19.

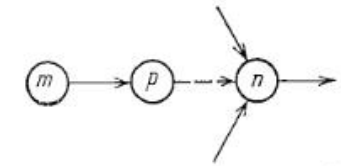


Рис. 20.

$$p_1 \underset{\alpha}{(} p_2 p_1 \vee p_3 \underset{\beta}{(} p_1 \underset{\alpha}{(} p_2 p_1 \vee p_1) \vee p_2 p_3 \underset{\alpha}{)} \underset{\beta}{)}.$$

При наличии циклов воспользуемся идеей Мак-Карти: сведем этот случай к уже известному. Разрежем все циклы, введя в каждом разрезе новое состояние (рис. 19, 20). В каждом автомате (с раз-

резанными или неразрезанными циклами) можно определить условие α_{ij}, при котором автомат, начиная движение из i-го состояния, рано или поздно достигает состояния j. Обозначим через S_{ij} оператор (последовательность микрокоманд), выдаваемый автоматом при переходе от i-го состояния в j-е. Этот оператор может, разумеется, оказаться и неопределенным (в случае недостижимости j-го состояния из i-го). Будем считать, что α_{ij} и S_{ij} определяются для случая, когда соответствующий цикл разрезан (см. рис. 20). Обозначим через α'_{ij} и S'_{ij} условие и оператор для случая склеенного цикла (см. рис. 19). Остальные циклы сохраняются в прежнем состоянии (они могут быть как разрезанными, так и неразрезанными).

Непосредственно из определений условий α_{ij}, α'_{ij} и операторов S_{ij}, S'_{ij} получаем следующие соотношения:

$$\alpha'_{ij} = \alpha_{ij} \vee (\alpha_{ip} \wedge S_{ip} \{S_{np}\}_{\alpha_{nj}} \alpha_{nj}), \tag{4.2}$$

$$S'_{ij} = (S_{ij} \vee S_{ip} \{S_{np}\}_{\alpha_{nj}} S_{nj})_{\alpha_{ij}}. \tag{4.3}$$

Оба эти соотношения отражают тот факт, что после склеивания цикла появляется дополнительная возможность достижения j-го состояния: нужно дойти (в автомате пока еще с разрезанным циклом) от i-го состояния до p-го; затем, пользуясь склеиванием, перескочить в n-е состояние, пройти какое-то число раз по циклу (т. е. в случае разрезанного цикла от n-го состояния до p-го) и из n-го состояния перейти в j-е.

Теперь можно предложить следующий **алгоритм регуляризации** любой микропрограммы. На первом шаге алгоритма представляем заданную микропрограмму в виде автомата. На втором сводим все переходы в автомате к раздвоениям в результате проверки в каждом состоянии не более чем одного условия. На третьем разрезаем все циклы. На четвертом строим описанным выше методом регулярную микропрограмму для оператора S_{1r}, представляемого полученным автоматом без циклов (1 — начальное, r — конечное состояния этого автомата). На пятом шаге, используя соотношения (4.2) и (4.3), один за другим восстанавливаем все разрезанные циклы. Выражение для оператора S_{1r} после восстановления всех циклов и является искомым регулярным представлением исходной микропрограммы. Регулярность этого представления вытекает из регулярности выражения для оператора S_{1r} в случае разрезанных циклов и регулярности правых частей соотношений (4.2) и (4.3).

Описанный алгоритм регуляризации микропрограмм, хотя и прост в обосновании, часто приводит к очень длинным выкладкам. В практическом отношении более удобен алгоритм [26], в котором используется регулярность событий, представимых в автомате [24].

Перейдем к рассмотрению формальных преобразований регулярных микропрограмм. Среди определяющих соотношений построенных выше алгебр (операторов и условий) существуют тождественные, справедливые для любых операторов и условий, и конкретные (индивидуальные), справедливые лишь для некоторых операторов и условий. Примерами тождественных соотношений могут служить соотношения

$$\{P\}_{\alpha} = \{P^2\}_{\alpha \vee P\alpha} (e \vee P)_{\alpha}, \tag{4.4}$$

$$\{P\}_{\alpha} = (e \vee P)_{\mu} \{P^2\}_{\alpha}, \tag{4.5}$$

где $\mu = \{P^2\}_{\alpha \vee P\alpha} \alpha$, P — произвольный оператор, α — произвольное логическое условие, e — тождественный оператор, осуществляющий тождественное отображение множества M. Справедливость соотношений (4.4) и (4.5) вытекает непосредственно из определений операций α-дизъюнкции, α-итерации и умножения условия на оператор.

Далее, пусть P, Q, R — операторы, α — условие, причем выполняются соотношения $PR = RQ$ и $R\alpha = \alpha$. Тогда, как легко видеть, выполняется соотношение

$$\{P\}_{\alpha} R = R \{Q\}_{\alpha}, \tag{4.6}$$

так как $\{P\}_{\alpha}$ представляет собой для каждого конкретного элемента преобразуемого множества некоторую степень $P : \{P\}_{\alpha} = P^n$ $(n = 0, 1, \ldots)$. Из соотношения же $PR = RQ$ следует, очевидно, соотношение $P^n R = RQ^n$. Кроме того, так как оператор R не изменяет условия α, то из последнего равенства следует, что для фиксированного n

$$R \{Q\}_{\alpha} = RQ^n = P^n R = \{P\}_{\alpha} R.$$

Для нахождения конкретных соотношений необходимо фиксировать некоторый операционный автомат B и соответствующую систему микроопераций и элементарных логических условий. Пусть в качестве B выбран операционный автомат, состоящий из k-двоичных бесконечных в обе стороны регистров. Будем считать, что разряды с неотрицательными номерами содержат целую, а с отрицательными — дробную часть соответствующих двоичных кодов. Зафиксируем также следующий набор микроопераций и элементарных логических условий автомата B:

e — тождественное преобразование;

s_{ij} — прибавление содержимого i-го регистра к j-му (код на i-м регистре при этом не изменяется);

p_i — прибавление единицы к содержимому i-го регистра;
l_i — сдвиг на один разряд в сторону старших разрядов на i-м регистре;
r_i — сдвиг на один разряд в сторону младших разрядов на i-м регистре;
o_i — очистка (установка нулевого кода) i-го регистра;
α_i — условие равенства нулю содержимого i-го регистра;
β_i — условие равенства нулю содержимого всех разрядов с неположительными номерами (включая нулевой разряд) на i-м регистре.

Кроме того, обозначим через x^{-1} обратное преобразование любого взаимно-однозначного преобразования x. Например, p_i^{-1} означает вычитание единицы из содержимого i-го регистра. Ясно также, что $r_i = l_i^{-1}$ и $l_i = r_i^{-1}$. Исходя из определения введенных микроопераций, легко установить справедливость следующих соотношений:

$$\left.\begin{aligned} l_i s_{ij} &= s_{ij}^2 l_i, \\ s_{ij} r_i &= r_i s_{ij}^2, \\ l_j s_{ij}^2 &= s_{ij} l_j, \\ r_j s_{ij} &= s_{ij}^2 r_j, \\ l_j p_i^2 &= p_i l_i, \\ l_i p_i^{-2} &= p_i^{-1} l_i, \\ r_i p_i &= p_i^2 r_i, \end{aligned}\right\} \tag{4.7}$$

$$r_i p_i^{-1} = p_i^{-2} r_i, \tag{4.8}$$

$$x_i o_i = o_i. \tag{4.9}$$

Здесь x_i — любое преобразование на i-м регистре. Кроме того, выполняются соотношения коммутативности преобразований на непересекающихся множествах регистров. Например, $s_{12} p_3 = p_3 s_{12}$, $r_1 p_2^{-1} = p_2^{-1} r_1$ и т. д.

Примерами соотношений в алгебре условий являются соотношения

$$\left.\begin{aligned} \alpha_i \vee \beta_i = \beta_i,\ \alpha_i \wedge \beta_i = \alpha_i. \\ l_i \alpha_i = \alpha_i,\ r_i \alpha_i = \alpha_i. \end{aligned}\right\} \tag{4.10}$$

Более сложный характер носит соотношение, справедливое для любых целочисленных кодов на i-м регистре:

$$\underset{\alpha_i \vee p_i^{-1}\alpha_i}{\{} (x p_i^{-1} y)^2 \} \alpha_i = \beta_i. \tag{4.11}$$

где x, y — любые операторы, не изменяющие содержимого i-го

регистра. Справедливость соотношения (4.11) легко установить, если заметить, что β_i представляет собой условие, при котором содержимое i-го регистра является целым четным числом. Оператор в левой части этого соотношения переводит, как легко видеть, любое целое число n на i-м регистре в нуль (если n четное) и единицу (если n нечетное). Таким образом, выполнение условия α_i после преобразования означает, что первоначальное число, хранившееся на регистре, было четным.

Для иллюстрации формальных преобразований с помощью выписанных соотношений рассмотрим микропрограмму умножения целых неотрицательных двоичных чисел x и z, установленных соответственно на первом и третьем регистрах, такую, что результат xz запоминается на втором регистре. Более точно, оператор Q умножения должен перевести три указанных регистра из состояния (x, y, z) в состояние $(0, xz, 0)$. Используя определение умножения как последовательного сложения, приходим к следующей регулярной микропрограмме оператора Q:

$$Q = o_2 \underset{\alpha_3}{\{} s_{12} p_3^{-1} \} o_1 o_3. \tag{4.12}$$

Эта микропрограмма осуществляет последовательное сложение первого сомножителя $0 + x + x + \cdots$ с самим собой, одновременно вычитая каждый раз единицу из второго сомножителя до тех пор, пока последний не обратится в нуль. Легко видеть, что число N микроопераций, выполняемых этой микропрограммой, зависит от величины второго сомножителя z и определяется формулой $N = 2z + 3$. При больших z микропрограмма будет, разумеется, чрезвычайно медленной.

Применяя к микропрограмме (4.12) соотношение (4.9) и соотношения коммутативности преобразований на разных регистрах, приводим ее к виду

$$Q = o_2 \underset{\alpha_3}{\{} s_{12} p_3^{-1} \} l_1 r_3 o_1 o_3.$$

С помощью соотношений коммутативности и соотношений (4.5) и (4.11) микропрограмму можно привести к виду

$$Q = o_2 \underset{\beta_3}{(} e \vee s_{12} p_3^{-1} \underset{\alpha_3}{)} \{ s_{12}^2 p_3^{-2} \} l_1 r_3 o_1 o_3.$$

Применяя соотношения (4.6)—(4.8), (4.10) и соотношения коммутативности, получаем

$$Q = o_2 \underset{\beta_3}{(} e \vee s_{12} p_3^{-1} \underset{\alpha_3}{)} l_1 r_3 \{ s_{12} p_3^{-1} \} o_1 o_3.$$

В результате выполненных преобразований число повторений оператора $s_{12} p_3^{-1}$ внутри цикла уменьшается вдвое (точнее, изменя-

ется от n к $\left[\frac{n}{2}\right]$). Повторяя указанные преобразования k раз, приводим микропрограмму к виду

$$Q = o_2 \Big(\underset{\beta_3}{(} e \vee s_{12} p_3^{-1}) \, l_1 r_3 \Big)^{\overset{k}{}} \underset{\alpha_3}{\{} s_{12} p_3^{-1} \} \, o_1 o_3.$$

Таким образом, мы получили новый цикл с оператором $(e \vee s_{12} p_3^{-1}) \, l_1 r_3$. При $k > \log_2 n$ второй цикл исчезает (сводится к тождественному оператору). Это неравенство выполняется (т. е. второй оператор исчезает), очевидно, при условии α_i, где $i = 3$. В результате получаем окончательную запись для преобразованной микропрограммы:

$$Q = o_2 \underset{\alpha_3}{\{} \underset{\beta_3}{(} e \vee s_{12} p_3^{-1}) \, l_1 r_3 \} \, o_1 o_3. \qquad (4.13)$$

Нетрудно видеть, что в таком виде микропрограмма описывает известный алгоритм позиционного умножения двоичных чисел, обычно применяющийся в ЭВМ с параллельными арифметическими устройствами. При перемножении больших чисел микропрограмма в виде (4.13) значительно эффективнее исходной микропрограммы (4.12).

Рассмотренный пример иллюстрирует методику формальных преобразований микропрограмм. Аналогично можно было бы выполнить и другие преобразования, например преобразование микропрограммы деления. Из вида, основанного на определении деления (как последовательного вычитания делителя из делимого), ее можно преобразовать к виду, обычно используемому в ЭВМ.

Подобные глубокие преобразования являются, как правило, наиболее эффективным средством оптимизации микропрограмм, однако их реализация в каждом конкретном случае может оказаться сложной. Вместе с тем в принципе менее эффективные преобразования линейных участков микропрограмм (описанные выше) просты в применении и сравнительно легко поддаются автоматизации. В результате их общий вклад в оптимизацию микропрограмм на практике оказывается весьма значительным.

Метод, положенный в основу доказательства теоремы 2, дает возможность регуляризовать, т. е. представить в соответствующей системе алгоритмических алгебр, произвольный алгоритм, и в частности любую микропрограмму или программу вычислительной машины. В этом состоит его фундаментальное значение в прикладной теории алгоритмов. Дальнейшее изучение абстрактной автоматной модели ЭВМ привело к созданию теории дискретных преобразователей [35, 53, 72], которая находит важные приложения в практике программирования. Анализ основных конструкций, входящих в сигнатуру операций алгоритмических алгебр, подтверждает их полное соответствие концепции структурного программирования [41]. Таким образом, в отличие от традиционной теории схем программ [45, 47, 65, 136] системы алгоритмических алгебр представляют собой теорию схем структурированных алгоритмов и программ.

В работах [116, 117, 121] в аппарат алгоритмических алгебр введены средства параллелизма и техники фильтров, которые могут быть использованы в качестве гибкого механизма управления процессом вычислений. На основе предложенного подхода нами разработана (см. § 4.2) методика формальных преобразований выражений алгоритмических алгебр к каноническим формам по аналогии с теорией д. н. ф. (к. н. ф.) [24, 49]. Полученные результаты находят применение при изучении проблемы аксиоматизации алгоритмических алгебр [58, 121] (см. также обзор [37]). Средства параллелизма позволяют ориентировать теорию алгоритмических алгебр на приложения, связанные с формализацией параллельных вычислений [63, 84] и решением других важных задач, возникающих с развитием систем математического обеспечения многопроцессорных вычислительных комплексов [44, 62, 82, 92].

§ 4.2. Тождественные соотношения в системе алгоритмических алгебр

Как отмечалось в § 4.1, сигнатура операций системы алгоритмических алгебр $\langle \mathfrak{A}_B; \mathfrak{V}_B \rangle$ содержит такие традиционно программистские средства, как условный переход (α-дизъюнкция), цикл (α-итерация), средства для составления сложных логических условий (операции над алгеброй условий) и др. Поэтому изучение алгебраических свойств операций в алгоритмических алгебрах и разработка для них техники тождественных преобразований представляют значительный интерес в связи с вопросами оптимизации программ по заданным критериям. В данном параграфе предлагается некоторое расширение сигнатуры операций системы $\langle \mathfrak{A}_B; \mathfrak{V}_B \rangle$, не нарушающее программистской ориентации аппарата алгоритмических алгебр. Приводятся тождественные соотношения, отражающие свойства операций системы $\langle \mathfrak{A}_B; \mathfrak{V}_B \rangle$, и рассматриваются канонические формы представлений для условий из $\mathfrak{V}_B$, аналогичные каноническим формам (д. н. ф., к. н. ф.) алгебры логики (см. § 3.4). Методическая близость предлагаемого подхода к решению одноименных проблем в алгебре логики позволяет надеяться на его естественность и удобство в применении.

Пусть $M = \{m_i \mid i \in I\}$ — информационное множество состояний операционного автомата B: $X = \{\alpha_1, \alpha_2, \ldots, \alpha_n\}$ и $Y = \{A_1, A_2, \ldots, A_m\}$ — совокупности элементарных соответственно условий

и операторов, P, Q, $R \in \mathfrak{A}_B$ — произвольные операторы, α, β, $\gamma \in \mathfrak{B}_B$ — произвольные условия.

Дополним сигнатуру алгоритмических алгебр операциями дизъюнкции $P \cup Q$, параллельного применения операторов P, Q и фильтрации, порождающей операторы-фильтры,

$$\underline{\alpha} = \begin{cases} E, & \text{если } \alpha = 1, \\ N, & \text{в противном случае,} \end{cases}$$

где E — тождественный, N — неопределенный операторы. Операция дизъюнкции $P \cup Q$ определена лишь в случае, когда операторы P и Q функционируют однозначно в согласованном вычислительном процессе. Более точно дизъюнкция $P \cup Q$ определена в состоянии $m \in M$, если операторы P и Q выдают в этом состоянии одну и ту же последовательность элементарных операторов, $\tilde{y} = A_{i_1} A_{i_2} \ldots A_{i_k}$, либо на некотором промежуточном этапе (при наличии одной и той же истории вычислений, $A_{i_1} A_{i_2} \ldots A_{i_r}$, $1 \leqslant r \leqslant k$) один из операторов прекращает функционировать, а другой выдает последовательность $\tilde{y}$.

Операция дизъюнкции ассоциативна и коммутативна:

$$P \cup (Q \cup R) = (P \cup Q) \cup R, \quad P \cup Q = Q \cup P.$$

Дизъюнкция также удовлетворяет следующим тождествам:

$$P(Q \cup R) = PQ \cup PR, \quad (P \cup Q) R = PR \cup QR, \quad P \cup P = P.$$

Операция умножения $P \cdot Q$ (последовательное применение операторов P и Q) ассоциативна,

$$(P \cdot Q) \cdot R = P \cdot (Q \cdot R),$$

и некоммутативна, причем выполняются тождества

$$P \cdot E = E \cdot P = P, \quad P \cdot N = N \cdot P = N.$$

Следовательно, операторы E и N играют роль констант в алгебре $\mathfrak{A}_B$.

Операторы-фильтры $\underline{\alpha}$ в зависимости от значений условий α осуществляют управление процессом вычислений, представленным соответствующим регулярным выражением. Иными словами, фильтр $\underline{\alpha}$ обращается в тождественный оператор E, не нарушая работы алгоритма, если условие α истинно в данном состоянии. Если в указанном состоянии $\alpha \neq 1$, фильтр $\underline{\alpha}$ обращается в N, обрывая соответствующую ветвь вычислений.

Для фильтров выполняются следующие тождества:

$$\underline{\alpha}\underline{\beta} = \underline{\alpha\beta}, \tag{4.14}$$

$$\underline{\alpha \vee \beta} = \underline{\alpha} \cup \underline{\beta}. \tag{4.15}$$

Из (4.14), (4.15), учитывая коммутативность конъюнкции и дизъюнкции, следует

$$\underline{\alpha}\,\underline{\beta} = \underline{\beta}\,\underline{\alpha}; \quad \underline{\alpha} \cup \underline{\beta} = \underline{\beta} \cup \underline{\alpha}.$$

Обобщенные булевы операции $\vee$, $\wedge$, $-$ (см. § 4.1) могут быть заданы таблицей истинности (табл. 4); здесь и далее предполагается, что на множестве $E_3 = \{0, \mu, 1\}$ логических значений условий алгебры $\mathfrak{B}$ введено отношение порядка $<$ так, что $0 < \mu < 1$, где 0 — значение лжи, μ — неопределенности, 1 — истины. Далее знак конъюнкции $\wedge$ часто опускается.

Таблица 4

$\alpha \wedge \beta$	0	μ	1	$\alpha \vee \beta$	0	μ	1	α	$\bar{\alpha}$
0	0	0	0	0	0	μ	1	0	1
μ	0	μ	μ	μ	μ	μ	1	μ	μ
1	0	μ	1	1	1	1		1	0

Следовательно, они удовлетворяют всем законам булевой алгебры (см. § 2.5, а также § 1.2), кроме закона исключенного третьего $\alpha \vee \bar{\alpha} = 1$ и закона противоречия $\alpha \wedge \bar{\alpha} = 0$. Это означает, в частности, что любое выражение трехзначной логики Φ_{E_3}, построенное с помощью указанных операций из элементарных логических условий $\alpha_1, \alpha_2, \ldots, \alpha_n$, может быть преобразовано к д. н. ф. или к. н. ф. (см. § 3.4).

Пусть теперь $U_1 \vee U_2 \vee \cdots \vee U_k$ — произвольная д. н. ф. трехзначной логики, где $U_i = \tilde{\alpha}_{i_1} \tilde{\alpha}_{i_2} \ldots \tilde{\alpha}_{i_{r_i}}$ $\left(\tilde{\alpha}_{i_j} = \begin{Bmatrix} \bar{\alpha}_{i_j} \\ \alpha_{i_j} \end{Bmatrix}\right.$, $j = 1, 2, \ldots, r_i$; $i = 1, 2, \ldots, k$). Тогда справедливо такое соотношение:

$$\underline{\bigvee_{i=1}^{k} U_i} = \bigcup_{i=1}^{k} \underline{\tilde{\alpha}_{i_1} \tilde{\alpha}_{i_2} \ldots \tilde{\alpha}_{i_{r_i}}}. \tag{4.16}$$

Используя фильтры и дизъюнкцию операторов, можно представить α-дизъюнкцию в виде

$$(P \underset{\alpha}{\vee} Q) = \underset{\alpha}{P} \cup \underset{\bar{\alpha}}{Q}. \tag{4.17}$$

Такое представление, учитывая (4.16), позволяет разложить α-дизъюнкцию по элементарным фильтрам.

Пример 1. Пусть задана α-дизъюнкция $(P \underset{\alpha}{\vee} Q)$, где $\alpha = \overline{(\bar{a}_1 \vee a_2)(\bar{a}_1 \vee a_3) \vee a_2}$. Применяя правило де Моргана, преобразуем условие α:

$$\alpha = \overline{(\bar{a}_1 \vee a_2)(\bar{a}_1 \vee a_3)} \cdot \bar{a}_2 = \overline{(\bar{a}_1 \vee a_2} \vee \overline{\bar{a}_1 \vee a_3)}\, \bar{a}_2 = \\ = (a_1 \bar{a}_2 \vee a_1 \bar{a}_3)\, \bar{a}_2.$$

Используя законы дистрибутивности, идемпотентности и правила поглощения, получаем $\alpha = a_1 \bar{a}_2 \vee a_1 \bar{a}_2 \bar{a}_3 = a_1 \bar{a}_2$. Наконец, вследствие (4.17) справедливо соотношение

$$(P \underset{\alpha}{\vee} Q)\, \underline{a_1 \bar{a}_2} P \cup \overline{\underline{a_1 \bar{a}_2}} Q = \underline{a_1 \bar{a}_2} P \cup (\underline{\bar{a}_1} \cup \underline{a_2})\, Q.$$

Дизъюнкция и фильтры позволяют сформулировать соотношения для α-итерации. Предварительно заметим, что из определения α-итерации следует справедливость тождеств

$$\underset{1}{\{P\}} = E, \quad \underset{0}{\{P\}} = \underset{\mu}{\{Q\}} = N, \quad \underset{\alpha\alpha}{\{\{P\}\}} = \underset{\alpha}{\{P\}}, \tag{4.18}$$

$$\left.\begin{aligned} \underset{\alpha}{\{P\}} &= \underset{\alpha}{\{\underline{\bar{\alpha}} P\}}, \\ \underset{\alpha}{\{P\}} &= \underline{\alpha} \cup \underline{\bar{\alpha}} P\, \underset{\alpha}{\{P\}}. \end{aligned}\right\} \tag{4.19}$$

Кроме того, для α-итерации выполняется совокупность соотношений

$$\underset{\alpha}{\{P\}} = \underset{\alpha}{\{(\underline{\bar{\alpha}} P)^n\}} \cup \underline{\bar{\alpha}} P\, \underset{\alpha}{\{(\underline{\bar{\alpha}} P)^n\}} \cup \cdots \cup (\underline{\bar{\alpha}} P)^{n-1} \cdot \underset{\alpha}{\{(\underline{\bar{\alpha}} P)^n\}} \tag{4.20}$$

при любом $n = 1, 2, \ldots$. Действительно, пусть в некотором состоянии $m \in M$ α-итерация $\underset{\alpha}{\{P\}}$ выполняется так, что α становится истинным после применения оператора P k раз. Тогда в силу (4.18) левая часть тождества (4.20) выдает последовательность $(\underline{\bar{\alpha}} P)^k$. Если при делении числа k на n получается остаток r, то все ветви правой части равенства (4.20) обращаются в N, кроме $(\underline{\bar{\alpha}} P)^r \underset{\alpha}{\{(\underline{\bar{\alpha}} P)^n\}}$, которая выдает ту же последовательность операторов.

Справедливо также соотношение, предложенное в работе [50] при рассмотрении системы алгоритмических алгебр с некоммутативной алгеброй условий:

$$P \underset{\alpha}{\{QP\}} = \underset{(P\alpha)}{\{PQ\}}\, P. \tag{4.21}$$

Из этого тождества в результате подстановки вместо операторов Q последовательности P^{n-1} следует справедливость соотношения $P \underset{\alpha}{\{P^n\}} = \underset{(P\alpha)}{\{P^n\}}\, P$ при любых $n = 1, 2, \ldots$. (Предполагается, что при $n = 1$ $P^{(0)} = E$).

При изучении проблемы тождеств в системе алгоритмических алгебр $\langle \mathfrak{A}_B; \mathfrak{B}_B \rangle$ представляет интерес соотношение

$$\underset{\alpha\beta}{\{\{P\}}\, \underset{(\alpha\vee\gamma)}{\{Q\}\}} = \underline{\alpha} \cup \underline{\bar{\alpha}}\, \underset{\beta}{\{P\}}\, \underset{\alpha\beta}{\{\{P\}}\, \underset{(\alpha\vee\gamma)}{\{Q\}\}}. \tag{4.22}$$

Суть тождества (4.22) состоит в том, что в отличие от (4.19) в нем за знак итерации можно выносить (при определенных условиях) не только весь оператор, стоящий в итерации, но и его часть.

С помощью фильтров и дизъюнкции операторов можно анализировать α-итерации по сложным условиям. Например, справедливо тождество

$$\underset{(\alpha\vee\beta)}{\{P\}} = \underset{\alpha}{\{\underline{\bar{\beta}} P\}} \cup \underset{\beta}{\{\underline{\bar{\alpha}} P\}}. \tag{4.23}$$

Действительно, пусть левая часть тождества (4.23) определена в некотором состоянии $m \in M$ и выдает последовательность операторов P^k. Это означает, что i-кратное применение оператора P к состоянию m $(0 \leqslant i < k)$ оставляет условия α и β ложными, тогда как последовательность P^k по крайней мере одно из них обращает в истину. В соответствии с этим правая часть тождества (4.23) выдает ту же последовательность по крайней мере по одной из своих ветвей. Если, например, $\alpha = 1$, а $\beta \neq \alpha$, то первая ветвь выдает последовательность P^k, а вторая ветвь становится неопределенной. Из тождества (4.23) в силу $\underset{(\mu\beta)}{\{P\}} = N$ для любых $\beta \in \mathfrak{B}_B$ и $P \in \mathfrak{A}_B$ следует $\underset{(\alpha\vee\mu\beta)}{\{P\}} = \underset{\alpha}{\{\underline{\bar{\beta}} P\}}$. Это позволяет в итерационных условиях исключать дизъюнктивные члены, имеющие в качестве сомножителя константу неопределенности μ.

Анализ итерации $\underset{\alpha\beta}{\{P\}}$ указывает на необходимость введения дополнительных средств, позволяющих прерывать итерации по одному из условий, например по α, для проверки другого, если условие α принимает значение μ. Заметим, что система операций $(\wedge, \vee, -)$ функционально неполная, так как из приведенных

выше таблиц истинности следует, что каждая из этих операций сохраняет константу μ. Поэтому далее в качестве основных операций алгебры условий $\mathfrak{B}_B$ рассматриваются булевы операции, дополненные тремя унарными операциями[1]: $\alpha^{0(1)}$, $\alpha^{\mu(1)}$, $\alpha^{1(1)}$ (табл. 5).

Таблица 5

α	$\alpha^{0(1)}$	$\alpha^{\mu(1)}$	$\alpha^{1(1)}$
0	1	0	0
μ	0	1	0
1	0	0	1

Операции $\alpha^{a(1)}$, $a = 0, \mu, 1$ удовлетворяют следующим соотношениям:

$$\alpha = \alpha^{1(1)} \vee \mu\alpha^{\mu(1)},$$
$$\bar{\alpha} = \alpha^{0(1)} \vee \mu\alpha^{\mu(1)}.$$

Таким образом, операция отрицания является производной от операций $\vee$, $\wedge$, $\alpha^{a(1)}$. Заметим, что операции $\alpha^{a(1)}$ не нарушают программистской ориентации аппарата алгоритмических алгебр, связанной с выполнением условных переходов и циклов, ввиду справедливости тождеств

$$\underset{\alpha}{(P \vee Q)} = \underbrace{\alpha^{1(1)}P} \cup \underbrace{\alpha^{0(1)}Q}; \quad \underset{\alpha}{\{P\}} = \underset{\alpha^{1(1)}}{\{\alpha^{0(1)}P\}}.$$

К производным операциям алгебры относятся также унарные операции $\alpha^{0(0)} = \overline{\alpha^{0(1)}}$, $\alpha^{\mu(0)} = \overline{\alpha^{\mu(1)}}$, $\alpha^{1(0)} = \overline{\alpha^{1(1)}}$. На множестве $E_2 = \{0, 1\}$ операции $\wedge$, $\vee$, — совпадают с соответствующими операциями двузначной логики. В то же время унарные операции $\alpha^{a(c)}$ ($a = 0, \mu, 1$; $c = 0, 1$) на множестве E_2 обращаются в α или $\bar{\alpha}$ либо вырождаются в булевы константы 0, 1. Отсюда следует, что на множестве P_{E_2} условий, принимающих значения из E_2, выполняются все известные тождества двузначной алгебры Буля.

Введение в алгебру $\mathfrak{B}_B$ унарных операций $\alpha^{a(c)}$ позволяет анализировать α-итерации по сложным условиям, и в частности по конъюнкции:

$$\underset{\alpha\beta}{\{P\}} = \underset{\alpha}{\{P \underset{\beta^{0(0)}}{\{P\}}\}} \underbrace{\beta} \cup \underset{\beta}{\{P \underset{\alpha^{0(0)}}{\{P\}}\}} \underbrace{\alpha}. \tag{4.24}$$

Попадая в одну из ветвей правой части, например в первую, применяем оператор P до тех пор, пока условие α не станет истинным при $\beta = 1$ или $\beta = \mu$. Если $\alpha = 1$, $\beta = \mu$, данная ветвь обратится в N. Одновременно вторая ветвь либо выдаст ту же последовательность операторов, что и первая, либо обратится в N.

В алгебре условий $\mathfrak{B}_B$ выполняется аналог законов исключенного третьего и противоречия

$$\alpha^{0(1)} \vee \alpha^{\mu(1)} \vee \alpha^{1(1)} = 1, \quad \alpha^{0(1)} \cdot \alpha^{1(1)} = \alpha^{0(1)} \cdot \alpha^{\mu(1)} = \alpha^{1(1)} \cdot \alpha^{\mu(1)} = 0.$$

[1] Б. А. Трахтенбротом замечено, что операции $\alpha^{a(1)}$ определяются предикатом $\alpha = a$, где $a = 0, \mu, 1$.

Операции $\alpha^{a(c)}$ могут быть опущены на элементарные условия с помощью следующей совокупности тождеств:

$$\left.\begin{aligned} (\bar{\alpha})^{a(c)} &= \alpha^{\bar{a}(c)}, \\ (\alpha^{a(c)})^{c_1(c_1)} &= \alpha^{a(c)}, \\ (\alpha^{a(c)})^{\bar{c}_1(c_1)} &= \alpha^{a(\bar{c})}, \\ (\alpha^{a(c)})^{\mu(c_1)} &= \bar{c}_1, \end{aligned}\right\} \tag{4.25}$$

где $\bar{0} = 1$, $\bar{\mu} = \mu$, $\bar{1} = 0$. Применение четного числа операций вида $(\,)^{\bar{c}(c)}$ к условию $\alpha^{a(c_1)}$ не изменяет этого условия, а нечетное число таких операций переводит его в $\alpha^{a(\bar{c}_1)}$. Справедливы также соотношения

$$\left.\begin{aligned} \alpha^{c(0)} &= \alpha^{\mu(1)} \vee \alpha^{\bar{c}(1)}, \quad \alpha^{c(1)} = \alpha^{\mu(0)} \cdot \alpha^{\bar{c}(0)}, \\ \alpha^{\mu(0)} &= \alpha^{c(1)} \vee \alpha^{\bar{c}(1)}, \quad \alpha^{\mu(1)} = \alpha^{c(0)} \cdot \alpha^{\bar{c}(0)}. \end{aligned}\right\} \tag{4.26}$$

Унарные операции $\alpha^{a(c)}$ могут быть внесены в дизъюнкции (конъюнкции) на основании тождеств

$$\left\{\begin{aligned} &(\alpha \vee \beta)^{c(c)} = \alpha^{c(c)} \vee \beta^{c(c)}, \quad (\alpha\beta)^{c(c)} = \alpha^{c(c)}\beta^{c(c)}, \\ &(\alpha \vee \beta)^{\bar{c}(c)} = \alpha^{\bar{c}(c)}\beta^{\bar{c}(c)}, \quad (\alpha\beta)^{\bar{c}(c)} = \alpha^{\bar{c}(c)} \vee \beta^{\bar{c}(c)}, \\ &(\alpha \vee \beta)^{\mu(1)} = \alpha^{\mu(1)}\beta^{\mu(1)} \vee \alpha^{\bar{c}(c)}\beta^{\mu(1)} \vee \alpha^{\mu(1)}\beta^{\bar{c}(c)}, \\ &(\alpha\beta)^{\mu(0)} = (\alpha^{\mu(0)} \vee \beta^{\mu(0)})(\alpha^{\bar{c}(c)} \vee \beta^{\mu(0)})(\alpha^{\mu(0)} \vee \beta^{\bar{c}(c)}), \\ &(\alpha \vee \beta)^{\mu(0)} = (\alpha^{\mu(0)} \vee \beta^{\mu(0)})(\alpha^{c(c)} \vee \beta^{\mu(0)})(\alpha^{\mu(0)} \vee \beta^{c(c)}), \\ &(\alpha\beta)^{\mu(1)} = \alpha^{\mu(1)}\beta^{\mu(1)} \vee \alpha^{c(c)}\beta^{\mu(1)} \vee \alpha^{\mu(1)}\beta^{c(c)}. \end{aligned}\right. \tag{4.27}$$

Операция $A\alpha$ левого умножения условия на оператор удовлетворяет следующим тождествам:

$$A\mu = \mu, \quad N\alpha = \mu, \quad \overline{A\alpha} = A\bar{\alpha}. \tag{4.28}$$

Заметим, что $c \vee Ac = c$, $c(Ac) = c$, где c — const, A — всюду определенный оператор.

Справедливы также тождества

$$A\alpha = {}_A\Delta(A\alpha) \vee {}_A\bar{\Delta}\mu, \tag{4.29}$$

где ${}_A\Delta$ — характеристическое условие, истинное, если оператор A определен в данном состоянии, и ложное в противном случае:

$$A(\alpha\beta) = (A\alpha)(A\beta), \tag{4.30}$$

$$A(\alpha \vee \beta) = (A\alpha) \vee (A\beta). \tag{4.31}$$

С помощью определенных выше тождеств (4.24)—(4.31) можно доказать справедливость соотношений

$$A \cdot 1 = {}_A\Delta \vee \mu, \quad A0 = {}_A\bar{\Delta}\mu, \tag{4.32}$$

$$\left.\begin{aligned} {}_A\bar{\Delta}A\alpha = {}_A\bar{\Delta}\mu, \quad {}_A\Delta \vee A\alpha = {}_A\Delta \vee \mu, \\ {}_A\Delta\,(A \cdot 1) = {}_A\Delta, \quad {}_A\bar{\Delta} \vee (A0) = {}_A\bar{\Delta}, \end{aligned}\right\} \tag{4.33}$$

$$(A\alpha)^{\alpha(c)} = {}_A\Delta\,(A\alpha^{\alpha(c)}) \vee {}_A\bar{\Delta}\,(\mu)^{\alpha(c)}. \tag{4.34}$$

Как и в алгебре логики, доказательство этих тождеств может быть проведено с помощью непосредственного вычисления значений истинности их левых и правых частей в зависимости от истинности значений переменных и характеристических функций, из которых они составлены. Кроме того, допускаются преобразования выражений, основанные на доказанных тождествах. Например, если в тождестве (4.34) определен оператор A, левая часть тождества принимает то же значение, что и первый дизъюнктивный член в его правой части, тогда как ${}_A\bar{\Delta}$, а значит, и второй член дизъюнкции в этом случае обращается в нуль. В то же время, если оператор A не определен в данном состоянии, то левая и правая части тождества совпадают с условием $(\mu)^{\alpha(c)}$. Следовательно, тождество (4.34) справедливо.

Пусть $f(\alpha_1, \alpha_2, \ldots, \alpha_n; A_1, A_2, \ldots, A_m)$ — логическая функция, составленная из элементарных условий $\tilde{X} = \{\alpha_1, \ldots, \alpha_n\}$ и переменных операторов $A_1, \ldots, A_m$ с помощью операций алгебры $\mathfrak{B}_B$. Тождества (4.25), (4.27), (4.28), (4.34) позволяют опустить унарные операции непосредственно на элементарные условия.

Функция $f^{(-)}(\alpha_1, \ldots, \alpha_n; A_1, \ldots, A_m)$ **инверсна** к функции $f(\alpha_1, \alpha_2, \ldots, \alpha_n; A_1, \ldots, A_m)$, если $f^{(-)}(\alpha_1, \alpha_2, \ldots, \alpha_n; A_1, \ldots, A_m) = f(\bar{\alpha}_1, \bar{\alpha}_2, \ldots, \bar{\alpha}_n; A_1, \ldots, A_m)$.

Условие, входящее в α-итерацию некоторого оператора или в фильтр, называется операторным.

Функция $f^*(\alpha_1, \ldots, \alpha_n; A_1, \ldots, A_m)$ **двойственна** к функции $f(\alpha_1, \ldots, \alpha_n; A_1, \ldots, A_m)$, если она может быть получена из функции f в результате замены 1) каждого операторного условия, входящего в f, соответствующим инверсным условием; 2) каждого вхождения операции $\vee$ в неоператорные условия функции f операцией $\wedge$, и наоборот; 3) каждого вхождения символа 1 в неоператорные условия функции f символом 0, и наоборот; 4) каждого вхождения характеристических условий ${}_{A_i}\Delta$ в неоператорные условия, входящие в f, условиями ${}_{A_i}\bar{\Delta}$ $(i = 1, 2, \ldots, m)$, и наоборот. Примером двойственных выражений в алгебре $\mathfrak{B}_B$ может служить любая из пар тождеств (4.26), (4.27).

Теорема 4.3 (принцип двойственности). *Пусть*

$$f = g(\alpha_1, \ldots, \alpha_n; A_1, \ldots, A_m) \tag{4.35}$$

является произвольным тождеством алгебры $\mathfrak{B}_B$, *составленным из элементарных условий* $\alpha_1, \ldots, \alpha_n$ *и переменных операторов* $A_1, \ldots, A_m$. *Тогда выражение*

$$f^* = g^*(\alpha_1, \ldots, \alpha_n; A_1, \ldots, A_m)$$

также является тождеством алгебры $\mathfrak{B}_B$.

Доказательство. Ввиду справедливости тождества (4.35) при любых значениях элементарных условий $\alpha_1, \ldots, \alpha_n$ в результате подстановки вместо условия α_i отрицания $\bar{\alpha}_i$ для каждого $i = 1, 2, \ldots, n$ получаем новое тождество

$$f = g(\bar{\alpha}_1, \ldots, \bar{\alpha}_n; A_1, \ldots, A_m) \tag{4.36}$$

в алгебре $\mathfrak{B}_B$. Инвертируя левую и правую части тождества (4.36), получаем

$$\bar{f} = \bar{g}(\bar{\alpha}_1, \ldots, \bar{\alpha}_n; A_1, \ldots, A_m).$$

Используя правила де Моргана, закон двойного отрицания, а также тождества (4.25), (4.28), опускаем знак отрицания на элементарные неоператорные условия. В результате получаем тождество

$$f^* = g^*(\alpha_1, \ldots, \alpha_n; A_1, \ldots, A_m),$$

двойственное тождеству (4.35). Теорема доказана.

Аналогично теории д. н. ф. (к. н. ф., см. § 3.4) булевой алгебры введем основные понятия и определения, связанные с каноническими нормальными формами в алгебре $\mathfrak{B}_B$. Предварительно рассмотрим формы представлений функций в алгебре Φ_{E_3} — алгебре трехзначной логики с логическими операциями $\vee$, $\wedge$, $\alpha^{\alpha(1)}$ $(\alpha = 0, \mu, 1)$.

Пусть $\tilde{X} = \{\alpha_1, \alpha_2, \ldots, \alpha_n\}$ — элементарные условия. Конъюнкция $V_i = k_i\hat{\alpha}_{i_1}\hat{\alpha}_{i_2} \ldots \hat{\alpha}_{i_r}$ $(r \leqslant n)$, где $k_i = 1, \mu$, $\hat{\alpha}_{i_j} = \alpha_{ij}^{\alpha(1)}$ $(\alpha = 0, \mu, 1)$, называется **элементарной**, если символы $\alpha_{i_j} \in \tilde{X}$, входящие в V_i, попарно-различны. Константу 1 также будем считать элементарной конъюнкцией, рассматривая конъюнкцию пустого множества букв. **Элементарная дизъюнкция** W_i определяется двойственным образом как дизъюнкция конечного множества попарно-различных условий, принадлежащих набору $\tilde{X}$ с навешенными на них унарными операциями:

$$W_i = t_i \vee \hat{\alpha}_{i_1} \vee \cdots \vee \hat{\alpha}_{i_r},$$

где $r \leqslant n$, $t_i = 0, \mu$. По определению константа 0 рассматривается как элементарная дизъюнкция пустого множества букв.

Дизъюнкция конечного числа попарно-различных и не поглощающих друг друга элементарных конъюнкций называется

дизъюнктивной нормальной формой (д. н. ф.) в алгебре $\mathfrak{B}_B$. Двойственным образом определяется **конъюнктивная нормальная форма** (к. н. ф.).

Элементарная конъюнкция

$$K_i = \alpha_1^{a_{i_1}(1)}\alpha_2^{a_{i_2}(1)} \ldots \alpha_n^{a_{i_n}(1)},$$

включающая в себя все буквы из набора $\tilde{X}$, называется **конституэнтой 1**. Каждому набору $\tilde{s}_i = (a_{i_1}, a_{i_2}, \ldots, a_{i_n})$ значений переменных, входящих в $\tilde{X}$, соответствует конституэнта 1:

$$K(\tilde{s}_i) = \alpha_1^{a_{i_1}(1)}\alpha_2^{a_{i_2}(1)} \ldots \alpha_n^{a_{i_n}(1)},$$

причем разным наборам $\tilde{s}_1$, $\tilde{s}_2$ соответствуют разные конституэнты 1: $K(\tilde{s}_1)$, $K(\tilde{s}_2)$. Число всех наборов значений переменных, входящих в $\tilde{X}$, составляет 3^n, следовательно, и число всех конституэнт 1, соответствующих указанным наборам, также равно 3^n.

Заметим, что всякая конституэнта 1, $K(\tilde{s}) = \alpha_1^{a_{i_1}(1)}\alpha_2^{a_{i_2}(1)} \ldots \alpha_n^{a_{i_n}(1)}$, принимает значение единицы лишь на соответствующем наборе $\tilde{s} = (a_1, \ldots, a_n)$, обращаясь в нуль на всех остальных наборах.

Пусть $f(\alpha_1, \alpha_2, \ldots, \alpha_n) \not\equiv 0$ — некоторая функция трехзначной логики, все переменные которой, как и сама функция f, принимают значения из множества $E_3 = \{0, \mu, 1\}$. Рассмотрим произвольный набор $\tilde{s} = (a_1, a_2, \ldots, a_n)$ значений переменных $\alpha_1, \ldots, \alpha_n$. Пусть $[f]_{\tilde{s}}$ — значение функции f на наборе $\tilde{s}$. Справедливо следующее равенство:

$$f(\alpha_1, \ldots, \alpha_n) = \bigvee_{i=0}^{3^n-1} [f]_{\tilde{s}_i} K(\tilde{s}_i). \qquad (4.37)$$

Действительно, на произвольном наборе $\tilde{s} = (a_1, \ldots, a_n)$ равенство (4.37) имеет вид

$$f(a_1, \ldots, a_n) = 0 \vee \ldots \vee 0 \vee ([f]_{\tilde{s}} 1) \vee 0 \vee \ldots \vee 0 = [f]_{\tilde{s}}.$$

Ввиду произвольности выбора $\tilde{s}$ равенство (4.37) доказано. Поскольку нулевые члены дизъюнкции можно опустить, равенство (4.37) может быть преобразовано:

$$f(\alpha_1, \ldots, \alpha_n) = D_1(f) \vee \mu D_2(f), \qquad (4.38)$$

где D_1 (D_2) — дизъюнкция всех конституэнт 1, соответствующих наборам, на которых функция f принимает значение единицы (соответственно μ). Правая часть равенства (4.38) называется **совершенной д. н. ф. (с. д. н. ф.)** функции f. Если $f(\alpha_1, \ldots, \alpha_n) = D_1(f)$, то функция f представлена **простой с. д. н. ф.** Из приведенного построения следует единственность представления функции f в совершенной д. н. ф.

Таким образом, справедлива следующая теорема.

Теорема 4.4. *Всякая функция $f \not\equiv 0$ трехзначной логики Φ_{E_3} однозначно представима совершенной д. н. ф.*

Из теоремы 4.4 следует, в частности, полнота системы операций $\{\vee, \wedge, \alpha^{a(1)}\}$ в трехзначной логике Φ_{E_3}. Теорема 4.4 указывает на способ, по которому может быть построена с. д. н. ф. произвольной функции $f \not\equiv 0$, заданной таблицей истинности. Для этого достаточно построить все конституэнты 1, соответствующие наборам, на которых функция f обращается в единицу, и все конституэнты 1, соответствующие наборам, на которых функция f принимает значение μ, после чего нетрудно получить с. д. н. ф. (4.38) данной функции.

Заметим, что произвольная д. н. ф. функции $f(\alpha_1, \ldots, \alpha_n) = V_1 \vee V_2 \vee \ldots \vee V_k$ также может быть преобразована в с. д. н. ф. данной функции. Для этого достаточно каждую элементарную конъюнкцию V_i заменить равным ей выражением

$$V_i(\alpha_{i_1}^{0(1)} \vee \alpha_{i_1}^{\mu(1)} \vee \alpha_{i_1}^{1(1)}) \ldots (\alpha_{i_r}^{0(1)} \vee \alpha_{i_r}^{\mu(1)} \vee \alpha_{i_r}^{1(1)}),$$

где $\alpha_{i_1}, \alpha_{i_2}, \ldots, \alpha_{i_r}$ — переменные из набора $\tilde{X} = \{\alpha_1, \ldots, \alpha_n\}$, не входящие в V_i. Раскрыв затем скобки, получим с. д. н. ф. функции f.

Рассмотрим нормальные формы представлений в алгоритмической алгебре условий $\mathfrak{B}_B$.

Пусть $F(\alpha_1, \ldots, \alpha_n; A_1, \ldots, A_m)$ — произвольная формула (условие) алгебры $\mathfrak{B}_B$, составленная из элементарных условий $\tilde{X} = \{\alpha_1, \ldots, \alpha_n\}$ и переменных операторов $\tilde{G} = \{A_1, \ldots, A_m\}$. **Расширенной элементарной конъюнкцией** или ***r*-конъюнкцией** называется выражение

$$R_i = V_i \Delta_i (A_{i_1} V_{i_1}) \ldots (A_{i_k} V_{i_k}),$$

где V_i — произвольная элементарная конъюнкция алгебры Φ_{E_3}, составленная из условий, принадлежащих набору $\tilde{X}$, и константы $k_i = \{1, \mu\}$; $V_{i_1}, \ldots, V_{i_k}$ — элементарные конъюнкции алгебры Φ_{E_3}, составленные из условий, принадлежащих набору $\tilde{X}$; Δ_i — булева элементарная конъюнкция, составленная из характеристических условий, принадлежащих набору $\tilde{\Delta} = \{{}_{A_1}\Delta, {}_{A_2}\Delta, \ldots, {}_{A_m}\Delta\}$ (причем характеристические функции ${}_{A_j}\Delta$, $j = 1, 2, \ldots, k$, либо отсутствуют в конъюнкции Δ_i, либо входят в Δ_i без знака отрицания); $A_{i_1}, \ldots, A_{i_k}$ — попарно-различные операторы, принадлежащие набору $\tilde{G}$.

Под **расширенной д. н. ф.**, или **r-д. н. ф.**, W в алгебре $\mathfrak{B}_B$ понимается дизъюнкция конечного числа попарно-различных и не поглощающих друг друга r-конъюнкций:

$$W = (\bigvee_{i=1}^{k_1} R_i) \vee \mu (\bigvee_{j=1}^{k_2} R_j),$$

где R_i, R_j — r-конъюнкции с коэффициентом $k = 1$.

Лемма 4.1. *Произвольная формула* $F(a_1, \ldots, a_n; A_1, \ldots, A_m)$ *алгебры* $\mathfrak{B}_B$ *с помощью тождеств* (4.24)—(4.31) *может быть преобразована в эквивалентную* r-д. н. ф. W_F.

Алгоритм приведения произвольной формулы F алгебры $\mathfrak{B}_B$ к r-д. н. ф. W_F состоит в следующем.

1. Используя правила де Моргана, а также тождества (4.25), (4.27), (4.28), (4.34), опускаем символы унарных операций алгебры $\mathfrak{B}_B$ на элементарные условия $a_1, \ldots, a_n$.

2. На основании тождеств (4.26), а также тождества (4.25) исключаем вхождения операций отрицания и $a^{a(0)}$ над элементарными условиями $a_1, \ldots, a_n$.

3. С помощью первого дистрибутивного закона и тождества (4.31) раскрываем скобки.

4. Используя коммутативность конъюнкции в каждом дизъюнктивном члене, группируем условия вида $(Aa_1)(Aa_2)\ldots(Aa_r)$ с последующим выносом оператора A за знак конъюнкции (см. тождество (4.30)).

5. Группируем характеристические условия, входящие в каждый дизъюнктивный член, с помощью тождеств алгебры Буля, а также тождеств (4.28), (4.29), (4.32), (4.33) и преобразуем выделенные группы в элементарные конъюнкции Δ_i.

6. На основании правил идемпотентности и тождеств для констант упрощаем полученное выражение.

7. Используя тождества (4.32), исключаем вхождения условий вида $A0$ и раскрываем скобки, если они появляются.

8. Учитывая коммутативность условий, идемпотентность конъюнкции и справедливость тождеств для констант, преобразуем каждый дизъюнктивный член полученного выражения в r-конъюнкцию.

9. Применяя правила поглощения, а также учитывая идемпотентность дизъюнкции и справедливость тождеств для констант, преобразуем полученное выражение в r-д. н. ф. W_F.

Лемма доказана.

Пример 2 Преобразуем в r-д. н. ф. выражение

$$F(a_1, a_2, A_1, A_2) = (a_1^{\mu(0)} \vee a_2^{1(0)})^{0(1} \cdot {}_{A_1}\Delta \overline{(A_1 a_1^{\mu(1)} \vee A_1 a_2^{1(0)})} \cdot A_2 a_1^{0(1)}.$$

Для этого на основании (4.27) раскрываем скобки и, используя (4.25), упрощаем

$$F(a_1, a_2; A_1, A_2) = a_1^{\mu(1)} \cdot a_2^{1(1)} \cdot {}_{A_1}\Delta \overline{(A_1 a_1^{\mu(1)} \vee A_1 a_2^{1(0)})} \cdot A_2 a_1^{0(1)}.$$

На основании правила де Моргана, а также тождества (4.28) получаем

$$a_1^{\mu(1)} a_2^{1(1)} {}_{A_1}\Delta\, (A_1 a_1^{\mu(0)})\, (A_1 a_2^{1(1)})\, A_2 a_1^{0(1)}.$$

Затем, используя (4.26), а также учитывая (4.31) и раскрывая скобки, получаем

$$a_1^{\mu(1)} \cdot a_2^{1(1)} \cdot {}_{A_1}\Delta\, (A_1 a_1^{1(1)})\, (A_1 a_2^{1(1)}) \cdot A_2 a_1^{0(1)} \vee a_1^{\mu(1)} \cdot \cdot a_2^{1(1)} \cdot {}_{A_1}\Delta \cdot (A_1 \cdot a_1^{0(1)}) \cdot (A_1 a_2^{1(1)}) \cdot A_2 a_1^{0(1)}.$$

Наконец, в силу (4.30) выносим оператор A_1 так, что

$$a_1^{\mu(1)} \cdot a_2^{1(1)} \cdot {}_{A_1}\Delta A_1 a_1^{1(1)} \cdot a_2^{1(1)} \cdot A_2 a_1^{0(1)} \vee a_1^{\mu(1)} \cdot \cdot a_2^{1(1)} \cdot {}_{A_1}\Delta \cdot A_1 a_1^{0(1)} \cdot a_2^{1(1)} \cdot A_2 a_1^{0(1)}.$$

Полученное выражение и является r-д. н. ф. искомой формулы.

Пусть, далее, $\tilde{s} = (a_1, a_2, \ldots, a_n)$, $\tilde{q} = (q_1, q_2, \ldots, q_m)$ — наборы значений соответственно элементарных $\tilde{X} = \{a_1, \ldots, a_n\}$ и характеристических $\tilde{\Delta} = \{{}_{A_1}\Delta, {}_{A_2}\Delta, \ldots, {}_{A_m}\Delta\}$ условий.

Конъюнкция

$$K(\tilde{s}, \tilde{q}) = a_1^{a_1(1)} a_2^{a_2(1)} \ldots a_n^{a_n(1)} \wedge {}_{A_1}\Delta^{q_1} {}_{A_2}\Delta^{q_2} \ldots {}_{A_m}\Delta^{q_m},$$

где

$${}_{A_i}\Delta^{q_i} = \begin{cases} {}_{A_i}\Delta, & \text{если } q_i = 1, \\ {}_{A_i}\overline{\Delta}, & \text{если } q_i = 0 \end{cases} \quad (i = 1, 2, \ldots, m),$$

называется **обобщенной конституэнтой 1**, соответствующей паре наборов $(\tilde{s}, \tilde{q})$.

Любая обобщенная конституэнта 1, $K(\tilde{s}, \tilde{q})$, принимает значение единицы лишь на соответствующей паре наборов $(\tilde{s}, \tilde{q})$, обращаясь в нуль при любых других значениях условий, входящих в $\tilde{x}$, $\tilde{\Delta}$. Число всех обобщенных конституэнт 1 составляет $(3^n \cdot 2^m)$.

Пусть $i_1, i_2, \ldots, i_k$ $(k \leqslant m)$ — номера всех компонент набора $\tilde{q} = (q_1, q_2, \ldots, q_m)$ таких, что $q_{i_1} = q_{i_2} = \ldots = q_{i_k} = 1$. Под **полной r-конъюнкцией**, соответствующей наборам $\langle \tilde{s}, q, \tilde{s}_{i_1}, \ldots \ldots, \tilde{s}_{i_k} \rangle$, понимается выражение

$$\widehat{R}(\tilde{s}, \tilde{q}, \tilde{s}_{i_1}, \ldots, \tilde{s}_{i_k}) = kK(\tilde{s}, \tilde{q})\,[A_{i_1} K(\tilde{s}_{i_1})] \wedge \wedge [A_{i_2} K(\tilde{s}_{i_2})] \wedge \ldots \wedge [A_{i_k} K(\tilde{s}_{i_k})],$$

где $K(\tilde{s}_{i_1})$, $K(\tilde{s}_{i_2})$, ..., $K(\tilde{s}_{i_k})$ — конституэнты 1, соответствующие наборам $\tilde{s}_{i_1}, \tilde{s}_{i_2}, \ldots, \tilde{s}_{i_k}$ значений условий, входящих в $\tilde{X}$ ($k = \{1, \mu\}$). Заметим, что полная r-конъюнкция $\widehat{R}\langle \tilde{s}, \tilde{q}, \tilde{s}_{i_1}, \ldots, \tilde{s}_{i_k}\rangle$ принимает значение k лишь на соответствующей системе наборов $\langle \tilde{s}, \tilde{q}, \tilde{s}_{i_1}, \ldots, \tilde{s}_{i_k}\rangle$, обращаясь в нуль на любой другой системе.

Полной r-д. н. ф. $\widehat{W}$ называется дизъюнкция конечного числа попарно-различных и не поглощающих друг друга полных r-конъюнкций:

$$\widehat{W} = \bigvee_{i=1}^{k_1} \widehat{R}_i \vee \mu \left(\bigvee_{j=1}^{k_2} \widehat{R}_j\right).$$

Существует алгоритм приведения произвольной r-д. н. ф.

$$W = \bigvee_{s=1}^{k_1} R_s \vee \mu \left(\bigvee_{t=1}^{k_2} R_t\right) \tag{4.39}$$

к полной r-д.н.ф. $\widehat{W}$. Для этого каждую r-конъюнкцию

$$R_q = V_q \Delta_q [A_{q_1} V_{q_1}] \ldots [A_{q_k} V_{q_k}],$$

входящую в (4.39), необходимо преобразовать следующим образом.

1. Используя правила алгебры Буля, преобразуем булеву элементарную конъюнкцию Δ_s в с. д. н. ф., состоящую из конституэнт 1, соответствующих наборам значений характеристических условий, входящих в $\tilde{\Delta}$.

2. Раскрываем скобки по первому дистрибутивному закону и применяем правила идемпотентности и поглощения.

3. На основании тождества (4.33) преобразуем каждую из полученных r-конъюнкций в r-конъюнкцию вида

$$R_l = V_l {}_{A_1}\Delta^{q_1}{}_{A_2}\Delta^{q_2} \ldots {}_{A_m}\Delta^{q_m} [A_{l_1} V_{l_1}] \wedge \cdots \wedge [A_{l_r} V_{l_r}], \tag{4.40}$$

где $l_1, l_2, \ldots, l_r$ — номера всех компонент набора $\tilde{q} = (q_1, \ldots, q_m)$ таких, что $q_{l_1} = q_{l_2} = \cdots = q_{l_r} = 1$.

4. Каждую элементарную конъюнкцию V_l, $V_{l_1}, V_{l_2}, \ldots, V_{l_r}$, входящую в R_l (4.40), преобразуем в простые с. д. н. ф., используя преобразования алгебры Φ_{E_2}.

Раскрывая скобки согласно первому дистрибутивному закону и тождеству (4.31), выполняя допустимые упрощения после перегруппировки полных r-конъюнкций и вынося за скобки коэффициент μ, получаем полную r-д. н. ф. $\widehat{W}$. Таким образом, в силу леммы 4.1 и приведенного алгоритма каждая формула (условие)

$F(\alpha_1, \ldots, \alpha_n; A_1, \ldots, A_m)$ алгебры $\mathfrak{B}_B$ может быть преобразована в эквивалентную полную r-д. н. ф.

$$\widehat{W}_F = \bigvee_{i=1}^{k_1} \widehat{R}_i \vee \mu \left(\bigvee_{j=1}^{k_2} \widehat{R}_j\right), \tag{4.41}$$

называемую **полной r-д. н. ф. условия F.**

Выражение $D_F = \bigvee_{i=1}^{k_1} \widehat{R}_i$ называется **истинной частью** полной r-д. н. ф. $\widehat{W}_F$ (4.41). Покажем единственность такого представления. Пусть $\widehat{W}_F$ и $\widehat{W}'_F$ — различные полные r-д. н. ф. условия F. Тогда найдется по крайней мере одна полная r-конъюнкция

$$\widehat{R}\langle \tilde{s}, \tilde{q}, \tilde{s}_{i_1}, \ldots, \tilde{s}_{i_k}\rangle = kK(\tilde{s}, \tilde{q})[A_{i_1} K(\tilde{s}_{i_1})] \wedge \wedge [A_{i_2} K(\tilde{s}_{i_2})] \wedge \cdots \wedge [A_{i_k} K(\tilde{s}_{i_k})],$$

входящая в одну из указанных полных r-д. н. ф. (например, в $\widehat{W}_F$) и не входящая в другую. Присваивая соответствующим условиям, входящим в $\widehat{W}_F$ и $\widehat{W}'_F$, указанные для них значения в наборах $s, \tilde{q}, \tilde{s}_{i_1}, \ldots, \tilde{s}_{i_k}$, замечаем, что $\widehat{W}_F$ принимает значение $k\{1, \mu\}$, а $\widehat{W}'_F$ в данном случае обращается в нуль. Но это противоречит первоначальному предположению о том, что $\widehat{W}_F$ и $\widehat{W}'_F$ являются полными r-д. н. ф. одного и того же условия F.

Таким образом, справедливо следующее утверждение.

Теорема 4.5. *Произвольная формула $F(\alpha_1, \ldots, \alpha_n; A_1, \ldots, A_m)$ алгебры $\mathfrak{B}_B$ однозначно представима в полной r-д. н. ф.*

Пример 3. Преобразуем r-д. н. ф., полученную в примере 2, в полную r-д. н. ф.:

$$F(\alpha_1, \alpha_2; A_1, A_2) = \alpha_1^{\mu(1)} \alpha_2^{1(1)} {}_{A_1}\Delta A_1 \alpha_1^{1(1)} \alpha_2^{1(1)} \cdot A_2 \alpha_1^{0(1)} \vee \vee \alpha_1^{\mu(1)} \cdot \alpha_2^{1(1)} {}_{A_1}\Delta \cdot A_1 \alpha_1^{0(1)} \cdot \alpha_2^{1(1)} \cdot A_2 \alpha_1^{0(1)}.$$

Для образования обобщенных конституэнт 1, используя тождество ${}_A\Delta \vee {}_A\bar{\Delta} = 1$ и раскрывая скобки, вводим в каждую r-конъюнкцию характеристическое условие ${}_{A_2}\Delta^{q_2}$:

$$\alpha_1^{\mu(1)} \cdot \alpha_2^{1(1)} {}_{A_1}\Delta (A_2 \Delta \vee {}_{A_2}\bar{\Delta}) A_1 \alpha_1^{1(1)} \cdot \alpha_2^{1(1)} A_2 \alpha_1^{0(1)} \vee \alpha_1^{\mu(1)} \alpha_2^{1(1)} \cdot$$
$$\cdot {}_{A_1}\Delta (A_2 \Delta \vee {}_{A_2}\bar{\Delta}) A_1 \alpha_1^{0(1)} \cdot \alpha_2^{1(1)} A_2 \alpha_1^{0(1)} = \alpha_1^{\mu(1)} \alpha_2^{1(1)} {}_{A_1}\Delta_{A_2}\Delta \cdot$$
$$\cdot A_1 \alpha_1^{1(1)} \cdot \alpha_2^{1(1)} \cdot A_2 \alpha_1^{0(1)} \vee \alpha_1^{\mu(1)} \cdot \alpha_2^{1(1)} {}_{A_1}\Delta_{A_2}\bar{\Delta} A_1 \alpha_1^{1(1)} \cdot \alpha_2^{1(1)} \cdot$$
$$\cdot A_2 \alpha_1^{0(1)} \vee \alpha_1^{\mu(1)} \alpha_2^{1(1)} \cdot {}_{A_1}\Delta_{A_2}\Delta A_1 \alpha_1^{0(1)} \cdot \alpha_2^{1(1)} \cdot A_2 \alpha_1^{0(1)} \vee$$
$$\vee \alpha_1^{\mu(1)} \cdot \alpha_2^{1(1)} {}_{A_1}\Delta_{A_2}\bar{\Delta} \cdot A_1 \alpha_1^{0(1)} \alpha_2^{1(1)} \cdot A_2 \alpha_1^{0(1)}.$$

Для полученных обобщенных конституэнт вводим обозначения $\hat{k}_1 = \alpha_1^{\mu(1)}\alpha_2^{1(1)}{}_{A_1}\Delta_{A_2}\Delta$, $\hat{k}_2 = \alpha_1^{\mu(1)}\alpha_2^{1(1)}{}_{A_1}\Delta_{A_2}\bar{\Delta}$. Кроме того, пусть $k_1 = \alpha_1^{1(1)}\alpha_2^{1(1)}$, $k_2 = \alpha_1^{0(1)}\alpha_2^{1(1)}$. Тогда на основании (4.32), исключая во втором и четвертом дизъюнктивных членах вхождения условия $A_2\alpha_1^{0(1)}$ и вынося константы μ за скобки, получаем

$$\hat{k}_1A_1k_1A_2\alpha_1^{0(1)} \vee \hat{k}_1A_2k_2A_2\alpha_1^{0(1)} \vee \mu(\hat{k}_2A_1\hat{k}_1 \vee \hat{k}_2A_1k_2).$$

Применяя обобщенный закон исключенного третьего ($\alpha^{1(1)} \vee \alpha^{0(1)} \vee \alpha^{\mu(1)} = 1$), вводим в условия $A_2\alpha_1^{0(1)}$ фиктивную переменную α_2:

$$\hat{k}_1A_1k_1 \cdot A_2\alpha_1^{0(1)}(\alpha_2^{1(1)} \vee \alpha_2^{0(1)} \vee \alpha_2^{\mu(1)}) \vee \hat{k}_1A_1k_2\alpha_1^{0(1)}(\alpha_2^{1(1)} \vee \alpha_2^{0(1)} \vee \alpha_2^{\mu(1)}) \vee \mu(\hat{k}_2A_1k_1 \vee \hat{k}_2A_1k_2).$$

Наконец, раскрывая скобки (учитывая (4.31)), получаем

$$\hat{k}_1A_1k_1A_2k_2 \vee \hat{k}_1A_1k_1A_2k_3 \vee \hat{k}_1A_1k_1A_2k_4 \vee \hat{k}_1A_1k_2A_2k_2 \vee \hat{k}_1A_1k_2A_2k_3 \vee \hat{k}_1A_1k_2A_2k_4 \vee \mu(\hat{k}_2A_1k_1 \vee \hat{k}_2A_1k_2),$$

где $\alpha_1^{0(1)}\alpha_2^{0(1)} = k_3$, $\alpha_1^{0(1)}\alpha_2^{\mu(1)} = k_4$. Полученное выражение является полной r-д. н. ф. условия $F(\alpha_1, \alpha_2; A_1, A_2)$.

Таблица 6

$\alpha \to \beta$	0	μ	1
0	1	1	1
μ	0	1	1
1	0	0	1

Остановимся кратко на одной из важных проблем, связанных с формальными преобразованиями в алгебре $\mathfrak{B}_B$, — проблеме минимизации в классе r-д. н. ф., состоящем в нахождении минимальной (по числу букв) r-д. н. ф. некоторого условия. Предварительно введем операцию импликации $\alpha \to \beta$ условий α, β, заданную таблицей истинности (табл. 6). Импликация может быть представлена в алгебре Φ_{E_3} в виде

$$\alpha \to \beta = \alpha^{0(1)} \vee \alpha^{\mu(1)} \cdot \beta^{\mu(1)} \vee \beta^{1(1)}.$$

Введем понятие импликанты и простой импликанты в алгебре условий Φ_{E_3}. Пусть $F(\alpha_1, \ldots, \alpha_n; A_1, \ldots, A_m)$ и $G(\alpha_1, \ldots, \alpha_n; A_1, \ldots, A_m) \in \mathfrak{B}_B$ — выражения, составленные из элементарных условий $\alpha_1, \ldots, \alpha_n$ и переменных операторов $A_1, \ldots, A_m$. Условие $F(\alpha_1, \ldots, \alpha_n; A_1, \ldots, A_m)$ называется **импликантой** условия $G(\alpha_1, \ldots, \alpha_n; A_1, \ldots, A_m)$, если выполняется тождество $F(\alpha_1, \ldots, \alpha_n; A_1, \ldots, A_m) \to G(\alpha_1, \ldots, \alpha_n; A_1, \ldots, A_m) \equiv 1$. Иными словами, при любой интерпретации I (присвоении конкретных значений условиям $\alpha_1, \ldots, \alpha_n$ и операторам $A_1, \ldots, A_m$) $[F]_I \leqslant [G]_I$, где $[F]_I$, $[G]_I$ — значения условий соответственно F и G при интерпретации I. Если r-конъюнкция R — импликанта условия F, причем никакая собственная часть конъюнкции R (полученная в результате отбрасывания хотя бы одного условия, входящего в R) уже не является импликантой условия F, то R называют **простой импликантой** условия F. Дизъюнкция всех простых импликант условия F называется **сокращенной** r-д. н. ф. данного условия.

Теорема 4.6. *Каждое условие F алгебры $\mathfrak{B}_B$ однозначно представимо сокращенной r-д. н. ф. данного условия.*

Доказательство этой теоремы строится аналогично доказательству результата о сокращенных д. н. ф. булевых функций [24]. Как и при минимизации д. н. ф. булевых функций, процесс нахождения минимальной r-д. н. ф. некоторого условия состоит из двух этапов. На первом этапе строится сокращенная r-д. н. ф. данного условия. Для этого можно применить, например, аналог алгоритма Квайна, используя операции неполного склеивания:

$$\alpha\beta^{0(1)} \vee \alpha\beta^{\mu(1)} \vee \alpha\beta^{1(1)} = \alpha \vee \alpha\beta^{0(1)} \vee \alpha\beta^{\mu(1)} \vee \alpha\beta^{1(1)},$$

$$\alpha\,{}_A\Delta \vee \alpha\,{}_A\bar{\Delta} = \alpha \vee \alpha\,{}_A\Delta \vee \alpha\,{}_A\bar{\Delta},$$

а также операцию **поглощения** $\alpha\beta \vee \alpha = \alpha$. На втором этапе в результате исключения «лишних» простых импликант из сокращенной r-д. н. ф. условия F строятся все тупиковые r-д. н. ф. данного условия. Теория r-к. н. ф. в алгебре $\mathfrak{B}_B$, как и в алгебре Буля, строится двойственным способом.

§ 4.3. Полугруппы периодически определенных преобразований

Одним из важных вопросов, касающихся применения аппарата алгоритмических алгебр к задачам автоматизации проектирования ЭВМ и программирования, является выбор структуры состояний операционного автомата B, которые образуют информационное множество M данного автомата. На множестве M определяются элементарные операторы и логические условия, составляющие систему образующих в алгоритмических алгебрах $\langle \mathfrak{A}_B, \mathfrak{B}_B \rangle$, связанных с операционным автоматом B (см. § 4.1). При этом не всякий оператор (логическое условие) на множестве M может быть выбран в качестве элементарного.

Определение структуры состояний информационного множества M и совокупности допустимых элементарных операторов и логических условий связано с понятием абстрактного регистра. **Абстрактный регистр** — это устройство для хранения слова в некотором конечном алфавите $\mathfrak{A}$. Каждая буква запоминаемого слова хранится в некотором (абстрактном) элементе, называемом **разрядом** данного регистра. Разряды нумеруются последовательно целыми числами. Числа, соответствующие разрядам регистра, образуют отрезок целых чисел $[m, n]$, называемый **нумерующим отрезком** данного регистра ($m \leqslant n$). Если обе границы этого отрезка конечны,

то конечен и сам регистр (число его элементов в этом случае равно $n - m + 1$). При этом m-й разряд регистра называется его **младшим**, а n-й — **старшим разрядами**. Не исключен также случай, когда одно или оба из пары чисел (m, n) бесконечны ($m = -\infty$, $n = +\infty$). При этом слова, запоминаемые на регистре, также бесконечны. Ограничимся, однако, лишь такими бесконечными словами, у которых почти все буквы (все, кроме конечного числа букв) одинаковы.

Абстрактный регистр полностью определяется заданием своего алфавита $\mathfrak{A}$ и нумерующего отрезка $[m, n]$. Для превращения абстрактного регистра в **физический** необходимо дополнительно задать структуру его разрядов и кодирование букв алфавита $\mathfrak{A}$. Структура разряда (предполагаем ее одинаковой для всех разрядов данного регистра) определяется числом состояний элементарных запоминающих ячеек, из которых строится этот разряд. Обычно в современных ЦВМ используются двоичные элементы. Поэтому зададим структуру разряда абстрактного регистра числом p составляющих его двоичных элементов. Соответствующий разряд при этом называется **p-битовым разрядом**. Кодирование букв алфавита $\mathfrak{A}$ при p-битовых разрядах задается взаимно-однозначным отображением этого алфавита в множество кортежей длины p, составленных из нулей и единиц. Для возможности такого кодирования число p должно выбираться так, чтобы 2^p было не меньше числа букв в алфавите $\mathfrak{A}$.

Многорегистровый операционный автомат B определяется заданием некоторого конечного множества абстрактных регистров с определенными на них элементарными операторами и логическими условиями. При этом у каждого регистра в общем случае может быть свой алфавит $\mathfrak{A}$. Запоминанием на регистрах слов в соответствующих алфавитах определяется состояние операционного автомата B. Каждый элементарный оператор задает некоторое отображение множества M состояний данного автомата в себя, а всякое элементарное логическое условие определяет отображение множества M в двухэлементное множество {и, л}. Однако не всякое отображение M в себя задает элементарный оператор. Необходимым (и достаточным) условием для этого является условие простоты реализации отображения M в себя. То же относится и к элементарным логическим условиям.

Если состояние автомата B представить в виде упорядоченной последовательности слов $q_1, q_2, \ldots, q_k$, хранящихся соответственно на регистрах $1, 2, \ldots, k$, то любое отображение f множества M в себя может быть задано в виде совокупности функций $f_{ij}(\ldots a_{i_1}, \ldots, a_{i_2}, \ldots, a_{i_k}, \ldots)$. Функция f_{ij} определяет значение j-го разряда i-го регистра данного автомата. Аргументами этой функции в общем случае служат все буквы комбинированного слова $l = q_1 q_2 \ldots q_k$. Простыми естественно считать те из функций f_{ij}, которые фактически зависят лишь от относительно небольшого числа своих аргументов.

Заметим, что существует принципиальная возможность построения достаточно простых схем для реализации семейства функций и в том случае, когда каждая следующая функция фактически зависит от все большего и большего числа аргументов. Классическим примером такого рода является обыкновенный параллельный двоичный сумматор. Для каждого разряда сумматора определяющая значение этого разряда булева функция зависит от трех переменных (значений соответствующих разрядов слагаемых и переноса из предыдущего разряда). Перенос же хотя в каждом очередном разряде и определяется простой схемой, зависит фактически от всех предыдущих разрядов слагаемых. Параллельный двоичный сумматор представляет собой лишь частный пример так называемых каскадных схем, в котором схема каждого разряда относительно проста. В совокупности, однако, последовательность каскадов может быть достаточно сложной функцией.

Современная технология дает возможность резко снизить стоимость изготовления сложных схем, если они составлены из однородных элементарных блоков. В связи с этим особый интерес представляют каскадные схемы, у которых все каскады (за исключением, может быть, нескольких начальных и конечных) одинаковы. С однородными структурами связано понятие периодически определенного преобразования, предложенного В. М. Глушковым [18].

Пусть $\widetilde{X} = \{\ldots, x_{+1}, x_0, x_{-1}, \ldots\}$ — бесконечный абстрактный регистр с нумерующим отрезком $(+\infty, -\infty)$; α_i — содержимое элемента x_i до и после преобразования. Преобразование на регистре называется **периодически определенным**, если существует такая функция $f(\tau_{i_1}, \tau_{i_2}, \ldots, \tau_{i_n})$, что при всех $i = \cdots, -2, -1, 0, 1, 2, \ldots$

$$\beta_i = f(\alpha_{i_1+i}, \alpha_{i_2+i}, \ldots, \alpha_{i_n+i}). \tag{4.42}$$

Функция f называется **порождающей функцией** данного преобразования.

Понятие периодически определенного преобразования естественно переносится на случай конечного регистра. При этом следует найти значения тех переменных x_{i_k+i}, для которых значения индекса $i_k + i$ выходят за пределы рассматриваемого регистра. Наиболее просто эта задача решается добавлением к регистру (фиктивных) разрядов с постоянным запоминанием. При преобразованиях содержимое этих разрядов не изменяется, но может быть использовано в порождающих функциях.

Периодически определенное преобразование может быть задано с помощью некоторого целого числа k — коэффициента данного

преобразования и порождающей функции $f(\tau_1, \tau_2, \ldots, \tau_r)$, которая существенно зависит от своих крайних аргументов τ_1, τ_r, тогда как внутренние переменные τ_i $(1 < i < r)$ могут быть как существенными, так и фиктивными [8]. Заданное таким образом преобразование $F_{k;f^r}$ переводит регистр X из состояния $\tilde{\alpha} = (\ldots, \alpha_1, \alpha_0, \alpha_{-1}, \ldots)$ в состояние $\tilde{\beta} = (\ldots, \beta_1, \beta_0, \beta_{-1}, \ldots)$, где $\beta_i = f(\alpha_{i+k}, \alpha_{i+k+1}, \ldots, \alpha_{i+k+r-1})$, $-\infty < i < +\infty$.

Периодически определенные преобразования на регистре X называют **однорегистровыми операторами**. Примером однорегистрового оператора является **сдвиг** $C_{k_1; l_0^0}$ на регистре X, где коэффициент k_1 указывает направление и число, на которое сдвигаются элементы регистра (при сдвиге вправо $k_1 > 0$, при сдвиге влево $k_1 < 0$), а $l_0^0(\tau) = \tau$. В двузначном случае, т. е. при булевом регистре X, если $f(\tau) = \bar{\tau}$, а коэффициент $k_1 = 0$, однорегистровый оператор $I_{0;f}$ реализует **инверсию** на регистре X. Однорегистровыми операторами являются оператор 0 — очистка регистра X и оператор T — установка регистра X в единичное состояние $\tilde{\alpha}_1 = (\ldots, 1, 1, \ldots)$. Первый задается порождающей функцией $f \equiv 0$, а второй — $f \equiv 1$.

Пусть X — двоичный регистр и $F_{k;f^r}$ — однорегистровый оператор на X, заданный коэффициентом k и порождающей булевой функцией $f(\tau_1, \tau_2, \ldots, \tau_r)$.

Теорема 4.7. *Каждому однорегистровому оператору $F_{k;f^r}$ однозначно соответствуют коэффициент k и функция $f(\tau_1, \tau_2, \ldots, \tau_r)$, задающие данный оператор.*

Действительно, пусть заданы операторы $F_{k_1;f^r}$ и $G_{k_2;g^s}$. Если

$$F_{k_1;f^r} \equiv G_{k_2;g^s}, \tag{4.43}$$

то $k_1 = k_2$ и $f(\tau_1, \tau_2, \ldots, \tau_r) = g(\tau_1, \tau_2, \ldots, \tau_s)$. Допустим, что $k_1 \neq k_2$. Тогда найдется состояние $\tilde{\alpha} \in M$ регистра X, для которого

$$F_{k_1;f^r}(\tilde{\alpha}) \neq G_{k_2;g^s}(\tilde{\alpha}). \tag{4.44}$$

Для определенности предположим, что $k_1 < k_2$. Функция $f(\tau_1, \tau_2, \ldots, \tau_r)$ существенно зависит от переменной τ_1. Это означает, что найдется набор $q_2, q_3, \ldots, q_r$ значений переменных $\tau_2, \tau_3, \ldots, \tau_r$ такой, что $f(0, q_2, q_3, \ldots, q_r) \neq f(1, q_2, q_3, \ldots, q_r)$. Тогда $\tilde{\alpha} \in M$ регистра X выберем так, что

$$\alpha_{i+k_1+1} = q_2, \quad \alpha_{+k_1+2} = q_3, \quad \ldots, \quad \alpha_{i+k_1+r-1} = q_r.$$

Но ввиду того что $i + k_1 < i + k_2$, состояние α_{i+k_1} элемента x_{i+k} можно выбрать так, что

$$f(\alpha_{i+k_1}, q_2, q_3, \ldots, q_r) \neq g(\alpha_{i+k_2}, \alpha_{i+k_2+1}, \ldots, \alpha_{i+k_2+s-1}).$$

Отсюда следует справедливость неравенства (4.44), что противоречит (4.43). Таким образом, $k_1 = k_2$. Аналогично доказывается равенство функций $f(\tau_1, \ldots, \tau_r) = g(\tau_1, \tau_2, \ldots, \tau_s)$.

Композиция, или произведение, т. е. последовательное применение двух однорегистровых операторов, также является однорегистровым оператором.

Пусть $F_{k_1;f^r}$ и $G_{k_2;g^s}$ — однорегистровые операторы. $F_{k_1;f^r} \times G_{k_2;g^s}$ — их композиция $H_{k;h^t}$. Коэффициент k и функцию h следует определить. Для этого рассмотрим суперпозицию функций $f(\tau_1, \tau_2, \ldots, \tau_r)$ и $g(\tau_1, \tau_2, \ldots, \tau_s)$, заданную равенством

$$(f \times g)(\tau_1, \tau_2, \ldots, \tau_{r+s-1}) = f(g(\tau_1, \tau_2, \ldots, \tau_s), g(\tau_2, \tau_3, \ldots, \tau_{s+1}), \ldots, g(\tau_r, \tau_{r+1}, \ldots, \tau_{r+s-1})) = \tilde{h}(\tau_1, \tau_2, \ldots, \tau_{r+s-1}). \tag{4.45}$$

Пусть $m_{\tilde{h}} \geqslant 0$ и $n_{\tilde{h}} \leqslant r + s - 1$ — индексы соответственно первой слева и последней переменной, от которых существенно зависит функция $\tilde{h}$. Тогда $k = k_1 + k_2 + \Delta_H$, где $\Delta_H = m_{\tilde{h}}$, $t = n_{\tilde{h}} - m_{\tilde{h}}$, а функция $h(\tau_1, \tau_2, \ldots, \tau_{n_{\tilde{h}} - m_{\tilde{h}}})$ получается из функции $\tilde{h}(\tau_1, \tau_2, \ldots, \tau_{r+s-1})$ в результате отбрасывания всех фиктивных переменных τ_j, для которых $j < m_{\tilde{h}}$ или $j > n_{\tilde{h}}$. Число Δ_H называется **дефектом композиции** $H_{k;h^t}$, а функция h — **композицией функций** f, g. Естественно, операция композиции ассоциативна. Таким образом, множество всех однорегистровых операторов образует полугруппу (см. § 3.6).

Полугруппой является также алгебра Ф булевых функций, порождающих однорегистровые операторы на двоичном регистре X. Основное множество этой полугруппы состоит из функций, существенно зависящих от первого и последнего аргументов с операцией композиции функций, определенной выше.

В полугруппе П зададим бинарное отношение ρ следующим образом: операторы $F_{k_1;f^r}$ и $G_{k_2;g^s}$ находятся в отношении ρ: $F_{k_1;f^r} \rho G_{k_2;g^s}$ тогда и только тогда, когда найдется сдвиг $C_{k;l_0^0}$ такой, что $F_{k_1;f^r} = G_{k_2;g^s} \times C_{k;l_0^0}$. Очевидно, что отношение ρ является отношением эквивалентности. Более того, ρ — конгруэнция (см. § 3.5). Таким образом, справедлива следующая теорема.

Теорема 4.8. *Фактор-полугруппа* П$/\rho$ *полугруппы* П *по конгруэнции* ρ *изоморфна полугруппе* Ф *булевых функций.*

На основании теоремы 4.8 изучение операторных алгебр, порожденных полугруппой П, в значительной степени можно свести к рассмотрению модифицированных алгебр логики, сигнатуры операций которых содержат операцию $(f \times g)$, определяемую соотношением (4.45).

Прежде всего возникает вопрос, существует ли конечная система образующих в полугруппе Φ. Рассмотрим некоторые важные подполугруппы полугруппы Φ [8]. Состояние $\alpha = (\ldots \alpha_{+1}, \alpha_0, \alpha_{-1}, \ldots)$ регистра X называется **периодическим**, если $\alpha_{i+n} = \alpha_i$ при любом i $(-\infty < i < +\infty)$, где n — натуральное число, называемое **периодом состояния** α. Множество всех периодических состояний с периодом n обозначим $\mathfrak{M}_n$. Очевидно, для произвольного однорегистрового оператора $F_{k;\, f^r} \in \Pi$, $F_{k;\, f^r}(\alpha) \in \mathfrak{M}_n$ при любом $\alpha \in \mathfrak{M}_n$.

Рассмотрим множество Φ_n функций таких, что $f(\tau_0, \tau_1, \ldots \ldots, \tau_r) \in \Phi_n$ тогда и только тогда, когда оператор $F_{k;\, f^r} \in \Pi$ осуществляет взаимно-однозначное отображение множества $\mathfrak{M}_n$ на себя при любом коэффициенте k (достаточно, чтобы это отображение осуществлялось при каком-нибудь одном k). Множества Φ_n $(n = 1, 2, \ldots)$ являются подполугруппами полугруппы Φ (см. § 3.5).

Лемма 4.2. *Полугруппы Φ_n $(n = 1, 2, \ldots)$ являются изолированными подполугруппами полугруппы Φ.*

Покажем, что если по крайней мере одна из функций f, $g \in \Phi$ не принадлежит подполугруппе Φ_n, т. е. осуществляет не взаимно-однозначное отображение $\mathfrak{M}_n$ в себя, то и композиция $h = f \times g$ не принадлежит Φ_n. При этом возможны два случая.

1. $f \in \Phi_n$. Тогда $g \overline{\in} \Phi_n$. Это означает, что для оператора $G_{k_2;\, g^s} \in \Pi$ найдется пара состояний $\tilde{\alpha}$, $\tilde{\beta} \in \mathfrak{M}_n$ такая, что $G_{k_2;\, g^s}(\tilde{\alpha}) = G_{k_2;\, g^s}(\tilde{\beta})$. Но оператор $F_{k;\, f^r} \in \Pi$ осуществляет взаимно-однозначное отображение множества $\mathfrak{M}_n$ на себя, и, следовательно, найдутся состояния $\tilde{\alpha}'$, $\tilde{\beta}' \in \mathfrak{M}_n$, для которых $F_{k_1;\, f^r}(\tilde{\alpha}') = \tilde{\alpha}$ и $F_{k_1;\, f^r}(\tilde{\beta}') = \tilde{\beta}$. Тогда для композиции $H_{k;\, h^t} = F_{k_1;\, f^r} \times G_{k_2;\, g^s}$ выполняется равенство $H_{k;\, h^t}(\tilde{\alpha}') = H_{k;\, h^t}(\tilde{\beta}')$ откуда следует, что $h = (f \times g) \overline{\in} \Phi_n$.

2. $f \overline{\in} \Phi_n$. Тогда для оператора $F_{k_1;\, f^r}$ найдется пара состояний $\tilde{\alpha}$, $\tilde{\beta} \in \mathfrak{M}_n$ таких, что $F_{k_1;\, f^r}(\tilde{\alpha}) = F_{k_1;\, f^r}(\tilde{\beta})$, откуда следует, что $H_{k;\, h^t}(\tilde{\alpha}) = H_{k;\, h^t}(\tilde{\beta})$, где $H_{k;\, h^t} = F_{k_1;\, f^r} \times G_{k_2;\, g^s}$ и поэтому $h = (f \times g) \overline{\in} \Phi_n$.

Справедливо следующее утверждение [8].

Теорема 4.9. *Полугруппа Φ не имеет конечных систем образующих.*

Доказательство. Достаточно показать, что подполугруппы, входящие в множество $\{\Phi_n\}$ $(n = 1, 2, \ldots)$, различны. Для всякой подполугруппы Φ_{n+k} $(k = 1, 2, \ldots)$ найдется функция $f \in \Phi_n$ такая, что $f \overline{\in} \Phi_{n+k}$. Действительно, рассмотрим функцию

$$f(\tau_0, \tau_1, \ldots, \tau_{n+k}) = \begin{cases} \tau_0, & \text{если } \tau_0 = \tau_n,\ \tau_1 = \tau_{n+1},\ \ldots,\ \tau_k = \tau_{n+k}, \\ 0 & \text{во всех остальных случаях.} \end{cases}$$

Очевидно, $f(\tau_0, \tau_1, \ldots, \tau_{n+k}) \in \Phi_n$, так как $F_{0;\, f^{n+k}}(\tilde{\alpha}) = \tilde{\alpha}$ для

любого состояния $\tilde{\alpha} \in \mathfrak{M}_n$, и потому оператор $F_{0;\, f^{n+k}}$, а значит и любой оператор $F_{k;\, f^{n+k}}$, является взаимно-однозначным отображением множества $\mathfrak{M}_n$ на себя. Однако $f(\tau_0, \tau_1, \ldots, \tau_{n+k}) \overline{\in} \Phi_{n+k}$, так как для оператора $F_{0;\, f^{n+k}}$ $F_{0;\, f^{n+k}}(\tilde{\alpha}^0) = F_{0;\, f^{n+k}}(\tilde{\beta}^0) = \tilde{\alpha}^0$, где

$$\tilde{\alpha}^0 = (\ldots, 0, \ldots, 0, 0, \ldots, 0, \ldots),\ \tilde{\beta}^0 = (\ldots, \underbrace{1, 0, \ldots, 0}_{n+k}, \ldots$$

$$\ldots, \underbrace{1, 0, \ldots, 0}_{n+k}, \underbrace{1, 0, \ldots, 0}_{n+k}, \ldots, \underbrace{1, 0, \ldots, 0, \ldots}_{n+k}) \in \mathfrak{M}_{n+k}.$$

Таким образом, существует бесконечная совокупность $\{\Phi_n\}$ $(n = 1, 2, \ldots)$ изолированных подполугрупп полугруппы Φ. Но алгебра с бесконечной совокупностью изолированных подалгебр не содержит конечных систем образующих (см. § 3.5, теорему 3.15, следствие 2). Теорема доказана.

Следствие. Полугруппа Π однорегистровых операторов не имеет конечных систем образующих.

Следствие справедливо в силу изоморфизма полугруппы Φ с фактор-полугруппой Π/Φ (см. теорему 4.8).

Можно рассматривать также многорегистровые операторы [131]. Пусть $\{X^s \mid s = 1, 2, \ldots\}$ — счетная совокупность бесконечных в обе стороны регистров, где $X^s = (\ldots, x_1^s, x_0^s, x_{-1}^s, \ldots)$, причем x_i^s принимает значения из множества $E_k = (0, 1, \ldots, k-1)$ $(-\infty < i < +\infty)$. Как и ранее, под состоянием регистра X^s понимается бесконечная в обе стороны последовательность из элементов множества E_k: $\tilde{\alpha}^s = (\ldots, \alpha_1^s, \alpha_0^s, \alpha_{-1}^s, \ldots)$, где α_i^s — состояние (значение) i-го элемента регистра. Рассмотрим последовательность целых чисел $k_1, \ldots, k_m$ и функцию k-значной алгебры Поста (см. § 3.1, пример 10) $f(\tau_1^1, \ldots, \tau_{r_1}^1; \tau_1^2, \ldots, \tau_{r_2}^2; \ldots; \tau_1^m, \ldots \ldots, \tau_{r_m}^m)$, где $(\tau_1^1, \tau_{r_1}^1)$, $\ldots$, $(\tau_1^m, \tau_{r_m}^m)$ — пары нефиктивных аргументов. Преобразование $F^n_{k_1, \ldots, k_m;\, f}(x^1, \ldots, x^m)$ называется **многорегистровым оператором** на регистрах $X^1, \ldots, X^m$, если новое состояние $\tilde{\beta}^n = (\ldots, \beta_1^n, \beta_0^n, \beta_{-1}^n, \ldots)$ регистра X^n определяется по формуле

$$\beta_i^n = f(\alpha^1_{i+k_1}, \alpha^1_{i+k_1+1}, \ldots, \alpha^1_{i+k_1+r_1-1}, \ldots \ldots, \alpha^m_{i+k_m}, \alpha^m_{i+k_m+1}, \ldots, \alpha^m_{i+k_m+r_m-1}), \tag{4.46}$$

где $\{\tilde{\alpha}^j \mid j = 1, 2, \ldots, m\}$ — состояния регистров X^j, $k_1, \ldots, k_m$ — коэффициенты оператора $F_{k_1, \ldots, k_m;\, f}$, а $f(\tau_1^1, \ldots, \tau_{r_m}^m)$ — порождающая функция данного оператора, причем верхний индекс ее аргумента указывает номер регистра, к которому он отнесен. (Следует заметить, что, в частности, $n \in \{1, \ldots, m\}$, т. е. регистр

X^n является одним из регистров $X^1, \ldots, X^m$). Очевидно, одно-регистровые операторы являются частным случаем многорегистровых. Примерами многорегистровых операторов являются операторы

$$\bigwedge^s_{0,\,0;\,f_1}(X^1, X^2),\ \bigvee^s_{0,\,0,\,f_2}(X^1; X^2),\ \bigoplus^s_{0,\,0,\,f_3}(X^1, X^2),$$

где $f_1(\tau_1^1, \tau_1^2) = \tau_1^1 \wedge \tau_1^2$, $f_2(\tau_1^1, \tau_1^2) = \tau_1^1 \vee \tau_1^2$, $f_3(\tau_1^1, \tau_1^2) = \tau_1^1 \oplus \tau_1^2$ — реализующие известные поразрядные логические операции — конъюнкцию, дизъюнкцию, сумму (mod 2). В терминах введенных операторов можно представить сложение и другие арифметические операции машины (см. § 4.5). В качестве примера приведем микропрограмму операции $S'(X^1, X^2)$ — сложения содержимых регистров X^1, X^2 с хранением результата на регистре X^1, написанную на адресном языке [128]:

$$M: \bigoplus^1_{0,\,0;\,f_3}(X^1, X^2);\ \bigwedge^2_{0,\,0;\,f_1}(X^1, X^2),$$
$$P\{[\theta^2]\}M1,$$
$$C^2_{-1};\ l^0_0 M,$$
$$M1: \ldots$$

Здесь $[\theta^s]$ — логическое условие, истинное, когда регистр X_s находится в нулевом состоянии $\tilde{\alpha}^s_0 = (\ldots, 0, 0, \ldots)$, и ложное в противном случае. Оператор $P\{\theta\}M1$ означает переход на метку $M1$, если условие θ истинно, и — пустой оператор в противном случае.

Регулярное представление данной микропрограммы в системе алгоритмических алгебр, образующими которых служат приведенные операторы и условия, имеет вид

$$S^1(X^1, X^2) = \{\bigoplus^1_{\theta\ 0,\,0;\,f_3}(X^1, X^2) \cup \bigwedge_{-1,\,-1;\,f_1}(X^1, X^2)\}!,$$

где $\cup$ — дизъюнкция операторов (см. § 4.2), ! — оператор останова.

Арифметические операции умножения, вычитания, деления реализуются с помощью сложения и операторов сдвига $C^s_{k,\,i^s_k}$ и инверсии $I^s_{0;\,z}$.

Как и в случае с одно-регистровым оператором, можно ввести ассоциативную операцию композиции многорегистровых операторов и рассмотреть соответствующую полугруппу Π_M.

На полугруппе Π_M задаем бинарное отношение ρ так, что два оператора F, G находятся в отношении ρ, если существует последовательность сдвигов, переводящая один оператор в другой. Отношение ρ является конгруэнцией, так что справедлив следующий аналог теоремы 4.8.

Теорема 4.10. *Фактор-полугруппа Π_M/ρ изоморфна полугруппе Φ_M порождающих функций многорегистровых операторов.*

Многорегистровые операторы описывают лишь микрооперации с независимыми сколь угодно далекими друг от друга разрядами (в случае бесконечных регистров). Известно, что даже в машинах с традиционной структурой широко используются некоторые микрооперации, не удовлетворяющие этому условию. К ним относятся микрооперации прибавления единицы к коду на двоичном регистре (двоичный счетчик) и сложения (параллельного) двух чисел. Для описания такого рода микроопераций понятие периодически определенного преобразования необходимо расширить с помощью так называемых вспомогательных переменных. Для каждого разряда основного регистра со значением α_i добавляется некоторое конечное число вспомогательных переменных $x_i^{(1)}, x_i^{(2)}, \ldots, x_i^{(n)}$. В общем случае алфавиты, которым принадлежат значения соответствующих вспомогательных переменных, могут быть различными (эти значения существуют только в момент выполнения микроопераций в качестве сигналов в некоторых точках соответствующих схем).

Обозначим через α_i и β_i значения i-го разряда рассматриваемого основного регистра соответственно до и после выполнения преобразования. Тогда периодически определенное преобразование с n вспомогательными переменными на этом регистре может быть задано системой $n+1$ соотношений:

$$\beta_i = f_0(\alpha_{i_1+i}, x^{(1)}_{i_1+i}, \ldots, x^{(n)}_{i_1+i};\ \alpha_{i_2+i}, x^{(1)}_{i_2+i}, \ldots$$
$$\ldots, x^{(n)}_{i_2+i};\ \ldots;\ \alpha_{i_q+i}, x^{(1)}_{i_q+i}, \ldots, x^{(n)}_{i_q+i}),$$
$$x_i^{(1)} = f_1(\alpha^{(1)}_{i_1+i}, x^{(1)}_{i^{(1)}_1+i}, \ldots, x^{(n)}_{i^{(1)}_1+i};\ \alpha^{(1)}_{i_2+i}, x^{(1)}_{i^{(1)}_2+i}, \ldots$$
$$\ldots, x^{(n)}_{i^{(1)}_2+i};\ \alpha^{(1)}_{i_{q^{(1)}}+i}, x^{(1)}_{i^{(1)}_{q^{(1)}}+i}, \ldots, x^{(n)}_{i^{(1)}_{q^{(1)}}+i}),$$
$$\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots$$
$$x_i^{(n)} = f_n(\alpha^{(n)}_{i_1+i}, x^{(1)}_{i^{(n)}_1+i}, \ldots, x^{(n)}_{i^{(n)}_1+i};\ \alpha^{(n)}_{i_2+i}, x^{(1)}_{i^{(n)}_2+i}, \ldots$$
$$\ldots, x^{(n)}_{i^{(n)}_2+i};\ \ldots;\ \alpha^{(n)}_{i_{q^{(n)}}+i}, x^{(1)}_{i^{(n)}_{q^{(n)}}+i}, \ldots, x^{(n)}_{i^{(n)}_{q^{(n)}}+i}).$$

Эти соотношения, для того чтобы из них действительно можно было найти значения вспомогательных переменных, должны удовлетворять некоторым дополнительным условиям. В некоторых (начальных) разрядах значения вспомогательных переменных должны быть заданы. Простейшее из таких условий при ограниченности слева нумерующего отрезка регистра состоит в том, что все индексы вида i^t_k следует считать отрицательными. При ограниченности нумерующего отрезка справа все эти индексы достаточно считать положительными. В первом случае в качестве на-

чального выбирается самый младший разряд регистра, во втором — самый старший. Если необходимо, к началу добавляется еще некоторое количество фиктивных разрядов.

Одним из наиболее распространенных преобразований такого рода является **пересчет на регистре**. В случае двоичного бесконечного влево регистра X со значениями переменных 0 и 1 пересчет может быть задан с помощью булевых уравнений

$$\beta_i = \alpha_i \oplus p_i, \quad p_i = \alpha_{i-1} p_{i-1},$$

где значение вспомогательной переменной p_i есть не что иное, как перенос в i-й разряд регистра из $(i-1)$-го разряда. Если $X = \{\ldots x_3 x_2 x_1\}$, то для осуществления пересчета необходимо положить $p_1 = 1$. Дополнения регистра фиктивными разрядами при таком способе введения вспомогательной переменной не требуется.

Периодически определенные преобразования с вспомогательными переменными на регистре могут быть распространены на многорегистровые операторы. Примером является **сложение** целых чисел, расположенных в младших разрядах бесконечных влево регистров X^1, X^2 с хранением результата на регистре X^2. Пусть $\tilde{\alpha}^1$, $\tilde{\beta}^2$ — состояние регистров X^1, X^2, а $\tilde{\gamma}^2$ — результат преобразования на регистре X^2. Тогда сложение может быть задано следующей системой уравнений:

$$\gamma_i^2 = \alpha_i^1 \oplus \beta_i^2 \oplus p_i,$$

$$p_i = \alpha_{i-1}^1 \cdot \beta_{i-1}^2 \oplus \alpha_{i-1}^1 p_{i-1} \oplus \beta_{i-1}^2 p_{i-1} \quad (i = 1, 2, \ldots).$$

Предполагается, что начальное значение переноса p_1 равно нулю.

§ 4.4. Модифицированные алгебры Поста

Модифицированная алгебра Поста, соответствующая полугруппе П однорегистровых операторов, может быть получена из k-значной алгебры Поста (см. § 3.1, пример 10) в результате замены операции $\times$ бинарной операцией $\times$ — композиции функций, которая определяется равенством

$$(f \times g)(x_1, x_2, \ldots, x_{n+m-1}) = f(g(x_1, x_2, \ldots, x_n), g(x_2, x_3, \ldots, x_n, x_{n+1}), \ldots, g(x_m, x_{m+1}, \ldots, x_{m+n-1})), \tag{4.48}$$

где f, g — произвольные m- и n-местные функции из Φ_{E_k}. По определению полагаем $(f \times g)(x_1, x_2, \ldots, x_m) = f(g(x_1), g(x_2), \ldots, g(x_m))$ для случая, когда g — одноместная функция из Φ_{E_k}. Обозначим полученную таким образом модифицированную алгебру $\Phi^0_{E_k}$.

Кроме операции $\times$ алгебра $\Phi^0_{E_k}$ содержит также все унарные операции ζ, τ, Δ, ∇ k-значной алгебры Поста, которые позволяют выполнять отождествление и перестановку аргументов функции f, а также приписывание фиктивных аргументов.

Заметим, что схемная интерпретация перечисленных унарных операций в применении к преобразованиям, описанным в § 4.3, отличается простотой, так как связана лишь с пересоединением и переименованием элементов регистра.

Нетрудно видеть, что k-значная алгебра Поста Φ_{E_k} и модифицированная алгебра $\Phi^0_{E_k}$ представляют собой однотипные алгебраические системы (см. § 3.1). Каждая из этих алгебр содержит четыре унарных и одну бинарную операции.

Однако алгебры $\Phi^0_{E_k}$ и Φ_{E_k} не эквивалентны, тогда как каждая операция алгебры $\Phi^0_{E_k}$ производна в алгебре Φ_{E_k}. Обратное утверждение неверно. Можно утверждать, что система операций алгебры $\Phi^0_{E_k}$ слабее системы операций алгебры Φ_{E_k} в том смысле, что произвольная функция f k-значной логики, построенная в алгебре $\Phi^0_{E_k}$ из функций $f_1, f_2, \ldots, f_s$, может быть получена и в алгебре Φ_{E_k} из тех же функций, но не наоборот. Отсюда следует, в частности, что если $\mathfrak{M}$ является подалгеброй алгебры Φ_{E_k} (функционально-замкнутым классом k-значной логики [133]), то $\mathfrak{M}$ образует подалгебру алгебры $\Phi^0_{E_k}$.

Например, рассмотренные в § 3.4 подалгебры монотонных, самодвойственных, линейных функций и функций, сохраняющих константы, являются также подалгебрами алгебры $\Phi^0_{E_2}$. В дальнейшем будем обозначать ее Φ^0. Вместе с тем, если $\Sigma = \{f_1, f_2, \ldots, f_s\}$ — система образующих алгебры $\Phi^0_{E_k}$, то Σ является также системой образующих алгебры Φ_{E_k}.

Рассмотрим далее бинарную операцию $\boxtimes$, определяемую следующим равенством:

$$(f \boxtimes g)(x_1, x_2, \ldots, x_{mn}) = f(g(x_1, x_2, \ldots, x_n), g(x_{n+1}, x_{n+2}, \ldots, x_{2n}), \ldots, g(x_{(m-1)n+1}, x_{(m-1)n+2}, \ldots, x_{mn})).$$

Полагаем $(f \boxtimes g)(x_1, x_2, \ldots, x_m) = f(g(x_1), g(x_2), \ldots, g(x_m))$, если g — одноместная функция. Операция $\boxtimes$ производна в алгебре $\Phi^0_{E_k}$. Действительно, пусть $f, g \in \Phi^0_{E_k}$ — произвольные m- и n-местные функции. Используя операцию ∇ (см. § 3.1) и присоединяя к функции f $(m-1)(n-1)$ фиктивных переменных, с помощью операций ζ и τ осуществляем перестановку переменных так, что в результате получаем функцию

$$f'(x_1, x_2, \ldots, x_{(m-1)n+1}) = f(x_1, x_{n+1}, x_{2n+1}, \ldots, x_{(m-1)n+1}).$$

Применяя операцию $\times$, получаем

$$(f' \times g)(x_1, x_2, \ldots, x_{mn}) = f(g(x_1, x_2, \ldots, x_n), g(x_{n+1}, x_{n+2}, \ldots, x_{2n}), \ldots, g(x_{(m-1)n+1}, x_{(m-1)n+2}, \ldots, x_{mn}) = (f \boxtimes g)(x_1, x_2, \ldots, x_{mn}).$$

Далее рассматривается алгебра Φ^0 булевых функций [113], являющаяся модификацией двузначной алгебры Поста (алгебры логики). Многие из полученных результатов могут быть распространены на k-значную алгебру.

Теорема 4.11. *Алгебра Φ^0 является алгеброй с конечной системой образующих.*

Доказательство. Покажем, что система функций

$$\{xy;\ x \oplus y;\ x \oplus y \oplus z \oplus 1\} \tag{4.49}$$

является системой образующих алгебры Φ^0. Пусть

$$f(x_1, \ldots, x_n) = u_1 \oplus u_2 \oplus \cdots \oplus u_h \oplus a_0 \tag{4.50}$$

является произвольной булевой функцией, представленной полиномом Жегалкина l-й степени (см. § 3.4). Это означает, что в (4.50) содержится по крайней мере одно слагаемое $u_l = x_{i_1} \cdot x_{i_2} \cdot \ldots \cdot x_{i_l}$ длины l и не содержится слагаемых большей длины. Прежде всего покажем, что с помощью функций $x \oplus y$, $x \oplus y \oplus z \oplus 1$ может быть построена любая линейная функция, в частности функция

$$\varphi_1(x_1, x_2, \ldots, x_h) = x_1 \oplus x_2 \oplus \cdots \oplus x_h \oplus a_0.$$

Действительно, с помощью функций $x \oplus y$ и $x \oplus y \oplus z \oplus 1$ может быть построена любая линейная функция, существенно зависящая от четного числа переменных. Но

$$(x \oplus y \oplus z \oplus 1) \oplus (y \oplus z \oplus u \oplus 1) \oplus (z \oplus u \oplus v \oplus 1) \oplus 1 = x \oplus z \oplus v, \tag{4.51}$$

а с помощью функций $x \oplus z \oplus v$ и $x \oplus y \oplus z \oplus 1$ может быть построена любая линейная функция, существенно зависящая от нечетного числа переменных. Используя xy, нетрудно найти функцию $\psi(x_1, x_2, \ldots, x_l) = x_1 \cdot x_2 \cdot \ldots \cdot x_l$. Применяя $\boxtimes$ к функциям φ и ψ, получаем

$$f'(x_1, x_2, \ldots, x_{kl}) = \varphi(\psi(x_1, x_2, \ldots, x_l), \psi(x_{l+1}, \ldots, x_{2l}), \ldots, \psi(x_{(k-1)l+1}, \ldots, x_{kl})) = x_1 \cdot x_2 \cdot \ldots \cdot x_l \oplus \bigoplus x_{l+1}x_{l+2} \ldots x_{2l} \oplus \cdots \oplus x_{(k-1)l+1}x_{(k-1)l+2} \ldots x_{kl} \oplus a_0. \tag{4.52}$$

Отождествляя в формуле (4.52) соответствующие переменные, находим полином Жегалкина функции $f(x_1, x_2, \ldots, x_n)$. Теорема доказана.

Пример. Используя систему (4.49), строим функцию

$$f(x_1, x_2, \ldots, x_6) = x_1x_2x_5 \oplus x_1x_3x_6 \oplus x_3x_5 \oplus x_4x_6 \oplus x_6 \oplus 1.$$

Применяя операцию $\boxtimes$ к функциям $x \oplus y \oplus z$ (см. (4.51)) и $x \oplus y \oplus z \oplus 1$, получаем $h(x_1, x_2, \ldots, x_9) = x_1 \oplus x_2 \oplus \cdots \oplus x_9 \oplus 1$. Отождествляя $x_6 \equiv x_7 \equiv x_8 \equiv x_9$, находим

$$\varphi(x_1, x_2, \ldots, x_5) = x_1 \oplus x_2 \oplus x_3 \oplus x_4 \oplus x_5 \oplus 1.$$

С помощью операций конъюнкции и композиции строим функцию $\psi(x_1, x_2, x_3) = (x_1x_2)(x_2x_3) = x_1x_2x_3$, и, наконец, применяя $\boxtimes$ к функциям φ и ψ, получаем

$$h'(x_1, x_2, \ldots, x_{15}) = \varphi(\psi(x_1, x_2, x_3), \psi(x_4, x_5, x_6), \ldots, \psi(x_{13}, x_{14}, x_{15})) = x_1x_2x_3 \oplus x_4x_5x_6 \oplus x_7x_8x_9 \oplus x_{10}x_{11}x_{12} \oplus x_{13}x_{14}x_{15} \oplus 1.$$

Отождествляя в полученном выражении переменные, находим функцию $f(x_1, x_2, \ldots, x_6)$.

Рассмотрим некоторые важные классы булевых функций [134]:

класс C_α всех **α-функций**, т. е. $f(x_1, \ldots, x_n) \in C_\alpha$ тогда и только тогда, когда $f(x, x, \ldots, x) = x$;

класс C_β всех **β-функций**, т. е. $f(x_1, \ldots, x_n) \in C_\beta$ тогда и только тогда, когда $f(x, x, \ldots, x) = 1$;

класс C_γ всех **γ-функций**, т. е. $f(x_1, \ldots, x_n) \in C_\gamma$ тогда и только тогда, когда $f(x, x, \ldots, x) = 0$;

класс $C_{\bar{x}}$ всех **δ-функций**, т. е. $f(x_1, x_2, \ldots, x_n) \in C_{\bar{x}}$ тогда и только тогда, когда $f(x, x, \ldots, x) = \bar{x}$.

Примерами α-функции служат конъюнкция и дизъюнкция; β-функций — импликация и отношение эквивалентности; γ-функции — сумма по mod 2 и обратная антиимпликация; δ-функции — отрицание, штрих Шеффера, стрелка Пирса (см. § 3.4, табл. 2).

Нетрудно видеть, что классы C_α, C_β, C_γ являются подалгебрами алгебры Φ^0. Действительно, любая суперпозиция α-функций также является α-функцией, так что $C_\alpha = C_2 \cap C_3$ образует подалгебру алгебры логики, а значит, и подалгебру Φ^0, где C_2, C_3 — классы функций, сохраняющих константы соответственно 1 и 0 (см. § 3.4)[2]. Докажем, что класс C_β образует подалгебру алгебры Φ^0 (для класса C_γ доказательство проводится аналогично).

Класс C_β замкнут по операциям отождествления, перестановки и добавления (фиктивных) аргументов, поэтому остается показать, что если функции $f(x_1, \ldots, x_m)$ и $g(x_1, \ldots, x_n)$ принадлежат C_β, то их композиция $h(x_1, \ldots, x_{m+n-1})$ также принадлежит классу C_β. Но это очевидно, так как

$$h(x, \ldots, x) = f(g(x, \ldots, x), g(x, \ldots, x), \ldots, g(x, \ldots, x)) = f(1, 1, \ldots, 1) = 1.$$

[2] Здесь и далее для подалгебр алгебры Φ^0, которые являются замкнутыми классами алгебры логики, сохранены обозначения Поста [134].

Пусть $C_{\tilde{x}} = C_4 \cup C_{\bar{x}}$, а $C_a = C_\beta \cup C_\gamma$. Очевидно, $f(x_1, \ldots, x_n) \in C_{\tilde{x}}$ тогда и только тогда, когда полином Жегалкина функции f содержит нечетное число отличных от констант слагаемых, и $f(x_1, \ldots, x_n) \in C_a$ тогда и только тогда, когда полином Жегалкина функции f содержит четное число отличных от констант слагаемых. Таким образом, алгебра Φ^0 является прямой суммой множеств $C_{\tilde{x}}$ и C_a (см. § 1.2), т. е. $\Phi^0 = C_{\tilde{x}} \cup C_a$, причем $C_{\tilde{x}} \cap C_a = \varnothing$.

Лемма 4.3. *Класс C_a образует подалгебру алгебры Φ^0.*

Действительно, класс C_a замкнут по операциям отождествления, перестановки и добавления (фиктивных) аргументов. Пусть функции $f(x_1, \ldots, x_m)$ и $g(x_1, \ldots, x_n)$ принадлежат C_a. Тогда их композиция $h(x_1, x_2, \ldots, x_{m+n-1})$ также принадлежит классу C_a, так как $h(x, x, \ldots, x) = f(g(x, \ldots, x), \ldots, g(x, \ldots, x)) = f(a, a, \ldots, a)$, где $a = g(x, \ldots, x)$. Лемма доказана.

Теорема 4.12. *Класс $C_{\tilde{x}}$ образует изолированную подалгебру алгебры Φ^0.*

Доказательство. Покажем сначала, что класс $C_{\tilde{x}}$ является подалгеброй алгебры Φ^0. Очевидно, что класс $C_{\tilde{x}}$ замкнут по операциям отождествления, перестановки и добавления (фиктивных) аргументов. Пусть

$$f(x_1, \ldots, x_m),\ g(x_1, \ldots, x_n) \in C_{\tilde{x}}.$$

Композиция $h(x_1, \ldots, x_{m+n-1})$ функций f и g также принадлежит классу $C_{\tilde{x}}$. Действительно, если $g(x, \ldots, x) = x$, то

$$h(x, x, \ldots, x) = f(g(x, \ldots, x), g(x, \ldots, x), \ldots, g(x, \ldots, x)) = f(x, \ldots, x) = x \oplus a_0,$$

где a_0 — свободный член в полиноме Жегалкина функции f. Если $g(x, x, \ldots, x) = \bar{x}$, то

$$h(x, x, \ldots, x) = f(g(x, \ldots, x), g(x, \ldots, x), \ldots, g(x, \ldots, x)) = f(\bar{x}, \bar{x}, \ldots, \bar{x}) = \bar{x} + a_0,$$

так как полином Жегалкина функции f содержит нечетное число отличных от констант слагаемых.

Перейдем к доказательству изолированности подалгебры $C_{\tilde{x}}$. Поскольку Φ^0 является прямой суммой множеств C_a и $C_{\tilde{x}}$ и класс C_a образует подалгебру алгебры Φ^0, достаточно рассмотреть следующие два случая:

а) $f(x_1, \ldots, x_m) \in C_a$; $g(x_1, \ldots, x_n) \in C_{\tilde{x}}$, тогда для композиции $h(x_1, \ldots, x_{m+n-1})$ функций f и g справедливо равенство

$$h(x, \ldots, x) = f(g(x, \ldots, x), \ldots, g(x, \ldots, x)) = a_0,$$

где a_0 — свободный член полинома $f(x_1, \ldots, x_m)$, откуда следует, что $h(x_1, \ldots, x_{m+n-1})$ принадлежит алгебре C_a;

б) $f(x_1, \ldots, x_m) \in C_{\bar{x}}$; $g(x_1, \ldots, x_n) \in C_a$, тогда $h(x, \ldots, x) = f(a, \ldots, a)$, где $a = g(x, \ldots, x)$, откуда следует, что и в этом случае композиция $h(x_1, \ldots, x_{m+n-1})$ функций f и g принадлежит подалгебре C_a. Теорема доказана.

Следствие 1. Всякая система образующих алгебры Φ^0 содержит подсистему, которая является системой образующих алгебры $C_{\tilde{x}}$.

Элемент, порождающий всю алгебру $\mathfrak{A}$, называется **универсальным** в этой алгебре.

Следствие 2. Не существует функций, универсальных в алгебре Φ^0.

Пусть $\mathfrak{M}^a = C_a \cup \mathfrak{M}$, где $\mathfrak{M} \subset C_{\tilde{x}}$ — некоторое множество функций.

Следствие 3. Класс $\mathfrak{M}^a$ является подалгеброй алгебры Φ^0 тогда и только тогда, когда $\mathfrak{M}$ также образует подалгебру алгебры Φ^0.

Следствие 4. Класс $\mathfrak{M}^a$ образует максимальную подалгебру алгебры Φ^0 тогда и только тогда, когда $\mathfrak{M}$ является максимальной подалгеброй алгебры $C_{\tilde{x}}$.

Перейдем к установлению критерия функциональной полноты в алгебре Φ^0. Рассмотрим классы булевых функций $C_4^a = C_a \cup C_4$, $D_3^a = C_a \cup D_3$, где D_3 — подалгебра самодвойственных функций алгебры логики (см. § 3.4).

Лемма 4.4. *Класс C_4^a образует подалгебру алгебры Φ^0.*

Лемма 4.5. *Класс D_3^a образует подалгебру алгебры Φ^0.*

Справедливость лемм 2 и 3 очевидна, так как классы C_4 и D_3 являются подалгебрами алгебры Φ^0 и, кроме того, C_4, $D_3 \subset C_{\tilde{x}}$ (см. теорему 4.12, следствие 3).

Напомним, что функция $f^*(x_1, x_2, \ldots, x_n) = \bar{f}(\bar{x}_1, \bar{x}_2, \ldots, \bar{x}_n)$ называется **двойственной** функции $f(x_1, x_2, \ldots, x_n)$; если $f^*(x_1, x_2, \ldots, x_n) = f(x_1, x_2, \ldots, x_n)$, то функция f **самодвойственна** (см. § 3.4).

Пусть $\mathfrak{M}$ — некоторое множество булевых функций. Множество $\mathfrak{M}^*$ всех функций, двойственных функциям из $\mathfrak{M}$, называется **двойственным множеству** $\mathfrak{M}$. Если $\mathfrak{M}^* = \mathfrak{M}$, то множество $\mathfrak{M}$ называется **самодвойственным**.

Пусть $M_4 = C_4 \cap M_1$, где M_1 — подалгебра монотонных функций алгебры логики (см. § 3.4). Поскольку C_4 и M_1 являются подалгебрами алгебры Φ^0, класс M_4 всех монотонных a-функций — также подалгебра алгебры Φ^0.

Введем класс M_5 такой, что $f(x_1, x_2, \ldots, x_n) \in M_5$ тогда и только тогда, когда одна из функций $f(x_1, \ldots, x_n)$, $\bar{f}(x_1, \ldots, x_n)$ принадлежит классу M_4. Нетрудно убедиться в том, что класс M_4,

а значит и M_5, образуют самодвойственное множество $M_4^* = M_4$, $M_5^* = M_5$.

Пусть $M_5^a = C_a \cup M_5$, тогда справедливо такое утверждение.

Лемма 4.6. *Класс M_5^a образует подалгебру алгебры Φ^0.*

Доказательство. На основании включения $M_5 \subset C_{\tilde{x}}$ достаточно показать, что класс M_5 образует подалгебру алгебры Φ^0. Очевидно, что класс M_5 замкнут по операциям отождествления, перестановки и добавления (фиктивных) аргументов. Пусть функции $f(x_1, \ldots, x_m)$ и $g(x_1, \ldots, x_n)$ принадлежат M_5. Покажем, что их композиция

$$h(x_1, \ldots, x_{m+n-1}) = f(g(x_1, \ldots, x_n), \ldots, g(x_m, x_{m+1}, \ldots \ldots, x_{m+n-1}))$$

принадлежит классу M_5. Возможны два случая:

а) $g(x_1, \ldots, x_n) \in M_4$, тогда $h(x_1, \ldots, x_{m+n-1}) \in M_5$, так как класс M_4 образует подалгебру алгебры Φ^0;

б) $g(x_1, \ldots, x_n) \overline{\in} M_4$, тогда согласно определению класса M_5 $g(x_1, \ldots, x_n) = \bar{g}_1(x_1, \ldots, x_n)$, где $g_1(x_1, \ldots, x_n) \in M_4$. Отсюда следует, что

$$h(x_1, \ldots, x_{m+n-1}) = f(\bar{g}_1(x_1, \ldots, x_n), \ldots, \bar{g}_1(x_m, \ldots$$

$$\ldots, x_{m+n-1})) = \bar{f}^*(g_1(x_1, \ldots, x_n), \ldots, g_1(x_m, \ldots, x_{m+n-1})),$$

где f^* — функция, двойственная f. Таким образом, ввиду самодвойственности и замкнутости класса M_5 относительно отрицания рассматриваемый случай сведен к предыдущему. Лемма доказана.

Рассмотрим класс $C_{\tilde{x}}^k$, который получается в результате присоединения к классу $C_{\tilde{x}}$ булевых констант.

Лемма 4.7. *Класс $C_{\tilde{x}}^k$ образует подалгебру алгебры Φ^0.*

Справедливость леммы следует из теоремы 4.12, а также из равенства композиции произвольной функции с константой одной из констант.

Справедлива теорема, устанавливающая критерий функциональной полноты (см. § 3.2) в алгебре Φ^0. Из нее следует описание всех максимальных подалгебр алгебры Φ^0, которыми являются подалгебры C_4^a, D_3^a, M_5^a, $C_{\tilde{x}}^k$.

Теорема 4.13. *Для того чтобы система функций $f_1, f_2, \ldots, f_s$ являлась системой образующих в алгебре Φ^0, необходимо и достаточно, чтобы в этой системе содержалась по крайней мере одна функция, 1) не принадлежащая C_4^a, 2) не принадлежащая D_3^a, 3) не принадлежащая M_5^a, 4) не принадлежащая $C_{\tilde{x}}^k$.*

Доказательство. *Необходимость* следует из лемм 4.4—4.7. *Достаточность*. Пусть

$$f_1(x_1, x_2, \ldots, x_{n_1}) \overline{\in} C_4^a, \tag{4.53}$$

$$f_2(x_1, x_2, \ldots, x_{n_2}) \overline{\in} D_3^a, \tag{4.54}$$

$$f_3(x_1, x_2, \ldots, x_{n_3}) \overline{\in} M_5^a, \tag{4.55}$$

$$f_4(x_1, x_2, \ldots, x_{n_4}) \overline{\in} C_{\tilde{x}}^k. \tag{4.56}$$

Используя (4.53), построим отрицание

$$\bar{x} = f_1(x, x, \ldots, x). \tag{4.57}$$

Выбираем функцию $f_2(x_1, x_2, \ldots, x_{n_2}) \overline{\in} D_3^a$. В силу (4.57)

$$f_2(x, \ldots, x) = x, \tag{4.58}$$

так как в противном случае этим свойством обладала бы функция $\bar{f}_2(x_1, x_2, \ldots, x_{n_2})$, которая также не принадлежала бы классу D_3^a. Так как функция $f_2(x_1, x_2, \ldots, x_{n_2})$ несамодвойственна, можно указать пару противоположных наборов $\tilde{\alpha} = (a_1, a_2, \ldots, a_{n_2})$ и $\tilde{\alpha}' = (\bar{a}_1, \bar{a}_2, \ldots, \bar{a}_{n_2})$ таких, что $f_2(\tilde{\alpha}) = f_2(\tilde{\alpha}') = C$. Разбиваем переменные, от которых зависит функция f_2, на две группы. К первой группе относим все переменные x_i такие, что $a_i = 0$, ко второй — все переменные x_i такие, что $a_i = 1$ $(i = 1, 2, \ldots, n_2)$. На основании (4.58) заключаем, что каждая из этих групп непустая. Отождествляя все переменные первой группы с переменной x, а второй — с y, получаем функцию

$$F_2(x, y) = \begin{cases} x \cdot y, & \text{если } C = 0, \\ x \vee y, & \text{если } C = 1. \end{cases} \tag{4.59}$$

Таким образом, используя функцию $f_2(x_1, x_2, \ldots, x_{n_2})$ и учитывая равенство (4.57), можно построить и конъюнкцию, и дизъюнкцию:

$$x \vee y = [(\bar{x} \times xy) \times \bar{x}] = \overline{\bar{x}\bar{y}} \quad \text{и} \quad xy = [(\bar{x} \times (x \vee y)) \times \bar{x}] = \overline{\bar{x} \vee \bar{y}}, \tag{4.60}$$

с помощью которых нетрудно найти функцию

$$\varphi(x_1, x_2, \ldots, x_9) = x_1x_2x_3 \vee x_4x_5x_6 \vee x_7x_8x_9. \tag{4.61}$$

В результате соответствующих отождествлений аргументов функции φ получаем

$$\left.\begin{array}{l} \psi_1(x_1, x_2, x_3) = x_1x_2 \vee x_2x_3 \vee x_1x_3; \ \psi_2(x_1, x_2, x_3) = x_1, x_2, x_3, \\ \psi_3 = (x_1, x_2, x_3) = x_1 \vee x_2 \vee x_3. \end{array}\right\} \tag{4.62}$$

Далее, выбираем функцию $f_3(x_1, x_2, \ldots, x_{n_3}) \overline{\in} M_5^a$. В силу (4.57)

$$f_3(x, x, \ldots, x) = x, \tag{4.63}$$

так как в противном случае этим свойством обладала бы функция $\bar{f}_3(x_1, x_2, \ldots, x_{n_3})$, которая также не принадлежала бы классу M_5^a; функция $f_3(x_1, x_2, \ldots, x_{n_3})$ существенно зависит не менее чем от трех переменных, так как в противном случае она принадлежала бы классу M_5^a (все a-функции от двух аргументов монотонны, см. § 3.4, табл. 2). Вследствие немонотонности функции $f_3(x_1, x_2, \ldots, x_{n_3})$ найдется пара соседних наборов

$$\tilde{\alpha} = (a_1, a_2, \ldots, a_{i-1}, 0, a_{i+1}, \ldots, a_{n_3}),$$
$$\tilde{\beta} = (a_1, a_2, \ldots, a_{i-1}, 1, a_{i+1}, \ldots, a_{n_3})$$

таких, что $f(\tilde{\alpha}) = 1 > 0 = f(\tilde{\beta})$. Разбиваем переменные, от которых зависит функция f_3, на три группы. К первой группе относим переменную x_i, ко второй — те из оставшихся переменных, для которых $a_j = 0$, к третьей — все переменные x_j такие, что $a_j = 1$ $(j = 1, 2, \ldots, i-1, i+1, \ldots, n_3)$. На основании (4.63) $f_3(0, 0, \ldots, 0) = 0$, а $f_3(1, 1, \ldots, 1) = 1$ и поэтому каждая группа переменных непустая. Отождествляя переменную первой группы с переменной x_1, все переменные второй группы с переменной x_2 и все переменные третьей группы с переменной x_3, находим функцию $F_3(x_1, x_2, x_3)$ такую, что $F_3(1, 1, 1) = 1$, $F_3(1, 0, 1) = 0$, $F_3(0, 0, 1) = 1$, $F_3(0, 0, 0) = 0$; применяя операцию $\boxtimes$ к функциям $F_3(x_1, x_2, x_3)$ и $\varphi(x_1, x_2, \ldots, x_8, x_9)$ (см. (4.61)) и отождествляя аргументы аналогично (4.62), можно получить следующую суперпозицию функций:

$$F_3(\psi_1(x_1, x_2, x_3), \psi_2(x_1, x_2, x_3), \psi_3(x_1, x_2, x_3)) = x_1 \oplus x_2 \oplus x_3. \quad (4.64)$$

В справедливости этого равенства нетрудно убедиться, построив таблицы истинности для его левой и правой частей. Применив к построенной функции $x_1 \oplus x_2 \oplus x_3$ отрицание, получим

$$\overline{x_1 \oplus x_2 \oplus x_3} = x_1 \oplus x_2 \oplus x_3 \oplus 1. \quad (4.65)$$

Наконец, выбираем функцию $f_4(x_1, x_2, \ldots, x_{n_4}) \in C_{\tilde{x}}^h$. Возможны два случая. В первом случае

$$f_4(x, x, \ldots, x) = 0. \quad (4.66)$$

Так как $f_4(x_1, x_2, \ldots, x_{n_4}) \not\equiv \text{const}$, найдется набор $\tilde{\alpha} = (a_1, a_2, \ldots, a_{n_4})$ такой, что $f_4(\tilde{\alpha}) = 1$. Разбиваем переменные, от которых зависит функция f_4, на две группы. К первой группе относим все переменные x_i такие, что $a_i = 0$, ко второй — все остальные переменные функции f_4. В силу равенства (4.66) каждая из групп непустая. Отождествляя все переменные первой группы с x, а все переменные второй — с y, получаем функцию

$$F_4(x, y) = \begin{cases} x \oplus y, & \text{если } f_4(\tilde{\alpha}') = 1, \\ \bar{x}y, & \text{если } f_4(\tilde{\alpha}') = 0, \end{cases}$$

где $\tilde{\alpha}'$ — набор, противоположный набору $\tilde{\alpha}$. Используя построенные ранее дизъюнкцию и композицию, получаем

$$h(x, y, z) = \bar{x}y \vee \bar{y}z, \quad (4.67)$$

откуда после отождествления переменных x и z

$$F(x, y) = \bar{x}y \vee \bar{y}x = x \oplus y. \quad (4.68)$$

Во втором случае $f_4(x, x, \ldots, x) = 1$. Выполняя построение, двойственное описанному в первом случае, получаем функцию $x \oplus y \oplus 1$, и в результате применения отрицания

$$\overline{x \oplus y \oplus 1} = x \oplus y. \quad (4.69)$$

Таким образом, функция $x \oplus y$ построена. Исходная система функций сведена к системе $\{xy, x \oplus y, x \oplus y \oplus z \oplus 1\}$, которая согласно теореме 4.11 является системой образующих в алгебре Φ^0. Теорема доказана.

Следствие 1. Всякая подалгебра алгебры Φ^0 (кроме всей алгебры Φ^0) содержится по крайней мере в одном из классов C_4^a, D_3^a, M_5^a, $C_{\tilde{x}}^h$.

Следствие 2. В алгебре Φ^0 существует только четыре максимальные подалгебры: C_4^a, D_3^a, M_5^a, $C_{\tilde{x}}^h$.

Следствие 3. Для того чтобы система функций $f_1, f_2, \ldots, f_s$ являлась системой образующих алгебры $C_{\tilde{x}}$, необходимо и достаточно, чтобы она содержала по крайней мере одну функцию, 1) не принадлежащую классу C_4, 2) не принадлежащую классу D_3, 3) не принадлежащую классу M_5.

Действительно, из доказательства теоремы 4.13 следует, что, используя функции $f_1 \in C_4$, $f_2 \in D_3$, $f_3 \in M_5$, можно построить систему $\{xy, x \oplus y \oplus z \oplus 1\}$, которая порождает любую функцию, принадлежащую подалгебре $C_{\tilde{x}}$. Этот факт можно доказать по аналогии с доказательством теоремы 4.11, если учесть, что каждая функция $f \in C_{\tilde{x}}$ представима полиномом Жегалкина, содержащим нечетное число слагаемых, отличных от константы.

Следствие 4. В алгебре $C_{\tilde{x}}$ существует только три максимальные подалгебры: C_4, M_5, D_3.

Таким образом, алгебра Φ^0 имеет четыре максимальные подалгебры, в отличие от алгебры логики, имеющей пять максимальных алгебр (см. § 3.4).

В работах [114, 115] подробно изучена структура подалгебры алгебры Φ^0. В частности, удалось установить, что в алгебру Φ^0 входят бесконечно порожденные подалгебры, в том числе подалгебры с бесконечным базисом. Отсюда следует, что все подалгебры алгебры Φ^0 являются подалгебрами континуальной мощности (см. § 3.3, теорема 3.8). Это означает, что алгебра Φ^0, как

и k-значные алгебры Поста (при $k \geqslant 3$), является алгеброй континуального типа, в отличие от алгебры логики, которая имеет счетное множество подалгебр [134]. Однако, несмотря на трудности, связанные с континуумом подалгебр, построена поверхность алгебры Φ^0 (см. § 3.3); она содержит свыше 200 типов подалгебр — все типы подалгебр алгебры логики (43). Изучены также некоторые классы бесконечно порожденных подалгебр, окаймляющих поверхность (пограничные и предельные подалгебры [115]). Заметим, что методика построения поверхности алгебры Φ^0 основана на существовании изолированной подалгебры $C_{\tilde{x}}$ (см. § 3.5).

Рассмотрим **модифицированную алгебру Поста**, связанную с полугруппой Π_M многорегистровых операторов [131]. Как отмечалось в § 4.3 (теорема 4.10), изучение свойств полугруппы Π_M в значительной степени сводится к изучению множества Φ_M k-значных функций с выделенными группами аргументов, причем верхний индекс означает номер регистра, к которому относится соответствующая группа аргументов.

Пусть функции $f(y_1^1, y_2^1, \ldots, y_{r_1}^1; \ldots; y_1^n, y_2^n, \ldots, y_{r_n}^n)$ и $g(y_1^1, y_2^1, \ldots, y_{l_1}^1; \ldots; y_1^m, y_2^m, \ldots, y_{l_m}^m)$ принадлежат множеству Φ_M. Под композицией функций f и g будем понимать бинарную операцию $\times$, определенную равенством

$$(f \times g)(x_1^1, x_2^1, \ldots, x_{r_1+l_1-1}^1; \ldots; x_1^m, x_2^m, \ldots, x_{r_1+l_m-1}^m; x_1^{m+1}, x_2^{m+1}, \ldots, x_{r_2}^{m+1}; \ldots; x_1^{m+n-1}, x_2^{m+n-1}, \ldots, x_{r_n}^{m+n-1}) =$$
$$= f(g(x_1^1, x_2^1, \ldots, x_{l_1}^1; \ldots; x_1^m, x_2^m, \ldots, x_{l_m}^m), g(x_2^1, x_3^1, \ldots, x_{l_1+1}^1; \ldots; x_2^m, x_3^m, \ldots, x_{l_m+1}^m), \ldots, g(x_{r_1}^1, x_{r_1+1}^1, \ldots, x_{r_1+l_1-1}^1; \ldots; x_{r_1}^m, x_{r_1+1}^m, \ldots, x_{r_1+l_m-1}^m); x_1^{m+1}, x_2^{m+1}, \ldots, x_{r_2}^{m+1}; \ldots; x_1^{m+n-1}, x_2^{m+n-1}, \ldots, x_{r_n}^{m+n-1}) \quad (4.70)$$

Композиция функций (4.70) связана с композицией многорегистровых операторов (§ 4.3). Функция g подставляется (со сдвигом в нумерации переменных) вместо каждого аргумента функции f, отнесенного к первой группе. В частности, если функции f и g состоят лишь из одной группы аргументов, то композиция функций соответствует случаю однорегистровых операторов (4.48).

По аналогии с k-значными алгебрами Поста (см. § 3.1, пример 10) введем совокупность унарных операций типа ζ, τ, Δ, ∇, позволяющих отождествлять и переставлять аргументы в каждой группе, а также переставлять, соединять и добавлять (фиктивные) различные группы переменных. Схемная интерпретация унарных операций состоит в склеивании, перестановке и перенумерации регистров, а также в пересоединении и перенумерации элементов в каждом регистре. Реализация этих унарных операций проста:

кроме того, они не нарушают логической структуры схемы, «навешенной» на регистры.

Построенную алгебру обозначим $\Phi_M(k)$ и назовем **расширенной модификацией алгебры Поста**. Очевидно, алгебра Φ^0, связанная с однорегистровыми операторами, является подалгеброй алгебры $\Phi_M(2)$. Отсюда следует, что примененные выше основные свойства структуры подалгебры Φ^0 могут быть перенесены на алгебру $\Phi_M(2)$. В частности, алгебра $\Phi_M(2)$, как и алгебра Φ^0, содержит бесконечно порожденные подалгебры и множество всех подалгебр алгебры $\Phi_M(2)$ является множеством континуальной мощности.

Установим критерий функциональной полноты в алгебре $\Phi_M(k)$. Предварительно заметим, что, как и в случае алгебры Φ^0, каждой подалгебре R k-значной алгебры Поста соответствует в алгебре $\Phi_M(k)$ подалгебра R', состоящая из функций $f \in R$ с отмеченными группами аргументов. Например, подалгебре M_1 всех монотонных функций алгебры логики (см. § 3.4) соответствует подалгебра M_1', состоящая из монотонных функций с отмеченными группами аргументов. Пусть $f(x_1^1, x_2^1, \ldots, x_{r_1}^1; \ldots; x_1^n, x_2^n, \ldots, x_{r_n}^n)$ — произвольная функция алгебры $\Phi_M(k)$. **Проекцией функции** f называется функция

$$\hat{f}(x_1^1, x_1^2, \ldots, x_1^n) = f(x_1^1, \ldots, x_1^1; x_1^2, \ldots, x_1^2; \ldots, x_1^n, \ldots, x_1^n).$$

Так, если $f(x_1^1, x_1^1; x_1^2, x_2^2, x_3^2) = x_1^1 x_3^2 \vee x_1^1 x_2^1 \vee x_1^2 x_2^2$ — булева функция, принадлежащая $\Phi_M(2)$, то ее проекцией является функция

$$\hat{f}(x_1^1; x_1^2) = x_1^1 x_1^2 \vee x_1^1 \vee x_1^2 = x_1^1 \vee x_1^2.$$

На множестве всех проекций в алгебре $\Phi_M(k)$ любая операция k-значной алгебры Поста (см. § 3.1, пример 10) производна. Следовательно, в алгебре $\Phi_M(k)$ выполнима произвольная суперпозиция проекций.

Рассмотрим множество $\widehat{R}$ всех функций $f \in \Phi_M(k)$ таких, что $f \in \hat{R}$ тогда и только тогда, когда $\hat{f} \in R'$, где $R' \subset \Phi_M(k)$ соответствует некоторой подалгебре R k-значной алгебры Поста. Так, если в качестве R выбрать множество M_1 всех монотонных булевых функций, то множество $\widehat{M}_1$ составят все такие функции из $\Phi_M(2)$, проекции которых принадлежат подалгебре M_1'. Заметим, что при этом функции из $\widehat{M}_1$ не обязательно будут монотонными. Например, функция $f(x_1^1, x_2^1; x_1^2) = \bar{x}_1^1 x_2^1 \vee x_1^2$ немонотонна, так как $f(0, 1; 0) = 1$, а $f(1, 1; 0) = 0$; в то же время $f \in \widehat{M}_1$, поскольку проекция $\hat{f}(x_1^1; x_1^2) = x_1^2 \in M_1'$.

Лемма 4.8. *Множество* $\widehat{R}$ *образует подалгебру алгебры* $\Phi_M(k)$.

Действительно, множество $\widehat{R}$ замкнуто относительно всевозможных отождествлений, перестановок и добавлений (фиктивных) групп аргументов, а также соответствующих преобразований в каждой из групп, так как все эти операции, очевидно, не выводят из множества проекций R'. В то же время любая композиция h функций из $\widehat{R}$ имеет проекцию $\hat{h}$, которая является суперпозицией проекций из R', так что $\hat{h} \in R'$, и поэтому $h \in \widehat{R}$. Лемма доказана.

Пусть R — подалгебра k-значной алгебры Поста с базисом и $\{R_q \mid q \in I\}$ — множество всех максимальных подалгебр относительно R. Тогда в силу теоремы 3.9 (следствие 3) для подалгебры R справедлив критерий Поста. Рассмотрим множество $z_q = \widehat{R}_q \cap R'$ при любом $q \in I$. Согласно доказанной лемме множества $z_q \subset R'$ являются подалгебрами алгебры $\Phi_M(k)$.

Теорема 4.14. *Система функций* $\Sigma \subset R'$ *порождает подалгебру* R' *алгебры* $\Phi_M(k)$ *тогда и только тогда, когда для каждой из подалгебр* $z_q (q \in I)$ *в системе* Σ *найдется по крайней мере одна функция, не принадлежащая данной подалгебре.*

Доказательство. *Необходимость* очевидна. Для доказательства *достаточности* выберем подсистему $\Sigma_0 \subseteq \Sigma$ такую, что

$$\Sigma_0 = \{f_q(x_1^1, x_2^1, \ldots, x_{r_1}^1; \ldots; x_1^n, x_2^n, \ldots, x_{r_n}^n) \in z_q \mid \text{при любом } q \in I\}. \quad (4.71)$$

Для каждой из функций $f_q \in \Sigma_0$, отождествив аргументы в отмеченных группах, построим проекцию $\hat{f}_q$. Причем в силу (4.71) $\hat{f}_q \in R'_q$ (при любом $q \in I$). Пусть $f(x_1^1, x_2^1, \ldots, x_{t_1}^1; \ldots; x_1^m, x_2^m, \ldots \ldots, x_{t_m}^m)$ — произвольная функция, принадлежащая подалгебре R' алгебры $\Phi_M(k)$. Проекции $\{\hat{f}_q \mid q \in I\}$ порождают любую проекцию в подалгебре R', а значит, и функцию

$$f(x_1^1; x_1^2; \ldots; x_1^{t_1 + \cdots + t_m}). \quad (4.72)$$

Разбив (4.72) с помощью унарных операций алгебры $\Phi_M(k)$ на соответствующие группы аргументов, получим

$$f(x_1^1, x_2^1, \ldots, x_{t_1}^1; \ldots; x_1^m, x_2^m, \ldots, x_{t_m}^m).$$

Теорема доказана.

Следствие 1. Система функций $\Sigma \subset \Phi_M(2)$ порождает алгебру $\Phi_M(2)$ тогда и только тогда, когда для каждой из подалгебр $\widehat{L}_1$, $\widehat{M}_1$, $\widehat{D}_3$, $\widehat{C}_2$, $\widehat{C}_3 \subset \Phi_M(2)$ (L_1 — множество всех линейных функций, M_1 — все монотонные функции, D_3 — все самодвойственные функции, $C_2(C_3)$ — все функции, сохраняющие константу 1 (0)) в системе Σ найдется по крайней мере одна функция, не принадлежащая данной подалгебре.

Следствие справедливо в силу теоремы 3.13 (см. § 3.4).

Следствие 2. Если система функций $\{f(y_1, \ldots, y_{n_i}) \mid i = 1, 2, \ldots, m\}$ полна в k-значной алгебре Поста, то система $\{f_i(x_1^1; x_1^2; \ldots; x_1^{n_i}) \mid i = 1, 2, \ldots, m\}$ порождает алгебру $\Phi_M(k)$.

Полученные результаты могут быть использованы при анализе и синтезе линейных программ, реализующих многорегистровые периодически определенные операторы.

§ 4.5. Реализация регулярных схем адресных программ в однородных структурах

В данном параграфе намечается подход к использованию аппарата алгоритмических алгебр для описания средств языков программирования. С этой целью выбирается подходящая система образующих (элементарные операторы и условия), в терминах которой удобно описывать операторные и логические средства конкретного языка. Такое погружение языка программирования в систему алгоритмических алгебр позволяет представлять его программы в виде регулярных схем, и, следовательно, используя технику эквивалентных преобразований (см. § 4.1, 4.2), можно оптимизировать исходные программы по тем или иным критериям. Примером может служить погружение адресного языка, являющегося процедурно ориентированным языком программирования. Этот язык широко применяется для решения целого ряда задач, и в частности для описания алгоритмов трансляции [128] (см. также § 6.1).

В системе алгоритмических алгебр операторные и логические средства выбранного языка описываются в терминах многорегистровых операторов (см. § 4.3, 4.4), которые являются однородными структурами. Известны также работы по проектированию схем ЭВМ, связанных с применением в качестве элементной базы вычислительных сред (ВС) с однородной структурой [43, 69, 81]. Поэтому представляет интерес разработка методов реализации многорегистровых операторов в ВС. В частности, с их помощью можно перейти к проблеме настройки ВС на режим трансляции с заданного языка программирования на язык операционного устройства, реализованного в этой среде.

Пусть M — информационное множество над совокупностью бесконечных в обе стороны абстрактных регистров $\{X^s \mid s = 1, 2, \ldots\}$ (см. § 4.3). При решении конкретных задач на каждом регистре может быть выделен конечный участок (активная зона), который

используется в данном алгоритмическом процессе. Иными словами, под абстрактным регистром понимается как регистр некоторого операционного устройства, так и ячейка (или последовательность ячеек) оперативной памяти. Поэтому можно говорить об адресе или имени регистра, его содержимом и пр.

Рассмотрим систему алгоритмических алгебр, порождаемую следующим базисом.

Базисные операторы.

1. **Операторы ввода и вывода;** указываются номера регистров, участвующих в операциях ввода и вывода.

2. **Оператор штрих-операции** (выборки содержимого регистра по адресу регистра) задает отображение $A \to M$, где A — множество адресов, M — множество содержимых этих адресов; суперпозиция (многократное применение) штрих-операции, когда содержимым адреса является другой адрес, приводит к использованию адресов высших рангов.

3. **Поразрядные операторы** инверсии I^s и конъюнкции $\wedge^s_{u,v}$ (см. § 4.3); поразрядная конъюнкция применяется к содержимым регистров X^u, X^v с хранением результата в регистре X^s.

4. **Оператор сдвига** C^s_k (k — коэффициент сдвига) (см. § 4.3).

5. **Оператор останова «!»** (прекращение алгоритмического процесса).

Предполагается, что к порождающим функциям многорегистровых операторов применимы все операции алгебры $\Phi^0_{E_k}$ (см. § 4.4).

Базисные условия.

1. $\alpha^s = 0$ — условие равенства нулю содержимого регистра X^s.

2. $\alpha^s > 0$ — условие равенства нулю знакового разряда регистра X^s.

Предполагается, что при переработке числовой информации используются спаренные регистры: бесконечный регистр и однобитовый регистр знака.

В систему образующих операторной алгебры вводятся также следующие операторы, порожденные базисом.

1. **Тождественный оператор;** $\bigoplus^s_{u,v}$ **— поразрядная сумма по mod 2;** $\bigvee^s_{u,v}$ **— поразрядная дизъюнкция** (верхний индекс указывает номер результирующего регистра, нижние — номера регистров, к которым применяется поразрядная операция; если верхнего индекса нет, результирующим регистром является X^v).

В алгебре многорегистровых операторов набор базисных операторов (инверсия и поразрядная конъюнкция) полный (теорема 4.14, следствие 2). Однако при необходимости базисные операторы можно дополнить поразрядными операторами эквивалентности, импликации и др. Для осуществления преобразований в k-ичной системе счисления в базисе необходимо заменить операторы инверсии и поразрядной конъюнкции их обобщениями на k-значный случай $\tilde{I}^s$, $(\min)^s_{u,v}$, которые задаются порождающими функциями $\tilde{I}(q) = k-1-q$, $\min(q_1, q_2)$ $(q, q_1, q_2 \in E_k = \{0, 1, \ldots, k-1\})$. Для реализации арифметических операторов вводятся также операторы $(+ \bmod k)^s_{u,v}$, $(\max)^s_{u,v}$ и $\square^s_{u,v}$ с порождающей функцией

$$q_1 \square q_2 = \begin{cases} 0 & \text{при } q_1 \geqslant q_2, \\ 1 & \text{при } q_1 < q_2. \end{cases}$$

Тогда регулярное представление оператора сложения в k-ичной системе счисления имеет вид

$$(+)_{u,v} = \underset{(\alpha^u=0)}{\{}\ [(+\bmod k)_{u,v} \cup I^u \square^u_{u,v}]\, C^u_{-1}\}.$$

Таким образом, приведенное регулярное представление оператора сложения в k-ичной системе счисления является итерацией по условию равенства нулю содержимого регистра X^u. Пусть $\tilde{\alpha}^u = (\ldots, \alpha^u_i, \ldots)$, $\tilde{\beta}^v = (\ldots, \beta^v_i, \ldots)$ — содержимые регистров соответственно X^u и X^v $(-\infty < i < +\infty)$. Тогда на каждом шаге итерации параллельно выполняются поразрядное сложение (по $\bmod k$) с хранением результата на регистре X^v и формирование переноса на регистре X^u с последующим сдвигом влево. Для формирования переноса на регистре X^u вначале производится инверсия, а затем поразрядно применяется операция $\square^u_{u,v}$, переводящая i-й разряд регистра X^u в нуль, если $k-1-\alpha^u_i \geqslant \beta^v_i$, и в единицу в противном случае.

Данное регулярное представление является обобщением на k-значный случай микропрограммы сложения целых чисел в двоичной системе счисления (см. § 4.3). Другие арифметические операторы в k-ичной системе счисления реализуются с помощью сложения, инверсий и сдвигов. Допускается применение групповых операций сложения и умножения. Заметим, что перечисленные операторы могут выполняться над регистрами, адреса которых являются содержимым указанных в них регистров. Используя арифметические операторы, условия, входящие в базис, и операцию левого умножения условия на оператор, можно получить отношения $<$, $\leqslant$, $=$, $\neq$, $\geqslant$, $>$.

2. **Операторы типа засылки,** осуществляющие засылку содержимого одного регистра в другой. Если содержимым регистров являются адреса, то засылка носит сложный характер (предварительно выполняются все штрих-операции). К этой группе операторов относятся также очистка (обращение в нуль) всего регистра или его части, установление регистра (или его части) в состояние «1» и др.

3. **Операторы перехода по содержимому** регистра, которое является именем другого оператора, **перехода по ключу**, когда осуществляется упорядоченная проверка группы условий для определения последующего оператора. Операторы подобного рода могут быть легко реализованы с помощью α-дизъюнкции.

4. **Оператор выделения**, фиксирующий конечный участок на данном регистре.

5. **Оператор пересылки**, осуществляющий пересылку конечных участков с одного регистра на другой.

6. **Оператор пропуска скобок**, использующийся при анализе синтаксической структуры выражения, помещенного в регистр для пропуска некоторых подвыражений в указанном направлении просмотра.

7. **Оператор частичного сдвига**, осуществляющий сдвиг влево (вправо) на k символов части регистра, начиная с r-го разряда.

В работе [59] приводятся регулярные представления в терминах базисных операторов и условий для всех операторов, входящих в систему образующих алгоритмических алгебр, а также регулярные схемы некоторых алгоритмов, иллюстрирующие возможности применения алгоритмических алгебр. Основные из приведенных выше операторов были запрограммированы по составленным для них регулярным схемам.

Перейдем к реализации многорегистровых операторов в ВС.

Несколько обобщая подход, намеченный в работе [56], рассмотрим абстрактную модель ВС, состоящую из двух двумерных параллельных плат, площадь каждой из которых не ограничена ($\infty \times \infty$). Нижнюю плату A^0 назовем **информационной**, а верхнюю A^1 — **вспомогательной**. Для ориентации в выбранной модели ВС введем координатные оси: горизонтальную v и вертикальную u. Координаты элемента ВС, находящегося на пересечении координатных осей (начало координат): v, u (0, 0). Указанные платы A^r, $r = \{0, 1\}$, состоят из элементов $a^r_{v,u}$, каждый из которых соединен с пятью соседними элементами. Два элемента $a^{r'}_{v_1,u_1}$ и $a^{r''}_{v_2,u_2}$ считаются **соседними**, если

1) $v_1 = v_2 \pm 1$, $u_1 = u_2$, $r' = r''$,
2) $v_1 = v_2$, $u_1 = u_2 \pm 1$, $r' = r''$,
3) $v_1 = v_2$, $u_1 = u_2$, $r' = r'' \oplus 1$.

Предполагается, что базис элемента описанной модели ВС функционально и соединительно полный. Следовательно, далее каждая плата рассматривается как решетка, в узлах которой расположены элементы ВС. Направление передачи информации между этими элементами указывается стрелками. При этом предполагается также, что выполнена необходимая настройка элементов ВС.

Пусть задан многорегистровый оператор $F^s_{k_1, k_2, \ldots, k_n}: f^{r_1, r_2, \ldots, r_n}$ (кратко F^s) над регистрами $X^1, X^2, \ldots, X^n$ и требуется настроить

предложенную модель ВС на схему, реализующую этот оператор (s — номер регистра, в который засылается результат оператора F^s), причем предполагается, что $s \in \{1, 2, \ldots, n\}$; $f(\tau^1_1, \tau^1_2, \ldots, \tau^1_{r_1}; \ldots; \tau^n_1, \tau^n_2, \ldots, \tau^n_{r_n})$ — порождающая функция данного оператора. Решение этой задачи состоит в выполнении следующих основных этапов: реализации регистров, трассировки необходимых связей, реализации порождающей функции $f^{r_1, \ldots, r_n}$ (кратко f), задающей оператор F^s.

Представим оператор F^s в виде композиции операторов: $F^s = C^1_{k_1} \times \cdots \times C^n_{k_n} \times F^s_f$. Это представление оператора F^s означает, что при его реализации следует осуществить предварительно на регистрах $X^1, X^2, \ldots, X^n$ все необходимые сдвиги, а затем выполнить оператор F^s_f. Очевидно, что с помощью сдвига $C^j_{+1}(C^j_{-1})$ в результате композиций можно получить сдвиг с любым положительным (отрицательным) коэффициентом. Следовательно, выполнение оператора сдвига $C^j_{k_j}$ сводится к реализации оператора C^j_{+1} при $k_j > 0$ или оператора C^j_{-1} при $k_j < 0$. Для выполнения этих операторов в ВС (построенной из элементов потенциального типа) выбирается два регистра: основной X^j и вспомогательный X^j_1. Элементы ВС, расположенные между этими регистрами, настраиваются на схему, реализующую оператор сдвига C^j_{+1} или C^j_{-1}. При этом необходимо, чтобы в ЭВМ (далее называемой устройством управления) вырабатывались основные управляющие сигналы: $У_п$ ($У_л$) и $У_р$. Процедуру выполнения сдвига на регистре X^j можно описать следующим образом: синхронно с управляющим сигналом $У_п$ ($У_л$) содержимое регистра X^j со сдвигом на один разряд вправо (влево) пересылается на регистр X^j_1, а затем синхронно с управляющим сигналом $У_р$ содержимое регистра X^j_1 пересылается на регистр X^j. После выполнения описанной процедуры k_j раз регистр X^j будет переведен в новое состояние, соответствующее результату выполнения оператора $C^j_{k_j}$.

Заметим, что если $k_j = 0$ и результат выполнения оператора F^s определен на регистре X^j (т. е. $j = s$), то и в этом случае (для выполнения условия устойчивости функционирования проектируемой схемы) необходимо реализовать вспомогательный регистр X^j_1. Вначале значение функции f (предположим, что синхронно с управляющим сигналом У) засылается в регистр X^j_1, а затем пересылается с управляющим сигналом $У_р$ на регистр X^j. Следовательно, элементы ВС, расположенные между этими регистрами, настраиваются на схему, реализующую оператор пересылки.

Разобьем условно ВС по обе стороны оси u на равные вертикальные слои $S^r_i (-\infty < i < \infty)$, состоящие из двух полос $B^r_i, r =$

$= \{0, 1\}$. При этом предполагается, что левая грань полосы B_0^r проходит по оси u ($v = 0$). Заметим, что оптимальный выбор ширины полосы B_i^r зависит от параметров n, $r_{\max}$ ($r_{\max} = \max(r_1, r_2, \ldots, r_n)$) оператора F^s и от формы представления порождающей функции f. Обозначим ширину каждой полосы B_i^r через b и предположим, что эта ширина достаточна для размещения схемы, реализующей i-й разряд данного оператора (F^s).

Далее, выделим горизонтальный слой S' шириной R. При пересечении слоев S_i' со слоем S' образуются параллелепипеды, верхними и нижними гранями которых являются прямоугольники соответственно $\tilde{R}_i^1$ и $\tilde{R}_i^0$. Ширина прямоугольника $\tilde{R}_i^r$ (полосы S') зависит от числа различных аргументов функции f. Обозначим вершины прямоугольников $\tilde{R}_i^1$ и $\tilde{R}_i^0$ соответственно через e_1^1, e_2^1, e_3^1, e_4^1 и e_1, e_2, e_3, e_4. Пусть v_{e_1}, u_{e_1} (v_{e_2}, u_{e_2}) — координаты левой верхней (нижней) вершины прямоугольника $\tilde{R}_i^r$ (рис. 21).

1. **Реализация регистров** (предполагается, что описанный ниже метод размещения в ВС схемы, реализующей i-й разряд оператора F^s, применяется для построения схемы в $(i+1)$-м слое).

Регистры X^j и X_1^j ($j = 1, 2, \ldots, n$) реализуются на плате A^0 по прямым, уравнения которых имеют вид $u = u^j = \text{const}$ и $u = u_1^j = \text{const}$. Причем каждой полосе $B_{i'}^0$ ставятся в соответствие элементы X_i^j и X_1^j регистров X^j и X_1^j. Расположение регистров X^1, X_1^1, X^2, X_1^2, $\ldots$, X^{n_1}, $X_1^{n_1}$ на плате A^0 следующее: выше прямой $u = u_{l_1} = \text{const}$ — регистры X^{j_1} и $X_1^{j_1}$; при этом предполагается, что $u^{j_1} > u_1^{j_1}$, а $u^{j_1} < u^{j_1+1}$, где $j_1 = 1, 2, \ldots, n_1$ $\left(n_1 = \frac{n}{2}\right.$, если n — четное число, в противном случае $\left.n_1 = \frac{n+1}{2}\right)$; ниже прямой $u = u_{l_2} = \text{const}$ — регистры $X^{n_1+1}, X_1^{n_1+1}, \ldots, X^{n_1+n_2}, X_1^{n_1+n_2}$ ($n_2 = n - n_1$); при этом предполагается, что $u_1^{j_2} > u^{j_2}$, а $u^{j_2+1} < u^{j_2}$ ($j_2 = n_1 + 1, n_1 + 2, \ldots, n_1 + n_2$).

В качестве примера на рис. 21 приведена структурная интерпретация схемы, реализующей оператор F^s при $n = 4$, $r_{\max} = 2$ ($k_1 = k_2 = k_3 = k_4 = 0$). Предполагается, что результат выполнения данного оператора определен на регистре X^2, так что элементы ВС, расположенные между регистрами X^2 и X_1^2, настроены на схему, реализующую оператор пересылки.

2. **Трассировка необходимых связей.** Прямоугольник $\tilde{R}_i^r$ разбивается на два (правый и левый) прямоугольника. В прямоугольнике $\tilde{R}_i^1$ (на рис. 21 вершины прямоугольника $\tilde{R}_i^1$ обозначены через e_1', e_5', e_6', e_2', а прямоугольника $\tilde{R}_i^0$ — через e_1, e_5, e_6, e_2) осуществляется трассировка каналов (см. контурные линии), по которым

поступают сигналы, соответствующие значениям аргументов τ_ν^j ($j = 1, 2, \ldots, n$; а при $j = \text{const}$ $\nu = 0, 1, 2, \ldots, r_j$) функции f. Например, по каналам конфигураций $2 \to 2 \to 2$ в прямоугольник $\tilde{R}_i^1$ поступают сигналы, соответствующие значениям переменных τ_0^1,

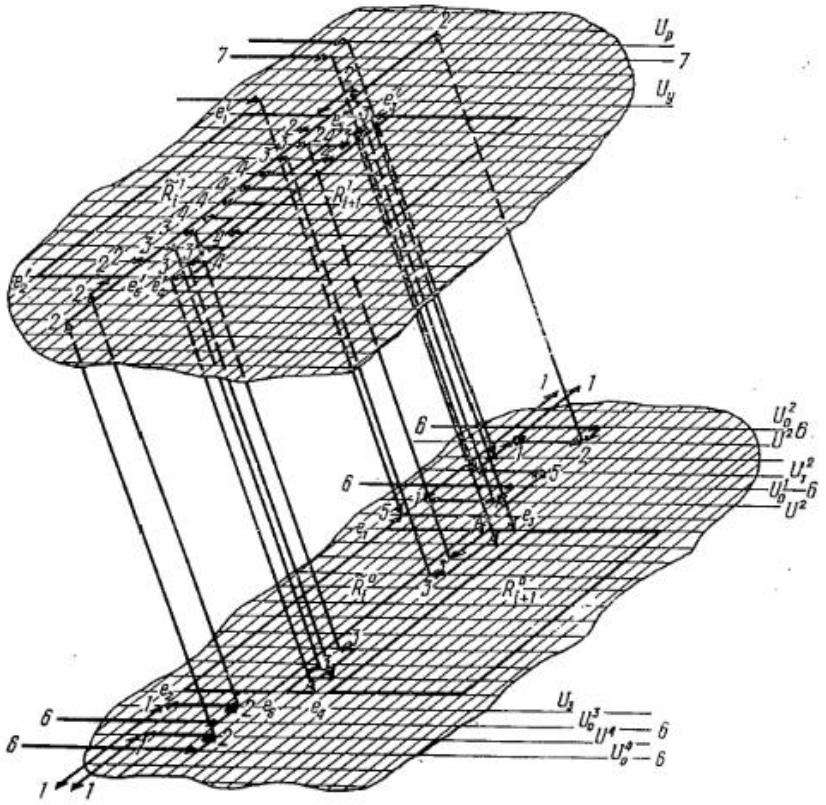


Рис. 21.

$\tau_0^2, \ldots, \tau_0^n$, т. е. единичным состояниям элементов $x_i^1, x_i^2, \ldots, x_i^n$, и снимающиеся с выходных полюсов элементов ВС, на которых реализованы элементы $x_i^1, x_i^2, \ldots, x_i^n$. По каналам конфигураций $3 \to 3 \to 3$ и $4 \to 4 \to 4$ из прямоугольника $\tilde{R}_{i+1}^1$ в прямоугольник $\tilde{R}_i^1$ поступают сигналы, соответствующие значениям переменных τ_ν^i (при $j = \text{const}$ $\nu_1 = 1, 2, \ldots, r_j$) функции f. Эти сигналы снимаются с выходных полюсов элементов ВС, расположенных на левой грани прямоугольника $\tilde{R}_{i+1}^1$.

3. **Реализация порождающей функции** f, задающей оператор F^s. В прямоугольнике $\widehat{R}_i^r$ строится схема, реализующая функцию f в соответствии с ее формой представления. Значение этой функции снимается с выходных полюсов элементов ВС, расположенных по прямым $u = u_{l_1} + 1 = \text{const}$ или $u = u_{l_2} - 1 = \text{const}$ соответственно, в зависимости от того, над или под прямоугольником R_i^r расположен регистр X^s. По каналу конфигурации $5 \to 5$ на вход элемента x_i регистра X_1^r поступает сигнал, соответствующий значению функции f.

На рис. 21 указаны также каналы $1 \to 1$, по которым осуществляется прием исходной информации в регистры $X^1, X^2, \ldots, X^n$ и выдача их содержимого в машину. Так как предполагается, что вводимая в регистр $X^j (X_1^j)$ информация поступает на нулевой вход его элементов, то необходимо, чтобы этот регистр предварительно был установлен в нулевое состояние, например управляющим сигналом $У_0^j (У_{0,1}^j)$, вырабатываемым в машине. Канал, по которому проходит сигнал $У_0^j (У_{0,1}^j)$, трассируется на плате $A^0 (A^1)$ (на рис. 21 обозначен цифрами $6 \to 6$ $(7 \to 7)$).

Таким образом, схему, реализующую оператор F^s, будем считать построенной в данной модели ВС.

Заметим, что реализация многорегистровых периодически определенных преобразований с вспомогательными переменными (см. § 4.3) может быть сведена к реализации многорегистровых операторов. Значения вспомогательных переменных $\gamma^1, \gamma^2, \ldots, \gamma^m$ вырабатываются в момент выполнения операций в качестве сигналов в некоторых точках соответствующих схем, и поэтому для их хранения не требуются специальные регистры. Обозначим такого рода операторы через $F^s(m)$ (s — номер регистра, в который засылается результат рассматриваемого преобразования). Тогда оператор $F^s(m)$ может быть реализован следующими многорегистровыми операторами:

$$F^s(m) = F^{(0)}_{k_1^0, k_2^0, \ldots, k_s^0, \ldots, k_n^0; f_0} \cup F^{(1)}_{k_1^1, k_2^1, \ldots, k_s^1, \ldots, k_n^1; f_1} \cup \cdots$$
$$\cdots \cup F^{(m)}_{k_1^m, k_2^m, \ldots, k_s^m, \ldots, k_n^m; f_m}, \qquad (4.73)$$

где $\cup$ — операция параллельного выполнения многорегистровых операторов $F^l_{k_1^l, k_2^l, \ldots, k_s^l; f_l} (l = 0, 1, 2, \ldots, m)$ с коэффициентами k_j^l, соответствующими j-м регистрам $(j = 1, 2, \ldots, n)$, и порождающими функциями $f_l(\Gamma_1^l, \Gamma_2^l, \ldots, \Gamma_{s-1}^l, \Gamma_s^l, \Gamma_{s+1}^l, \ldots, \Gamma_n^l)$, $\Gamma_j^l = (\tau_1^j, \tau_2^j, \ldots, \tau_{r_j^l}^j)$ — группа переменных, относящихся к j-му основному регистру; если $j = s$, то $\Gamma_s^l = (\tau_1^s, \tau_2^s, \ldots, \tau_{r_s^l}^s; \gamma_1^1, \gamma_2^1, \ldots$

$\ldots, \gamma_{r_s^1}^1; \ldots; \gamma_1^m, \gamma_2^m, \ldots, \gamma_{r_s^m}^m)$ — группа переменных s-го регистра и значения вспомогательных переменных. Преобразования на элементах регистров X^s задается операторами $F^{(0)}_{k_1^{(0)}, k_2^{(0)}, \ldots, k_s^{(0)}, \ldots, k_n^{(0)}; f_0}$ (кратко $F^{(0)}$) с помощью операторов $F^{(1)}, \ldots, F^{(m)}$; соответственно определяются значения вспомогательных переменных $\gamma^1, \gamma^2, \ldots, \gamma^m$.

Многорегистровые операторы с вспомогательными переменными обычно определяются на конечных (или бесконечных в одну сторону) регистрах. Однако для обеспечения возможности и однозначности определения результата выполнения оператора $F^s(m)$ необходимо, во-первых, в некоторых начальных разрядах заранее задать значения вспомогательных переменных и, во-вторых, выполнить условия согласования (см. § 4.3). Если оператор $F^s(m)$ представлен выражением (4.73), эти согласования можно выразить следующим образом: при выполнении вычислений справа налево необходимо, чтобы $k_s^l > 0$, а слева направо — $k_s^l < 0$ и $|k_s^l| - r_s^l > 0$.

К микрооперациям, описываемым с помощью многорегистровых операторов в виде $F^s(m)$, относится, например, микрооперация сложения двух положительных целых чисел (см. § 4.3).

В заключение заметим, что так как оператор $F^s(m)$ может быть задан с помощью многорегистровых операторов, то реализующую его схему можно построить по предложенному выше алгоритму реализации.

Концепция периодически определенного преобразования допускает дальнейшее обобщение его на случай n-мерных абстрактных регистров [158, 37]. При этом результаты, полученные в теории модифицированных алгебр Поста (§ 4.3, 4.4), оказываются справедливыми для периодически определенных преобразований на n-мерных регистрах и близких к ним однородных автоматных структурах (сотообразные структуры, итеративные сети, однородные среды и др.). Аппарат периодически определенных преобразований применяется при решении задач проектирования логических структур ЭВМ, связанных с теорией структур данных и организацией синхронных параллельных процессов для высокопроизводительных средств вычислительной техники [32, 33, 145].

ГЛАВА 5 ФОРМАЛЬНЫЕ ЯЗЫКИ И ГРАММАТИКИ

§ 5.1. Представление языков с помощью грамматик

Любой **формальный язык** представляет собой множество цепочек в некотором конечном алфавите[1]. К формальным языкам относятся, в частности, **искусственные** языки для общения человека с машиной — языки программирования (см. гл. 6). Для задания описания формального языка необходимо, во-первых, указать **алфавит**, т. е. совокупность объектов, называемых символами (или буквами), каждый из которых можно воспроизводить в неограниченном количестве экземпляров (подобно обычным печатным буквам или цифрам), и, во-вторых, задать **формальную грамматику** языка, т. е. перечислить правила, по которым из символов строятся их последовательности, принадлежащие определяемому языку, — **правильные цепочки.**

Заметим, что каждый символ алфавита рассматривается как неделимый в том смысле, что при построении цепочек никогда не используются его графические элементы (части символа) и всякая последовательность символов однозначно представляет некоторую цепочку. Практически это требование достигается, например, путем установления **пробела** (промежутка стандартной длины) между символами, который превышает длину любого из промежутков, встречающихся внутри символов алфавита.

Правила формальной грамматики можно рассматривать как **продукции** (правила вывода) — **элементарные операции**, которые, будучи применены в определенной последовательности к исходной цепочке (**аксиоме**), порождают лишь правильные цепочки. Сама последовательность правил, использованных в процессе порождения некоторой цепочки, является ее выводом. Определенный таким образом язык представляет собой **формальную систему.** Известными примерами формальных систем служат **логические исчисления** (исчисление высказываний, исчисление предикатов), которые подробно изучаются в соответствующих разделах математической логики [54, 60, 86].

[1] В лингвистике вместо термина «алфавит» употребляется термин «словарь», так как элементы, из которых он составлен, представляют собой слова, или, точнее, словоформы. В то же время цепочка над словарем рассматривается как словосочетание или предложение [22].

По способу задания правильных цепочек формальные грамматики разделяются на **порождающие и распознающие.** К порождающим относятся грамматики, по которым можно построить любую правильную цепочку с указанием ее структуры и нельзя построить ни одной неправильной цепочки. Распознающая грамматика — это грамматика, позволяющая установить, правильна ли произвольно выбранная цепочка и, если она правильна, выяснить ее строение.

Формальные грамматики широко применяются в лингвистике и программировании в связи с изучением естественных языков и языков программирования. В настоящей главе излагаются основные сведения о порождающих грамматиках. Автоматные структуры над внутренней памятью, ориентированные на процесс распознавания, рассматриваются в гл. 6.

Понятие порождающей грамматики впервые было предложено Хомским [109, 110].

Порождающей грамматикой или, кратко, грамматикой называется упорядоченная четверка $G = (\mathfrak{A}, V_н, \sigma, P)$, где $\mathfrak{A} = \{a_1, a_2, \ldots, a_m\}$ — **основной терминальный** алфавит; $V_н$ — **конечный вспомогательный (нетерминальный)** алфавит, символы которого обозначаются строчными греческими буквами; $\sigma \in V_н$ — **начальный (нетерминальный) символ;** $P = \{u_i \to v_i \mid i = 1, 2, \ldots, k\}$ — конечная система подстановок, левые и правые части которых суть цепочки $u_i, v_i \in F(v)$, где $F(v)$ — свободная полугруппа над **объединенным алфавитом** $V = (\mathfrak{A} \cup V_н)$; символ $\to$ внешний и не принадлежит к алфавиту V.

Иными словами, символы основного алфавита $\mathfrak{A}$ являются элементарными единицами определяемого языка; символы алфавита $V_н$ — **метапеременные,** использующиеся при выводе правильных цепочек (в естественных языках такими метапеременными являются грамматические классы: существительное, глагол и др.); σ — **метапеременная аксиома,** из которой выводятся все правильные цепочки (в естественных языках аксиоме соответствует грамматический класс «предложение»); P — **схема грамматики,** состоящая из продукций (правила вывода — грамматические правила определяемого языка). Например, порождающей грамматикой является грамматика $G_0 = (\{a, b, c\}, \{\sigma, \tau\}, \sigma, P_0)$, имеющая систему правил

$$P_0 = \begin{cases} \sigma \to a\sigma a, \\ \sigma \to c, \\ \sigma \to \tau\tau, \\ a\tau a \to b. \end{cases}$$

Определение языка $L(G)$, заданного порождающей грамматикой G, связано с **понятием выводимости.**

Пусть x, y — цепочки, принадлежащие свободной полугруппе $F(V)$. Цепочка y **непосредственно выводима** из цепочки x в грамматике G, $x \underset{G}{\Rightarrow} y$ (или просто $x \Rightarrow y$, когда G подразумевается), если в схеме P данной грамматики найдется продукция $u \to v$ такая, что $x = x_1 u x_2$, $y = x_1 v x_2$, где $x_1, x_2 \in F(V)$. Иными словами, цепочка y получается в результате применения к цепочке x продукции $u \to v \in P$, т. е. замены в цепочке x выделенного вхождения левой части u данной продукции ее правой частью v. Например, в грамматике G_0 $b\sigma c \Rightarrow b\tau\tau c$, $acaba \Rightarrow bba$ и т. д.

Цепочка y называется **выводимой** из цепочки x в грамматике G, $x \overset{*}{\underset{G}{\Rightarrow}} y$ (или просто $x \overset{*}{\Rightarrow} y$, когда G подразумевается), если цепочки x, y совпадают или существует последовательность цепочек $z_0, z_1, \ldots, z_k$ такая, что $z_0 = x$, $z_k = y$ и при любом i $(1 \leqslant i \leqslant k)$ $z_{i-1} \Rightarrow z_i$. Последовательность цепочек $W = (z_0, z_1, \ldots, z_k)$ называется **выводом** цепочки y из цепочки x в грамматике G. Например, в грамматике G_0 $\sigma \overset{*}{\Rightarrow} aba$, причем последовательность $(\sigma, a\sigma a, aa\sigma aa, aacaa, aba)$ является выводом цепочки aba из цепочки σ.

Заметим, что на каждом шаге вывода можно выбрать любую из продукций, применимых в данный момент. Это означает: порядок применения продукций в грамматике произволен и любую продукцию разрешается применять после какой угодно. Необходимо лишь, чтобы данная продукция была применимой на очередном шаге вывода. Этим понятие порождающей грамматики существенно отличается от понятия **нормального алгоритма,** в котором подстановки носят характер предписаний и строго выполняются в заранее указанной последовательности [80].

Вывод $x \overset{*}{\underset{G}{\Rightarrow}} y$ называется **полным,** если $y \in F(\mathfrak{A})$ (т. е. цепочка y ссстоит из терминальных символов). Любой полный вывод завершается применением продукций, правые части которых представляют собой терминальные цепочки. Такие продукции в дальнейшем будем называть **заключительными продукциями** данной грамматики.

Если $x \overset{*}{\underset{G}{\Rightarrow}} y$ и $y \overline{\in} F(\mathfrak{A})$, причем в системе P не существует правил, применимых к цепочке y, то вывод цепочки y из цепочки x в грамматике G называется **тупиковым.** Например, приведенный выше вывод цепочки aba из цепочки σ в грамматике G_0 — полный, продукции $\sigma \to c$ и $aca \to b$ — заключительные продукции грамматики G_0, $(\sigma, a\sigma a, aa\sigma aa, aa\tau\tau aa)$ — тупиковый вывод цепочки $aa\tau\tau aa$ из цепочки σ в грамматике G_0.

Рассмотрим теперь способ, которым порождающая грамматика $G = (\mathfrak{A}, V_н, \sigma, P)$ определяет соответствующий ей язык. Цепочка $x \in F(\mathfrak{A})$ называется **правильной,** если существует по крайней мере один полный вывод цепочки из аксиомы σ в грамматике G. Иными словами, цепочка x правильная, если 1) $x \in F(\mathfrak{A})$ (цепочка x состоит из терминальных символов); 2) $\sigma \overset{*}{\underset{G}{\Rightarrow}} x$ (существует вывод цепочки x из аксиомы σ). Множество всех правильных цепочек в грамматике G образует язык $L(G)$, **порожденный грамматикой** G. Например, грамматика G_0 порождает язык

$$L(G_0) = \{a^n b a^n \cup a^m c a^m \mid n, m = 0, 1, 2, \ldots\}.$$

Таким образом, каждой грамматике $G = (\mathfrak{A}, V_н, \sigma, P)$ однозначно соответствует язык $L(G)$, порожденный данной грамматикой. Однако это соответствие не взаимно-однозначно. Один и тот же язык может порождаться различными грамматиками. Это позволяет ввести отношение эквивалентности на множестве грамматик. Грамматики G и G' называются **эквивалентными** $(G \sim G')$, если $L(G) = L(G')$ (т. е. грамматики G и G' порождают один и тот же язык). Например, грамматика $G_1 = (\{a, b, c\}, \{\sigma\}, \sigma, P_1)$, схема которой имеет вид

$$P_1 = \begin{cases} \sigma \to a\sigma a, \\ \sigma \to b, \\ \sigma \to c, \end{cases}$$

порождает язык $L(G_1)$, который совпадает с языком $L(G_0)$, и, следовательно, $G_0 \sim G_1$.

В классах эквивалентных грамматик (порождающих один и тот же язык) естественно выделять **нормальные формы** — грамматики, продукции которых удовлетворяют некоторым ограничениям. В частности, справедлива такая теорема

Теорема 5.1. *Для каждой порождающей грамматики* $G = (\mathfrak{A}, V_н, \sigma, P)$ *существует такая эквивалентная ей грамматика* $G' = (\mathfrak{A}, V'_н, \sigma, P')$, *левая часть каждой продукции которой не содержит символов алфавита* $\mathfrak{A}$.

Действительно, пусть $G = (\mathfrak{A}, V_н, \sigma, P)$ — произвольная порождающая грамматика. Каждому терминальному символу $a \in \mathfrak{A}$ сопоставим дополнительный нетерминальный символ $a' \overline{\in} V_н$ так, чтобы различным символам алфавита $\mathfrak{A}$ соответствовали различные нетерминальные символы. Пусть $\mathfrak{A}' = \{a' \mid a \in \mathfrak{A}\}$ — полученное множество дополнительных нетерминальных символов. Построим грамматику $G' = (\mathfrak{A}, V'_н, \sigma, P')$, где $V'_н = V_н \cup \mathfrak{A}'$. Схема P' строится по продукциям грамматики G: в каждой продукции схемы P все

терминальные символы заменяются соответствующими нетерминальными; кроме того, в систему P' включаются продукции вида $a' \to a$ для каждого $a \in \mathfrak{A}$, которые являются заключительными продукциями грамматики G'. Из этого построения следует, что левая часть каждой продукции схемы P' не содержит символов основного алфавита $\mathfrak{A}$.

Докажем эквивалентность грамматик G и G'. Пусть $x = a_{i_1}a_{i_2} \ldots a_{i_r} \in L(G)$ и $W = (\sigma, z_1, \ldots, z_k)$, $x = z_k$, — некоторый полный вывод цепочки x в грамматике G. Тогда по данному выводу нетрудно построить полный вывод W' цепочки x в грамматике G'. Для этого достаточно на i-м шаге вывода W' использовать продукцию $P_i' \in P'$, соответствующую продукции P_i грамматики G, примененной на i-м шаге вывода W ($i = 1, 2, \ldots, k$). В результате получим вывод $W'' = (\sigma, z_1', z_2', \ldots, z_k')$, где $z_k' = a_{i_1}'a_{i_2}' \ldots a_{i_r}'$. Применив заключительные продукции грамматики G', получим полный вывод W' цепочки x в грамматике G'. Следовательно, $L(G) \subseteq (G')$.

Пусть теперь $x = a_{i_1}a_{i_2} \ldots a_{i_r} \in L(G')$ и $W' = (\sigma, z_1, \ldots, z_k)$, $x = z_k$, — некоторый полный вывод цепочки x в грамматике G'. Поскольку символы основного алфавита не входят в левые части продукций грамматики G', вывод W' может быть перестроен так, чтобы применение продукций $a' \to a$ использовалось лишь в самом конце, после применения всех остальных продукций грамматики G'. Перестроенный таким образом вывод W' может быть разбит на две части: первая, W'', не содержит применений заключительных продукций и завершается цепочкой $x' = a_{i_1}'a_{i_2}' \ldots a_{i_k}'$, а вторая состоит из последовательного применения заключительных продукций грамматики G', переводящих цепочку x' в цепочку x. Удалив все штрихи в первой части W'', получим вывод цепочки x в грамматике G так, что $L(G) \supseteq L(G')$.

Следовательно, $L(G) = L(G')$ и грамматики G и G' эквивалентны. Грамматику G' назовем **нормальной формой**, порождающей грамматику G. Такое представление порождающих грамматик используется при доказательстве некоторых теорем в § 5.2.

Пример 1. Рассмотрим порождающую грамматику с терминальным алфавитом $\mathfrak{A}_1 = \{a_1, \ldots, a_m, b\}$, схема которой имеет вид

$$\begin{cases} \sigma \to \sigma\psi_i a_i, \\ \sigma \to b, \\ a_i\psi_j \to \psi_j a_i, \\ b\psi_i \to a_i b, \end{cases}$$

где $i, j = 1, 2, \ldots, m$. По теореме 5.1 нормальная форма данной грамматики имеет вид

$$\begin{cases} \sigma \to \sigma\psi_i a_i, \\ \sigma \to \beta, \\ a_i\psi_j \to \psi_j a_i, \\ \beta\psi_i \to \alpha_i\beta, \\ \beta \to b, \\ \alpha_i \to a_i, \end{cases}$$

где $i, j = 1, 2, \ldots, m$. Приведенная грамматика порождает язык $\{xbx \mid$ для любого непустого слова $x \in F(\mathfrak{A})\}$, где $\mathfrak{A} = (a_1, \ldots, a_m)$.

Действительно, выберем произвольную цепочку

$$a_{i_1}a_{i_2} \ldots a_{i_k}ba_{i_1}a_{i_2} \ldots a_{i_k} \tag{5.1}$$

в данном языке. Тогда выводом этой цепочки является последовательность

$$\begin{aligned} &(\sigma, \sigma\psi_{i_k}a_{i_k}, \sigma\psi_{i_{k-1}}a_{i_{k-1}}\psi_{i_k}a_{i_k}, \ldots, \sigma\psi_{i_1}a_{i_1}\psi_{i_2}a_{i_2} \ldots \psi_{i_k}a_{i_k}, \\ &\beta\psi_{i_1}a_{i_1}\psi_{i_2}a_{i_2} \ldots \psi_{i_k}a_{i_k}, a_{i_1}\beta a_{i_1}\psi_{i_2}a_{i_2} \ldots \psi_{i_k}a_{i_k}, \\ &a_{i_1}\beta\psi_{i_2}a_{i_1}a_{i_2}\psi_{i_3}a_{i_3} \ldots \psi_{i_k}a_{i_k}, a_{i_1}a_{i_2}\beta a_{i_1}a_{i_2}\psi_{i_3}a_{i_3} \ldots \\ &\ldots \psi_{i_k}a_{i_k}, \ldots, a_{i_1} \ldots a_{i_k}\beta a_{i_1} \ldots a_{i_k}). \end{aligned}$$

Применив заключительные продукции $\alpha_i \to a_i$, $\beta \to b$, получим полный вывод цепочки (5.1). Можно показать, что приведенная грамматика порождает цепочки только данного языка.

Пример 2. Рассмотрим порождающую грамматику с терминальным алфавитом $\{a, b, c\}$, нетерминалами $\{\sigma, \alpha, \beta, \alpha_1, \beta_1\}$ и схемой

$$\begin{cases} p_1 : \sigma \to \alpha_1\sigma\alpha, \\ p_2 : \sigma \to \alpha_1\beta_1 c, \\ p_3 : c\alpha \to \beta c, \\ p_4 : \beta \to \alpha, \\ p_5 : \beta_1 \to b, \\ p_6 : ba \to b\beta_1 c, \\ p_7 : \alpha_1 \to a. \end{cases}$$

Эта грамматика порождает язык $\{a^n b^n c^n \mid n = 1, 2, \ldots\}$. Например, цепочка $a^2b^2c^2$ в данной грамматике имеет вывод, на каждом шаге которого последовательно применяются продукции p_1, p_2, p_3, p_4, p_5, p_6, p_5, p_7, p_7:

$$\begin{aligned} &(\sigma, \alpha_1\sigma\alpha, \alpha_1\alpha_1\beta_1 c\alpha, \alpha_1\alpha_1\beta_1\beta c, \alpha_1\alpha_1\beta_1\alpha c, \alpha_1\alpha_1 bac, \alpha_1\alpha_1 b\beta_1 cc, \\ &\alpha_1\alpha_1 bbcc, a\alpha_1 bbcc, aabbcc). \end{aligned}$$

Заменив в продукциях p_2, p_3, p_6 все вхождения символа c символом a, получим грамматику, порождающую язык: $\{a^n b^n a^n \mid n = 1, 2, \ldots\}$.

§ 5.2. Система составляющих. нс-Грамматики

Наряду с описанием языка $L(G)$, порожденного грамматикой G, важное значение имеет описание строения (синтаксической структуры) цепочек данного языка. Описание синтаксической структуры цепочки обычно тесно связано с выводом данной цепочки в грамматике G. В естественных языках для выяснения синтаксической структуры предложения проводят его грамматический анализ. В языках программирования для этой цели создаются специальные алгоритмы синтаксического анализа и контроля программ, которые используются при разработке трансляторов, при решении задач машинного перевода и других важных проблем (см. гл. 6). Описание синтаксической структуры правильных цепочек обычно связывают с понятием системы составляющих.

Пусть $x = a_{i_1}a_{i_2} \dots a_{i_k}$ — некоторая цепочка. Для подцепочки $y = a_{j_1}a_{j_2} \dots a_{j_p}$ выполняется **вхождение** (y, r) в цепочку x, если $a_{i_r} = a_{j_1}$, $a_{i_{r+1}} = a_{j_2}$, $\dots$, $a_{i_{r+p-1}} = a_{j_p}$, где $1 \leqslant r$, $r + p - 1 \leqslant k$. В частности, если y — символ (однобуквенная цепочка), говорят о вхождении символа y в цепочку x. Например, для $x = ababc$, $y = ab$ справедливы вхождения $(y, 1)$, $(y, 3)$ цепочки y в x, а также вхождения $(a, 1)$, $(b, 4)$, $(c, 5)$ символов a, b, c соответственно в цепочку x.

Пусть $y = a_{j_1}a_{j_2} \dots a_{j_p}$, $z = a_{l_1}a_{l_2} \dots a_{l_q}$ — подцепочки цепочки x, для которых выполняются вхождения (y, r) и (z, s). Вхождения (y, r) и (z, s) перекрываются, если $r \leqslant s \leqslant r + p - 1$ или $s \leqslant r \leqslant s + q - 1$. Если $r \leqslant s \leqslant r + p - 1$ и цепочка z является подцепочкой цепочки y, говорят, что вхождение (y, r) **покрывает** вхождение (z, s) (иными словами, вхождение (z, s) вложено в (y, r)). Так, если $x = ababc$, $y = ab$, $z = abc$, вхождения $(y, 3)$ и $(z, 3)$ перекрываются, причем $(y, 3)$ вложено в $(z, 3)$, а вхождения $(y, 1)$ и $(y, 3)$ не перекрываются.

Рассмотрим множество $\mathfrak{S}(x)$ вхождений всех подцепочек в цепочку x, причем $(x, 1) \in \mathfrak{S}(x)$, где $(x, 1)$ — вхождение цепочки x самой в себя. Система $C \subset \mathfrak{S}(x)$ называется **системой составляющих** цепочки x, если выполняются следующие условия: 1) C содержит вхождение $(x, 1)$, а также вхождения всех символов, из которых состоит цепочка x; 2) из любых вхождений $\alpha, \beta \in C$ либо одно вложено в другое, либо вхождения α, β не перекрываются.

Вхождения, принадлежащие системе C, называются **составляющими** цепочки x, а вхождения символов — **атомными** составляющими данной цепочки. Из условия 1 следует, что различные системы составляющих цепочки x содержат одно и то же множество ее атомных составляющих. Например, система $C = \{(a, 1), (b, 2), (a, 3), (b, 4), (c, 5), (ab, 1), (ab, 3), (ababc, 1)\}$ является системой составляющих цепочки $x = ababc$; система, содержащая все атомные составляющие цепочки x и вхождение $(x, 1)$, очевидно, является системой составляющих произвольной цепочки x.

С описанием синтаксической структуры цепочек в терминах их составляющих тесно связан важный класс порождающих грамматик — **грамматик непосредственно составляющих (нс-грамматик)**, т. е. грамматик $G = (\mathfrak{A}, V_н, \sigma, P)$, продукции которых имеют вид $u_1\psi u_2 \to u_1 z u_2$, где u_1, u_2 — произвольные цепочки из свободной полугруппы $F(v)$ над алфавитом $V = \mathfrak{A} \cup V_н$, z — непустая цепочка из этой же полугруппы, а $\psi \in V_н$ — некоторый нетерминальный символ. Цепочки u_1 и u_2 иногда называют соответственно **левым и правым контекстами** данной продукции. Язык $L(G)$, порожденный нс-грамматикой G, называется **нс-языком**.

Пример 1.

Любой конечный язык $L_1 = \{x_1, x_2, \dots, x_k\}$, где x_i $(1 \leqslant i \leqslant k)$ — непустое слово конечной длины в алфавите $\mathfrak{A} = \{a_1, a_2, \dots, a_m\}$, является нс-языком с порождающей нс-грамматикой $G_1 = (\mathfrak{A}, \{\sigma\}, \sigma, P_1)$, схема которой имеет вид

$$P_1 = \begin{cases} \sigma \to x_1, \\ \sigma \to x_2, \\ \dots \\ \sigma \to x_k. \end{cases}$$

Пример 2. Универсальный язык L_2, состоящий из всех непустых цепочек конечной длины в алфавите $\mathfrak{A} = \{a_1, a_2, \dots, a_m\}$, является нс-языком с порождающей нс-грамматикой $G_2 = (\mathfrak{A}, \{\sigma\}, \sigma, P_2)$, схема которой имеет вид

$$P_2 = \begin{cases} \sigma \to a_i\sigma, \\ \sigma \to a_i \end{cases}$$

при всех $i = 1, 2, \dots, m$.

Пример 3. Язык $L_3 = \{a^n b a^n \mid n = 0, 1, 2, \dots\}$ (предполагается, что $z^0 = e$ для любой цепочки $z \in F(v)$) является нс-языком с порождающей нс-грамматикой $G_3 = (\{a, b\}, \{\sigma\}, \sigma, P_3)$, схема которой имеет вид

$$P_3 = \begin{cases} \sigma \to a\sigma a, \\ \sigma \to b. \end{cases}$$

Пример 4. Язык $L_4 = \{a^n b^n \mid n = 1, 2, \dots\}$ является нс-языком с порождающей нс-грамматикой $G_4 = (\{a, b\}, \{\sigma\}, \sigma, P_4)$, схема которой имеет вид

$$P_4 = \begin{cases} \sigma \to a\sigma b, \\ \sigma \to ab. \end{cases}$$

Пример 5. Пусть $x = a_{i_1}a_{i_2}\ \ldots\ a_{i_k}$ — произвольное непустое слово в алфавите $\mathfrak{A} = \{a_1, a_2, \ldots, a_m\}$, $x^{-1} = a_{i_k}a_{i_{(k-1)}}\ \ldots\ a_{i_2}a_{i_1}$ — зеркальное отражение слова x. Язык $L_5 = \{xx^{-1} \mid$ для любого непустого слова $x \in F(\mathfrak{A})\}$ является нс-языком с порождающей нс-грамматикой $G_5 = (\mathfrak{A}, \{\sigma\}, \sigma, P_5)$, имеющей схему

$$P_5 = \begin{cases} \sigma \to a_i\sigma a_i, \\ \sigma \to a_i a_i \end{cases}$$

при всех $i = 1, 2, \ldots, m$.

Пример 6. Язык $L_6 = \{xbx \mid$ для любого непустого слова $x \in F(\mathfrak{A})\}$, где $\mathfrak{A} = \{a_1, a_2, \ldots, a_m\}$ (см. § 5.1, пример 1), является нс-языком с порождающей нс-грамматикой G_6, схема которой имеет вид

$$\begin{cases} p_1 : \sigma \to \sigma\psi_i a_i, \\ p_2 : \sigma \to \beta, \\ p_3 : \alpha_i\psi_j \to \delta_{ij}\psi_j, \\ p_4 : \delta_{ij}\psi_j \to \delta_{ij}a_i, \\ p_5 : \delta_{ij}\alpha_i \to \psi_j a_i, \end{cases} \qquad \begin{cases} p_6 : \beta\psi_i \to \tau_i\psi_i, \\ p_7 : \tau_i\psi_i \to \tau_i\beta, \\ p_8 : \tau_i\beta \to \alpha_i\beta, \\ p_9 : \beta \to b, \\ p_{10} : \alpha_i \to a_i \end{cases}$$

при любых $i, j = 1, 2, \ldots, m$. Построим вывод в данной грамматике цепочки $a_1a_2ba_1a_2 \in L_6$. Вначале, дважды применяя продукции вида p_1, получаем $(\sigma, \sigma\psi_2a_2, \sigma\psi_1a_1\psi_2a_2)$. Используя $p_3 - p_5$, выводим цепочку $\sigma\psi_1\psi_2a_1a_2$. Затем, используя p_2, переходим к цепочке $\beta\psi_1\psi_2a_1a_2$. Дважды применяя последовательность продукций $p_6 - p_8$, получаем цепочку $a_1a_2\beta a_1a_2$. Наконец, используя заключительные продукции p_9, p_{10}, выводим исходную цепочку $a_1a_2ba_1a_2$.

Пример 7. С помощью нс-грамматик описывается синтаксис ряда современных языков программирования, и в частности синтаксис АЛГОЛа-60. нс-Грамматика АЛГОЛ имеет около 150 металингвистических правил, записанных в так называемой **нормальной форме Бэкуса (БНФ)**, применяемой для описания многих языков программирования. Проиллюстрируем этот метод на описании множества L идентификаторов АЛГОЛа:

⟨идентификатор⟩ ::= ⟨буква⟩ | ⟨идентификатор⟩⟨буква⟩ | ⟨идентификатор⟩⟨цифра⟩.

Идентификаторы представляют собой последовательности, состоящие из букв латинского алфавита и цифр 0, 1, ..., 9 и начинающиеся с некоторой буквы. Идентификаторы используются для обозначения различных объектов языков программирования: переменных, массивов, процедур и др.

Язык L_7 является нс-языком с порождающей нс-грамматикой $G_7 = (\{A, B, C, \ldots, x, y, z, 0, 1, \ldots, 9\}, \{\alpha, \beta, \gamma\}, \alpha, P_7)$, схема которой имеет вид

$$P_7 = \begin{cases} \alpha \to \alpha\beta, \\ \alpha \to \alpha\gamma, \\ \alpha \to \beta, \\ \beta \to A, \\ \beta \to B, \\ \ldots \end{cases} \qquad \begin{cases} \beta \to Y, \\ \beta \to Z, \\ \gamma \to 0, \\ \gamma \to 1, \\ \ldots \\ \gamma \to 9. \end{cases}$$

Заметим, что класс нс-грамматик является собственным подмножеством множества всех порождающих грамматик, так как доказано, что не для всякой порождающей грамматики существует эквивалентная ей нс-грамматика (см., например, [20, 21]).

Существует естественная связь между выводом произвольной цепочки в нс-грамматике $G = (\mathfrak{A}, V_н, \sigma, P)$ и системой составляющих данной цепочки. Пусть

$$W = (\sigma, z_1, \ldots, z_{n-1}, x) \tag{5.2}$$

есть полный вывод цепочки $x \in F(\mathfrak{A})$ грамматики G. При любом $1 \leqslant i \leqslant n$ цепочки z_i, z_{i+1} могут быть представлены в виде

$$z_i = v_iu_i\xi_iu_i'v_i', \quad z_{i+1} = v_iu_iy_iu_i'v_i',$$

где $m_i : u_i\xi_iu_i' \to u_iy_iu_i'$ — продукция грамматики G, примененная на i-м шаге вывода.

Пусть, далее, $q_i = |v_iu_i|$ — длина подцепочки v_iu_i, входящей в цепочку z_i. Тогда каждому выводу (5.2) можно сопоставить последовательность $([q_1, m_1], [q_2, m_2], \ldots, [q_{n-1}, m_{n-1}])$, т. е. способ проведения вывода (5.2). Например, полному выводу

$$(\sigma, a\sigma a, a^2\sigma a^2, a^3\sigma a^3, a^3ba^3) \tag{5.3}$$

в ранее рассмотренной грамматике G_3 (см. пример 3) соответствует способ проведения вывода

$$([0, m_1], [1, m_2], [2, m_3], [3, m_4],$$

где $m_i : \sigma \to a\sigma a$ $(i = 1, 2, 3)$, $m_4 : \sigma \to b$.

Заметим, что некоторому выводу в грамматике может соответствовать несколько способов его проведения. Так, началу вывода

$$\sigma \underset{G}{\Rightarrow} a\sigma a \underset{G}{\Rightarrow} a\sigma\sigma a \tag{5.4}$$

в грамматике G, в систему правил которой входит правило $m : \sigma \to a\sigma\sigma a$, можно сопоставить два способа проведения вывода:

$$([0, m], [0, m]) \text{ и } ([0, m], [1, m]). \tag{5.5}$$

Каждому способу проведения вывода однозначно соответствует дерево вывода, которое в лингвистике называется **деревом составляющих или деревом синтаксического анализа.** Вершины этого дерева помечаются символами, входящими в цепочки, порождаемые процессом вывода, корень дерева (его начальная вершина) — аксиомой σ. Из каждой вершины, помеченной нетерминальным символом, вместо которого на очередном шаге вывода (при соответствующих контекстах) вставляется некоторая цепочка, направленные дуги ведут к вершинам, помеченным символами вставляемой цепочки. Например, выводу (5.3) однозначно соответствует дерево вывода, изображенное на рис. 22; способам проведения выводов, начала которых совпадают с (5.4), соответствуют деревья, имеющие один из следующих верхних фрагментов (рис. 23).

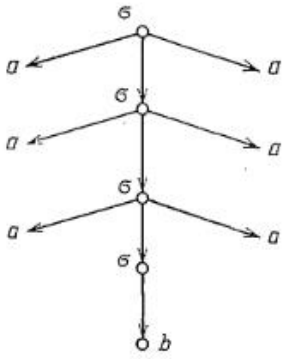


Рис. 22.

Рассмотрим некоторые понятия, связанные с деревьями выводов. Будем говорить, что вершина q' является непосредственным продолжением вершины q в некотором дереве вывода T, если стрелка из q направлена к вершине q'. Вершина q' является продолжением вершины q, если существует последовательность вершин $q_1, q_2, \ldots, q_n$ таких, что $q_1 = q$, $q_n = q'$ и вершина q_{i+1} — непосредственное продолжение вершины q_i при любом $i = 1, 2, \ldots, n-1$. Последовательность $\Pi = q_1, q_2, \ldots, q_n$, удовлетворяющая приведенному условию, называется **путем в дереве вывода** T, ведущим из q в q'. Вершина q называется **максимальной**, если не существует вершины, являющейся продолжением q. Иными словами, из максимальной вершины не выходит ни одна стрелка. В дереве вывода, соответствующем способу проведения некоторого полного вывода каждая максимальная вершина помечается терминальным символом (см. рис. 22, вершины a, b).

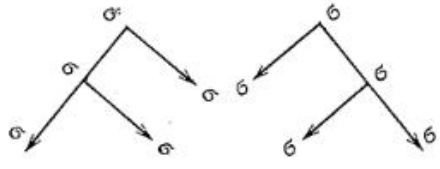


Рис. 23.

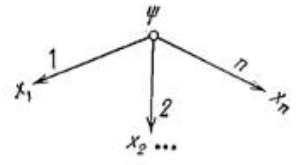


Рис. 24.

Пусть T — некоторое дерево вывода и $y = x_1 \ldots x_n$ — цепочка, вставляемая на некотором шаге вывода (при соответствующих контекстах) вместо нетерминального символа ψ. Перенумеруем

дуги, выходящие из соответствующей вершины, помеченной символом ψ, в соответствии с вхождениями символов x_i $(1 \leqslant i \leqslant n)$ в подставляемую цепочку y (рис. 24). Например, дерево вывода, изображенное на рис. 22, после перенумерации дуг имеет вид, представленный на рис. 25. Перенумерация дуг в дереве вывода T, а также понятие продолжения позволяют упорядочить вершины дерева T. А именно: для вершин q, q' справедливо отношение $q \rightarrowtail q'$, если выполняется одно из следующих условий: а) вершины q и q' совпадают; б) вершина q' является продолжением вершины q; в) если $q_0 q_1 \ldots q_r q$ и $q_0 q_1' \ldots q_s' q'$ — пути в дереве T, ведущие от корня q_0 к вершинам q и q' соответственно, причем k — наибольший индекс такой, что вершины q_i и q_i' совпадают $(q_i = q_i')$ при любом $i = 1, 2, \ldots, k$), то $n_{k+1} \leqslant n_{k+1}'$, где n_{k+1} (n_{k+1}') — номер дуги, ведущий к вершине q_{k+1} (q_{k+1}').

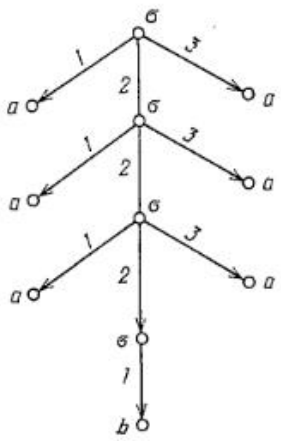


Рис. 25.

Отношение $\rightarrowtail$ является частичным порядком на множестве всех вершин дерева T. Более того, для любой пары несовпадающих вершин q, q' дерева T справедливо отношение $q \rightarrowtail q'$ либо $q' \rightarrowtail q$, и, следовательно, отношение $\rightarrowtail$ образует линейный порядок на множестве всех вершин дерева T (см. § 2.4).

Если дерево T соответствует способу проведения некоторого полного вывода в нс-грамматике G, то пометки при упорядоченной последовательности всех максимальных вершин составляют правильную цепочку $x \in L(G)$, порожденную данным выводом. Для произвольной вершины q дерева T обозначим через S_q упорядоченное множество максимальных вершин данного дерева, являющихся продолжением вершины q. Если q — максимальная вершина дерева T, то $S_q = \{q\}$. Пометки при вершинах, входящих в S_q, составляют терминальную цепочку x_q такую, что множеству S_q однозначно соответствует вхождение (x_q, t) в цепочку x, порожденную данным выводом.

Совокупность $C_T = \{x_q, t\}$, где q пробегает множество всех вершин дерева T, образует систему составляющих цепочки x. Действительно, в систему C_T входят все атомные составляющие цепочки x, которые являются вхождениями пометок при максимальных вершинах дерева T. Система C_T содержит также вхождение $(x, 1)$, которое соответствует множеству C_{q_0}, где q_0 — корень дерева T, помеченный аксиомой σ. Следовательно, система C_T удовлетворяет условию 1 в определении системы составляющих. Кроме того, для произвольных вершин q и q' дерева T либо q' является про-

должением q и тогда $S_{q'} \subset S_q$, либо q является продолжением q' и тогда $S_q \subset S_{q'}$, либо множества S_q и $S_{q'}$ не пересекаются, так как ни в одну вершину дерева не может входить более одной дуги. Следовательно, система C_T удовлетворяет условию 2 в определении системы составляющих.

Система C_T называется **системой составляющих цепочки x, ассоциированной с деревом вывода T.** Таким образом, справедливо следующее утверждение.

Теорема 5.2. *Для произвольной нс-грамматики G по дереву вывода T, соответствующему способу проведения некоторого полного вывода цепочки $x \in L(G)$, можно построить систему составляющих C_T, ассоциированную с деревом T.*

Например, дереву вывода, изображенному на рис. 25, соответствует система составляющих

$$\{(a^3ba^3, 1),\ (a^2ba^2, 2),\ (aba, 3),\ (b, 4),\ (a, 1),$$
$$(a, 2),\ (a, 3),\ (a, 5),\ (a, 6),\ (a, 7)\}.$$

Заметим, что в системе составляющих, ассоциированной с некоторым деревом вывода, каждой неатомной (возможно и некоторым атомным) составляющей соответствует синтаксический тип — нетерминальный символ, являющийся пометкой при вершине, от которой происходит данная составляющая. Так, в нашем примере каждой неатомной составляющей $(a^3ba^3, 1)$, $(a^2ba^2, 2)$, $(aba, 3)$, а также атомной составляющей $(b, 4)$ соответствует синтаксический тип σ.

Обращаясь вновь к определению нс-грамматик, заметим, что требование непустоты цепочки, вставляемой вместо нетерминального символа в продукции нс-грамматик, весьма существенно. Снятие этого ограничения приводит к понятию **обобщенной нс-грамматики**, или, кратко, **онс-грамматики.**

Справедлива следующая теорема.

Теорема 5.3. *По произвольной порождающей грамматике $G = (\mathfrak{A}, V_н, \sigma, P)$ может быть построена эквивалентная ей онс-грамматика.*

Подробное доказательство этой теоремы приведено в работе [20]. Основная его идея состоит в следующем. По теореме 5.1 грамматика G может быть представлена в нормальной форме $G' = (\mathfrak{A}, V'_н, \sigma, P')$. Далее по грамматике G' строится эквивалентная ей онс-грамматика G^*. Для этого из P' выбирается произвольная продукция $y_i \to z$, где $y_i \in F(V'_н)$, $z_i \in F(V'_н \cup \mathfrak{A})$. Возможны три вида продукций.

1. $|y_i| = 0$. Тогда $y_i = e$ — пустое слово. Продукциям такого вида в грамматике G^* соответствует совокупность продукций

$$\{\psi \to \psi z_i,\ \psi \to z_i \psi \mid \text{для любого } \psi \in V'_н\}, \qquad (5.6)$$

которая позволяет вставлять цепочку z_i после (или впереди) любого нетерминального символа из $V'_н$.

2. $|y_i| = 1$. Продукции $y_i \to z_i$ входят в схему грамматики G^* без всяких изменений.

3. $y_i = \psi_1\psi_2 \ldots \psi_r (r > 1)$.

В грамматике G^* эквивалентное преобразование осуществляется упорядоченной совокупностью продукций

$$\left.\begin{array}{l}
\psi_1\psi_2 \ldots \psi_r \to \tau_1^i\psi_2 \ldots \psi_r, \\
\tau_1^i\psi_2 \ldots \psi_r \to \tau_1^i\tau_2^i\psi_3 \ldots \psi_r, \\
\ldots\ldots\ldots\ldots\ldots \\
\tau_1^i\tau_2^i \ldots \tau_{r-1}^i\psi_r \to \tau_1^i\tau_2^i \ldots \tau_r^i, \\
\tau_1^i\tau_2^i \ldots \tau_r^i \to \tau_2^i\tau_3^i \ldots \tau_r^i, \\
\tau_2^i\tau_3^i \ldots \tau_r^i \to \tau_3^i \ldots \tau_r^i, \\
\ldots\ldots\ldots\ldots\ldots \\
\tau_{r-1}^i\tau_r^i \to \tau_r^i, \\
\tau_r^i \to z_i,
\end{array}\right\} \qquad (5.7)$$

где $\tau_1^i, \tau_2^i, \ldots, \tau_r^i \in V'_н$ — новые нетерминальные символы, причем для различных продукций вида 3 эти символы различны: $\tau_s^i \neq \tau_t^j$ (при $i \neq j$).

Объединение совокупностей продукций (5.6) и (5.7), соответствующих всем продукциям вида 1 и 3 грамматики G, а также продукций вида 2 образует схему P^* в грамматике G^*. Из вида продукций схемы P^* следует, что G^* является онс-грамматикой. Кроме того, каждому полному выводу W цепочки $x \in L(G)$ соответствует полный вывод этой же цепочки в грамматике G^*, в котором преобразования, соответствующие применениям продукций вида 1 или 3, осуществляются с помощью применения продукций из совокупности (5.6) или (5.7) соответственно. Наоборот, любой правильный вывод в грамматике G^* можно перестроить так, что продукции, входящие в какую-либо совокупность (5.7), будут применяться одна за другой в нужном порядке. Тогда соответствующую часть вывода можно заменить применением некоторой продукции в грамматике G' вида 3, а продукции, входящие в совокупность (5.6), — соответствующими продукциями грамматики G' вида 1.

Таким образом, каждому правильному выводу цепочки x в грамматике G^* соответствует правильный вывод этой же цепочки в грамматике G' и $G^* \sim G'$. Но $G \sim G'$, следовательно, $G^* \sim G$. Теорема доказана.

Итак, требование непустоты цепочки, вставляемой вместо нетерминального символа в продукциях нс-грамматики, весьма

существенно, следовательно, продукция $y_i \to z_i$ схемы некоторой нс-грамматики удовлетворяет условию

$$|y_i| \leqslant |z_i|. \tag{5.8}$$

Грамматики, продукции которых удовлетворяют этому условию, называются **неукорачивающими**. В частности, неукорачивающей является любая нс-грамматика.

Справедлива также следующая теорема, дающая некоторое представление о широте понятия нс-грамматики.

Теорема 5.4. *Для произвольной неукорачивающей грамматики* $G = (\mathfrak{A}, V_н, \sigma, P)$ *существует эквивалентная нс-грамматика.*

Как и при доказательстве теоремы (5.3), вначале переходим от грамматики G к эквивалентной ей грамматике G' (теорема 5.1). Затем строим совокупности продукций, равносильные продукциям $y_i \to z_i$ грамматики G. При $|y_i| \leqslant 1$ построение аналогично построению при доказательстве теоремы 5.3. При $|y_i| > 1$ существенно используется основное свойство (5.8), которому по определению удовлетворяют продукции любой неукорачивающей грамматики.

Пусть $y_i = \psi_1\psi_2 \ldots \psi_r$; $z_i = \sigma_1\sigma_2 \ldots \sigma_s$, где $r \leqslant s$, $y_i \in F(V'_н)$, $z_i \in F(\mathfrak{A} \cup V'_н)$. Тогда продукции $\psi_1\psi_2 \ldots \psi_r \to \sigma_1\sigma_2 \ldots \sigma_s$ равносильна упорядоченная совокупность продукций

$$\begin{array}{l}
\psi_1\psi_2 \ldots \psi_r \to \tau_1^i\psi_2 \ldots \psi_r, \\
\tau_1^i\psi_2 \ldots \psi_r \to \tau_1^i\tau_2^i\psi_3 \ldots \psi_r, \\
\ldots\ldots\ldots\ldots\ldots\ldots \\
\tau_1^i\tau_2^i \ldots \tau_{r-1}^i\psi_r \to \tau_1^i\tau_2^i \ldots \tau_r^i, \\
\tau_1^i\tau_2^i \ldots \tau_r^i \to \sigma_1\tau_2^i \ldots \tau_r^i, \\
\sigma_1\tau_2^i \ldots \tau_r^i \to \sigma_1\sigma_2\tau_3^i \ldots \tau_r^i, \\
\ldots\ldots\ldots\ldots\ldots\ldots \\
\sigma_1\sigma_2 \ldots \sigma_{r-1}\tau_r^i \to \sigma_1\sigma_2 \ldots \sigma_{r-1}\sigma_r\sigma_{r+1} \ldots \sigma_s,
\end{array} \tag{5.9}$$

где $\tau_1^i, \ldots, \tau_r^i$ — новые нетерминальные символы. Построенные таким образом совокупности продукций, равносильные продукциям грамматики G', образуют схему некоторой нс-грамматики G^*. В остальном доказательство аналогично доказательству теоремы 5.3.

Следствие. Языки $\{a^nb^nc^n \mid n = 1, 2, \ldots\}$ и $\{a^nb^na^n \mid n = 1, 2, \ldots\}$ являются нс-языками.

Действительно, каждый из этих языков порождается неукорачивающей грамматикой (см. § 5.1, пример 2).

Заметим, что языки, порожденные неукорачивающими или нс-грамматиками, являются рекурсивными множествами, т. е. существует распознающий алгоритм, позволяющий для любого языка $L(G)$, порожденного неукорачивающей грамматикой, установить,

является ли произвольная терминальная цепочка правильной ($x \in L(G)$). Суть этого алгоритма заключается в следующем.

Пусть $G = (\mathfrak{A}, V_н, \sigma, P)$ — неукорачивающая грамматика и x — произвольная цепочка длины n в алфавите $\mathfrak{A}$. При этом учитывается, что цепочки в процессе вывода не укорачиваются. Каждый шаг алгоритма состоит в построении некоторого полного вывода в грамматике G с последующей проверкой, завершается ли данный вывод цепочкой x. Если существует вывод, заканчивающийся цепочкой x, то, очевидно, $x \in L(G)$. В противном случае цепочка x неправильна. При этом рассматриваются лишь бесповторные выводы (не содержащие двух совпадающих цепочек), так как исключение всякой петли части вывода, заключенной между парой совпадающих цепочек, не влияет на начальную и конечную цепочки данного вывода. Ввиду того что грамматика G неукорачивающая, длины цепочек, получающихся в процессе вывода, не уменьшаются. Отсюда следует, что существует лишь конечное число выводов, завершающихся цепочками, длины которых не превышают n (длина цепочки x). Можно показать, что число различных таких выводов заведомо не превышает $(p^{n+2})!$, где p — число элементов, входящих в $\mathfrak{A} \cup V_н$ [22].

Таким образом, описанный алгоритм эффективно (не более чем за $(p^{n+2})!$ шагов) распознает, принадлежит ли цепочка x длины n языку $L(G)$.

§ 5.3. Контекстно-свободные языки

В предыдущем параграфе рассматривался важный класс порождающих грамматик, нс-грамматик. Продукции нс-грамматик позволяют вставлять некоторую непустую цепочку в объединенном алфавите $V = \mathfrak{A} \cup V_н$ вместо некоторого нетерминального символа при вполне определенных для данной продукции контекстах. В частности, можно рассматривать нс-грамматики, все продукции которых содержат пустые контексты. Такие грамматики называются **контекстно-свободными** или **кс-грамматиками**. Более точно: грамматика $G = (\mathfrak{A}, V_н, \sigma, P)$ называется **контекстно-свободной**, если каждое правило системы P имеет вид $\psi \to z$, где $\psi \in V_н$, z — непустая цепочка[2] над алфавитом $V = \mathfrak{A} \cup V_н$.

Язык $L(G)$, порожденный некоторой кс-грамматикой G, называется **кс-языком**. Так, приведенные в примерах 1—5, 7 § 5.2 грамматики и соответствующие им языки являются соответственно кс-грамматиками и кс-языками.

---

[2] Обычно в кс-грамматиках допускаются продукции с пустой правой частью, однако, как показано далее, это не приводит к существенному расширению класса порождаемых языков.

Из определения кс-грамматики (кс-языка) следует, что любая кс-грамматика (кс-язык) есть нс-грамматика (нс-язык), тогда как обратное утверждение уже неверно. Известны примеры нс-языков, которые не порождаются кс-грамматиками. К таким нс-языкам, в частности, относится язык L_6 (см. § 5.2, пример 6). Следовательно, класс нс-грамматик существенно шире класса кс-грамматик, т. е. множество кс-языков является собственным подмножеством множества нс-языков.

На множество кс-грамматик можно распространить некоторые введенные в § 5.2 понятия, связанные с нс-грамматиками (способ проведения вывода, дерево вывода, система составляющих, ассоциированная с данным деревом вывода, и др.). В частности, при доказательстве теорем в теории кс-грамматик бывает удобно рассматривать выводы с фиксированными способами проведения (левый и правый выводы).

Пусть $G = (\mathfrak{A}, V_н, \sigma, P)$ — произвольная кс-грамматика. **Левым выводом** цепочки $x \in L(G)$ в грамматике G называется вывод

$$W = (\sigma, z_1, \ldots, z_{n-1}, x) \tag{5.10}$$

со способом проведения

$$([q_1, p_1], [q_2, p_2], \ldots, [q_{n-1}, p_{n-1}]), \tag{5.11}$$

где $p_i : \psi \to y$ — правило грамматики G, примененное на i-м шаге к цепочке $z_i = u_i \psi v_i$, причем подцепочка u_i терминальна ($u_i \in F(\mathfrak{A})$, $|u_i| = q_i$) при любом $i = 1, 2, \ldots, n-1$. Иными словами, на каждом шаге левого вывода продукции p_i применяется к самому левому нетерминальному символу цепочки z_i ($i = 1, 2, \ldots, n-1$). Аналогично на каждом шаге **правого вывода** продукция p_i применяется к самому правому нетерминальному символу цепочки z_i, т. е. для способа проведения (5.11) правого вывода $z_i = u_i \psi v_i$, где $\psi \in V_н$ — левая часть правила p_i и $v_i \in F(\mathfrak{A})$.

Очевидно, каждому полному выводу некоторой цепочки x в кс-грамматике соответствует единственный левый (правый) вывод этой же цепочки. Например, пусть дана кс-грамматика

$$G_1 = (\{a, b\}, \{\sigma, \psi\}, \sigma, P), \tag{5.12}$$

где

$$P = \begin{cases} \sigma \to \sigma\psi, \\ \psi \to \psi\psi, \\ \sigma \to a, \\ \psi \to b. \end{cases}$$

Тогда выводы

$$(\sigma, \sigma\psi, a\psi, a\psi\psi, a\psi\psi\psi, ab\psi\psi, abb\psi, abbb), \tag{5.13}$$

$$(\sigma, \sigma\psi, \sigma\psi\psi, \sigma\psi\psi\psi, \sigma\psi\psi b, \sigma\psi bb, \sigma bbb, abbb) \tag{5.14}$$

являются соответственно левым и правым выводами цепочки $abbb$ в грамматике G_1.

Каждый полный вывод цепочки $x \in F(\mathfrak{A})$ в кс-грамматике $G = (\mathfrak{A}, V_н, \sigma, P)$ в результате некоторой перестановки правил, использованных при проведении данного вывода, может быть преобразован в левый (правый) вывод данной цепочки, так, что левый (правый) вывод является **канонической формой** проведения выводов в кс-грамматике G. Например, вывод $(\sigma, \sigma\psi, \sigma\psi\psi, \sigma b\psi, \sigma b\psi\psi, \sigma bb\psi, abb\psi, abbb)$ после изменения порядка применения продукций может быть преобразован в левый (правый) вывод цепочки $abbb$ в грамматике G_1. Заметим, что выводам, переводящимся друг в друга в результате некоторой перестановки в применении правил вывода, соответствует одно и то же дерево вывода.

Грамматика G называется **определенной**, если каждая правильная цепочка $x \in L(G)$ имеет единственный левый вывод, и неопределенной — в противном случае. При решении задач автоматизации программирования важную роль играет свойство определенности грамматики. В случае, когда кс-грамматика G не определена, т. е. для правильной цепочки $x \in L(G)$ существуют различные левые (правые) выводы, возникает проблема: какой вывод следует рассматривать, чтобы правильно обработать данную цепочку. Поэтому понятия определенности и неопределенности относятся к основным понятиям теории кс-языков и хорошо изучены [19].

Остановимся на требовании о непустоте правых частей продукций кс-грамматики. Это требование позволяет рассматривать кс-грамматики как частный случай нс-грамматик, в которых непустота правых частей продукции имеет весьма существенное значение (теорема 5.3). Выясним, какую роль играет требование непустоты правых частей продукций в кс-грамматиках.

Обобщенной кс-грамматикой или **окс-грамматикой** называется грамматика $G = (\mathfrak{A}, V_н, \sigma, P)$ такая, в которой каждая продукция схемы P имеет вид $\psi \to z$, где $\psi \in V_н$; z — произвольная цепочка над алфавитом $V = \mathfrak{A} \bigcup V_н$.

Теорема 5.5. *Для произвольной окс-грамматики* $G = (\mathfrak{A}, V_н, \sigma, P)$ *может быть построена кс-грамматика* $G' = (\mathfrak{A}, V_н, \sigma, P')$ *такая, что* $L(G') = L(G) \setminus e$, *причем, если* $e \overline{\in} L(G)$, *то* $L(G) = L(G')$; *в этом случае грамматики* G *и* G' *эквивалентны.*

Доказательство. Пусть $G = (\mathfrak{A}, V_н, \sigma, P)$ — произвольная окс-грамматика. Выполним индуктивное построение множества $M \subseteq V_н$ так, что $\psi \in M$, если в системе P есть правило $\psi \to e$. Далее, если $\psi_1, \psi_2, \ldots, \psi_k \in M$ и в систему P входит продукция $\psi \to \psi_1\psi_2 \ldots \psi_k$, то $\psi \in M$. Ввиду конечности алфавита $V_н$ и системы продукций P множество M может быть построено за конечное число шагов. Продукции, использованные при построении множества M, назовем **продукциями с исчезающей правой частью**. Затем каждой продукции $p \in P$, имеющей вид $p : \psi \to z$, где z не-

пусто, поставим в соответствие систему $s(p)$ всех продукций вида $\psi \to z'$, где цепочка $z' \neq e$ получена из z в результате исключения некоторой (возможно, пустой) совокупности символов из M. Например, пусть

$$G_1 = (\{a, b\}, \{\sigma, \psi_1, \psi_2, \psi_3\}, \sigma, P_1), \tag{5.15}$$

где

$$P_1 = \begin{cases} p_1 : \sigma \to a\sigma b, \\ p_2 : \sigma \to e, \\ p_3 : \psi_1 \to \psi_2\psi_3, \\ p_4 : \psi_2 \to \sigma\psi_3, \\ p_5 : \psi_3 \to e. \end{cases}$$

По схеме P_1 строим множество M_1. Очевидно, $\sigma, \psi_3 \in M_1$. Кроме того, из продукции p_4 следует, что $\psi_2 \in M_1$, а из продукции p_3 — что $\psi_1 \in M_1$. Таким образом, $M_1 = \{\sigma, \psi_1, \psi_2, \psi_3\}$ и p_2, p_3, p_4, p_5 — продукции с исчезающей правой частью. Продукции p_1 соответствует совокупность

$$S(p_1) = \begin{cases} p_{11} : \sigma \to a\sigma b, \\ p_{12} : \sigma \to ab, \end{cases}$$

а продукциям p_3 и p_4 — совокупности

$$S(p_3) = \begin{cases} p_{31} : \psi_1 \to \psi_2\psi_3, \\ p_{32} : \psi_1 \to \psi_3, \\ p_{33} : \psi_1 \to \psi_2, \end{cases}$$

$$S(p_4) = \begin{cases} p_{41} : \psi_2 \to \sigma\psi_3, \\ p_{42} : \psi_2 \to \psi_3, \\ p_{43} : \psi_2 \to \sigma. \end{cases}$$

Рассмотрим кс-грамматику $G' = (\mathfrak{A}, V_н, \sigma, P')$, где $P' = \bigcup\limits_p S(p)$ для всех продукций $p \in P$ с непустой правой частью. В приведенном выше примере окс-грамматике G_1 соответствует кс-грамматика $G_1' = (\mathfrak{A}, V_н, \sigma, P_1')$, где $P_1' = S(p_1) \cup S(p_3) \cup S(p_4)$. Покажем, что языки $L(G)$ и $L(G')$ совпадают с точностью до пустого слова. Действительно, применение каждой продукции $p' \in S(p)$ такой, что $p' \overline{\in} P$, можно заменить применением продукции p и некоторой последовательности продукций с исчезающей правой частью. Например, применение продукции $p_{12} : \sigma \to ab$ грамматики G_1' можно заменить в грамматике G_1 фрагментом вывода $(\sigma, a\sigma b, ab)$. Следовательно, справедливо заключение $L(G') \subseteq L(G)$.

Покажем теперь справедливость включения $(L(G) \setminus e) \subseteq L(G')$. Выберем произвольную непустую цепочку $x \in L(G)$. Пусть $W = (z_0, z_1, \ldots, z_n)$ — левый вывод цепочки $x = z_n$ в окс грамматике G; T — дерево, соответствующее выводу W (см. § 5.2). Достаточно предположить, что на некоторых шагах вывода W применяются продукции вида

$$\psi \to e, \tag{5.16}$$

в противном случае W является полным выводом в грамматике G' и $x \in L(G')$. Последовательно исключая применения продукций вида (5.16), можно преобразовать W в вывод W' цепочки x в грамматике G'.

Действительно, пусть в выводе W продукция вида (5.16) впервые применяется на i-м шаге ($1 \leqslant i \leqslant n$) к вхождению (ψ, q) символа ψ в цепочку $z_{i-1} = y\psi y_1$, где $|y| = q - 1$. Так как цепочка x непустая, то, очевидно, $i \neq 1$ и по крайней мере на одном из шагов, предшествующих i-му, применяется продукция

$$\tau \to z \tag{5.17}$$

такая, что $|z| > 1$, в противном случае $z_i = x = e$. Тогда можно указать наибольший номер $m < i$ такой, что на m-м шаге в выводе W будет применяться продукция p_m вида (5.17), $p_m : \tau \to u\psi'v$. При этом $z_m = y'u\psi'vW'$ и в дереве T есть путь Π из вершины, помеченной (ψ', q'), в вершину (ψ, q), где (ψ', q') — вхождение выделенного символа ψ' в цепочку z_m; (ψ, q) — вхождение ψ в цепочку z_i.

Пусть последовательность нетерминальных символов, которыми помечены вершины пути Π, имеет вид

$$\psi' = \psi_0, \psi_1, \ldots, \psi_r = \psi. \tag{5.18}$$

Так как m — максимальный номер, то из каждой вершины ψ_j ($j = 0, 1, \ldots, r-1$) выходит точно одна стрелка и каждому такому переходу соответствует применение правила

$$\psi_j \to \psi_{j+1} \tag{5.19}$$

на m_j-м шаге вывода W, $m < m_j < i$. Учитывая применение продукции $\psi \to e$ на i-м шаге вывода W, заключаем, что продукции (5.16) относятся к продукциям с исчезающей правой частью и все символы, входящие в последовательность (5.18), принадлежат множеству M. Это означает, в частности, что в совокупность продукций $S(p_m) \subseteq P'$ входит продукция $p_m' : \tau \to uv$.

Перестроим теперь вывод W следующим образом: на m-м шаге вместо продукции p_m применим продукцию p_m' и устраним последующие m_j-е ($j = 0, 1, \ldots, r$) шаги вывода, связанные с преобразованием исчезнувшего символа ψ', а также i-й шаг вывода W. Очевидно, подобная перестройка не повлияет на выводимость конечной цепочки $x \in L(G)$, в то же время число применений продукций (5.16) уменьшится на единицу по сравнению с выводом W. Продолжив описанный процесс, преобразуем вывод W цепочки x окс-

грамматики G в вывод W' этой же цепочки, на каждом шаге которого применяются продукции схемы p' так, что применение продукций вида (5.16) полностью исключено.

Таким образом, вывод W' цепочки x является выводом в кс-грамматике G' и $x \in L(G')$. Из произвольности выбора непустой цепочки $x \in L(G)$ следует справедливость включения $(L(G) \setminus e) \subseteq L(G')$. Теорема доказана.

На основании доказанной теоремы всякую окс-грамматику $G = (\mathfrak{A}, V_н, \sigma, P)$ можно преобразовать в эквивалентную окс-грамматику $\tilde{G} = (\mathfrak{A}, \tilde{V}_н, \tilde{\sigma}, \tilde{P})$ такую, в которой для любого нетерминального символа $\psi \in (\tilde{V}_н \setminus \tilde{\sigma})$ схема $\tilde{P}$ не содержит продукций вида $\psi \to e$ и аксиома $\tilde{\sigma}$ не входит в правые части продукций.

Действительно, для грамматики G существует кс-грамматика $G' = (\mathfrak{A}, V_н, \sigma, P')$, удовлетворяющая условиям теоремы 5.5. Грамматика $\tilde{G}$ строится по кс-грамматике G' следующим образом. В качестве аксиомы выбирается новый нетерминальный символ $\tilde{\sigma} \notin V_н$ так, что $\tilde{V} = V_н \cup \tilde{\sigma}$. Для системы продукций $\tilde{P}$ выполняется включение $P' \subset \tilde{P}$. Кроме того, для каждой продукции вида $\sigma \to z$ из P' в систему $\tilde{P}$ включается продукция $\tilde{\sigma} \to z$. Система $\tilde{P}$ содержит также продукцию $\tilde{\sigma} \to e$, если $e \in L(G)$. Заметим, что аксиома $\tilde{\sigma}$ не входит в правые части продукций системы $\tilde{P}$. Отсюда, ввиду отсутствия в схеме $\tilde{P}$ продукций вида $\psi \to e$ для любого нетерминального $\psi \neq \tilde{\sigma}$, следует, что цепочки $z \neq e$, порожденные при произвольном выводе грамматики $\tilde{G}$, не укорачиваются. Поэтому грамматика G называется **неукорачивающей формой** грамматики G. Так, неукорачивающей формой окс-грамматики G_1 (см. (5.15)), использованной при доказательстве теоремы 5.5, является окс-грамматика

$$\tilde{G}_1 = (\{a, b\}, \{\tau, \sigma, \psi_1, \psi_2, \psi_3\}, \tau, \tilde{P}_1), \tag{5.20}$$

схема которой имеет вид

$$\tilde{P}_1 = \begin{cases} p_0 : \tau \to e, \\ p_{11} : \sigma \to a\sigma b, \\ p'_{11} : \tau \to a\sigma b, \\ p_{12} : \sigma \to ab, \\ p'_{12} : \tau \to ab, \end{cases} \quad \begin{cases} p_{31} : \psi_1 \to \psi_2\psi_3, \\ p_{32} : \psi_1 \to \psi_3, \\ p_{33} : \psi_1 \to \psi_2, \\ p_{41} : \psi_2 \to \sigma\psi_3, \\ p_{42} : \psi_2 \to \psi_3, \\ p_{43} : \psi_2 \to \sigma. \end{cases}$$

Следствие. Любая окс-грамматика $G = (\mathfrak{A}, V_н, \sigma, P)$ может быть преобразована в эквивалентную ей окс-грамматику $\tilde{G}$, являющуюся неукорачивающей формой грамматики G.

Из теоремы 5.5 следует, что отказ от требования непустоты правых частей продукций в кс-грамматиках, в отличие от

кс-грамматик, не приводит к существенному расширению класса порождаемых языков. Поэтому понятия окс- и кс-грамматик часто не различаются. В дальнейшем под кс-грамматикой понимается произвольная окс-грамматика, представленная в неукорачивающей форме.

В схеме $\tilde{P}$ грамматики $\tilde{G} = (\mathfrak{A}, \tilde{V}_н, \tilde{\sigma}, \tilde{P})$ можно исключить все продукции вида

$$\psi \to \tau, \tag{5.21}$$

где $\psi, \tau \in \tilde{V}_н$. Для этого достаточно построить всевозможные выводы вида

$$\psi \overset{*}{\underset{\tilde{G}}{\Rightarrow}} \tau. \tag{5.22}$$

В силу конечности алфавита $\tilde{V}_н$ и схемы $\tilde{P}$, а также неукорачиваемости цепочек при выводе в грамматике $\tilde{G}$ построение всех выводов вида (5.22) осуществляется за конечное число шагов. Затем для каждого из выводов $\psi \overset{*}{\underset{\tilde{G}}{\Rightarrow}} \tau$ и любой продукции $\tau \to z$ схемы $\tilde{P}$, правая часть которой z не принадлежит $\tilde{V}_н$ (если такие продукции существуют), к схеме $\tilde{P}$ присоединяется новая продукция $\psi \to z$, после чего все продукции вида (5.21) могут быть устранены. Полученная таким образом грамматика $\bar{G} = (\mathfrak{A}, \tilde{V}_н, \tilde{\sigma}, \bar{P})$ эквивалентна грамматике $\tilde{G}$ и не содержит правил вида (5.21).

В качестве примера построим грамматику $\bar{G}_1$, эквивалентную грамматике $\tilde{G}_1$ (см. (5.20)). По схеме $\tilde{P}_1$ построим все выводы вида (5.22) в грамматике $\tilde{G}_1$:

$$\psi_1 \overset{*}{\Rightarrow} \psi_2, \quad \psi_1 \overset{*}{\Rightarrow} \psi_3, \quad \psi_1 \overset{*}{\Rightarrow} \sigma, \quad \psi_2 \overset{*}{\Rightarrow} \psi_3, \quad \psi_2 \overset{*}{\Rightarrow} \sigma.$$

Для вывода $\psi_1 \overset{*}{\Rightarrow} \psi_2$ и продукции $p_{41} \in \tilde{P}_1$ образуем продукцию $S_0 = \{\psi_1 \to \sigma\psi_3\}$, для вывода $\psi_1 \overset{*}{\Rightarrow} \sigma$ и продукций p_{11}, p_{12} — совокупность продукций $S_1 = \begin{cases} \psi_1 \to a\sigma b \\ \psi_1 \to ab \end{cases}$ и, наконец, для вывода $\psi_2 \overset{*}{\Rightarrow} \sigma$ и продукций p_{11}, p_{12} — совокупность продукций $S_2 = \begin{cases} \psi_2 \to a\sigma b \\ \psi_2 \to ab. \end{cases}$

Затем, присоединив к схеме $\tilde{P}_1$ совокупность продукций $S_0 \cup S_1 \cup S_2$ с последующим исключением продукций $p_{32}, p_{33}, p_{42}, p_{43}$, получим схему $\bar{P}_1$ грамматики

$$\bar{G}_1 = (\{a, b\}, \{\tau, \sigma, \psi_1, \psi_2, \psi_3\}, \tau, \bar{P}_1). \tag{5.23}$$

кс-Грамматика $G=(\mathfrak{A}, V_н, \sigma, P)$ существенно зависит от нетерминального символа $\psi \in V_н$, если выполняются следующие условия:

1) существует по крайней мере одна цепочка $z=u\psi v$, выводимая из аксиомы σ, $\sigma \overset{*}{\underset{G}{\Rightarrow}} z$;

2) существует хотя бы одна терминальная цепочка $x \in F(\mathfrak{A})$, выводимая из ψ, $\psi \overset{*}{\underset{G}{\Rightarrow}} x$.

В силу конечности алфавита $V_н$ и схемы P для любого $\psi \in V_н$ можно установить, существенно ли зависит грамматика $G=(\mathfrak{A}, V_н, \sigma, P)$ от символа $\psi \in V_н$. Очевидно, любая продукция грамматики G, содержащая хотя бы один **фиктивный** символ $\psi \in V_н$ (от которого грамматика G существенно не зависит), может быть исключена из системы P, а все фиктивные нетерминальные символы — из алфавита $V_н$.

кс-Грамматика $\overline{G}=(\mathfrak{A}, \overline{V}_н, \overline{\sigma}, \overline{P})$, существенно зависящая от каждого нетерминального символа, входящего в $\overline{V}_н$, называется **приведенной**.

Из изложенных выше рассуждений следует, что приведенная грамматика G^0 удовлетворяет следующим условиям:

1) приведенная грамматика неукорачивающая;

2) аксиома $\overline{\sigma}$ не входит в правые части продукций;

3) в схему грамматики не входят продукции вида (5.21).

Так, в грамматике G_1 (см. (5.23)) символы ψ_1, ψ_2 не удовлетворяют первому, а ψ_3 — обоим условиям существенной зависимости. Поэтому символы ψ_1, ψ_2, ψ_3 фиктивные и грамматике $\overline{G}_1$ эквивалентна приведенная грамматика $G_1^0=(\{a, b\}, \{\tau, \sigma\}, \tau, P_1^0)$, схема которой имеет вид

$$P_1^0=\begin{cases}\tau \to e,\\ \sigma \to a\sigma b,\\ \tau \to a\sigma b,\\ \sigma \to ab,\\ \tau \to ab.\end{cases}$$

Таким образом, справедлива следующая теорема.

Теорема 5.6. *Для произвольной кс-грамматики $G=(\mathfrak{A}, V_н, \sigma, P)$ существует эквивалентная ей приведенная кс-грамматика $G^0=(\mathfrak{A}, V_н^0, \sigma, P^0)$, где $V_н^0 \subseteq V_н$ — нетерминальный алфавит, полученный из $V_н$ в результате исключения всех фиктивных нетерминальных символов грамматики G; $P^0 \subseteq P$ — система всех продукций грамматики G, не содержащих фиктивных символов.*

Эта теорема позволяет рассматривать, не нарушая общности, только приведенные кс-грамматики.

По порождающей кс-грамматике $G=(\mathfrak{A}, V_н, \sigma, P)$ может быть построена система уравнений, задающая тот же язык $L(G)$ [95].

Пусть $V_н=\{\psi_1, \psi_2, \ldots, \psi_n\}$, где $\psi_1=\sigma$ — аксиома грамматики G. Каждому нетерминальному символу $\psi_i \in V_н$ поставим в соответствие уравнение $\psi_i ::= f_i(\psi_1, \ldots, \psi_n)$, где $f_i(\psi_1, \ldots, \psi_n) = z_{i_1} \cup z_{i_2} \cup \cdots \cup z_{i_{k_i}}$ — выражение, состоящее из всех правых частей продукций вида $\psi_i \to z_{ij} \in P$ $(j=1, 2, \ldots, k_i)$. Тогда грамматике G будет сопоставлена система уравнений

$$U_G=\{\psi_i ::= f_i(\psi_1, \psi_2, \ldots, \psi_n) \mid i=1, 2, \ldots, n\}. \quad (5.24)$$

Например, грамматике G_3 (см. § 5.2, пример 3) соответствует система U_{G_3}, состоящая из единственного уравнения

$$U_{G_3}=\{\sigma ::= a\sigma a \cup b\}, \quad (5.25)$$

а грамматике $G_2'=(\{a, b\}, \{\sigma, \psi\}, \sigma, P_2)$, схема которой имеет вид

$$P_2'=\begin{cases}\sigma \to \sigma\psi,\\ \sigma \to a,\\ \sigma \to b,\\ \psi \to a,\\ \psi \to b,\end{cases}$$

соответствует система, состоящая из двух уравнений

$$U_{G_2'}=\begin{cases}\sigma ::= f_1(\sigma, \psi)=\sigma\psi \cup a \cup b,\\ \psi ::= f_2(\sigma, \psi)=a \cup b.\end{cases} \quad (5.26)$$

Рассмотрим процесс порождения языка $L(G)$ по системе уравнений (5.24), соответствующей грамматике G. Пусть $\tilde{O}=(\varnothing, \ldots, \varnothing)$ — нулевой набор значений переменных $\psi_1, \ldots, \psi_n$, в котором $\psi_i=\varnothing$. На таком нулевом наборе функции $f_i(\psi_1, \ldots, \psi_n)$, являющиеся правыми частями уравнений системы (5.24), принимают значения $\alpha_i^0=f_i(\varnothing, \ldots, \varnothing)$ при каждом $i=1, 2, \ldots, n$. Совокупность этих значений обозначим через $\tilde{\alpha}^0=(\alpha_1^0, \ldots, \alpha_n^0)$ и рассмотрим значения $\alpha_1^1=f_1(\tilde{\alpha}^0), \ldots, \alpha_n^1=f_n(\tilde{\alpha}^0)$ функций $f_i(\psi_1, \ldots, \psi_n)$ на наборе $\tilde{\alpha}^0$: $\tilde{\alpha}^1=(\alpha_1^1, \ldots, \alpha_n^1)$. Рекурсивно продолжив процесс порождения, получим $\tilde{\alpha}^k=(\alpha_1^k, \ldots, \alpha_n^k)$, где $\alpha_i^k=f_i(\tilde{\alpha}^{k-1})=f_i(\alpha_1^{k-1}, \ldots, \alpha_n^{k-1})$ при любом $k=1, 2, \ldots$

Обозначим через α_i^∞ множество $\bigcup\limits_{k=0}^{\infty} \alpha_i^k$ при любом $i=1, 2, \ldots, n$. Набор $\tilde{\alpha}^\infty=(\alpha_1^\infty, \ldots, \alpha_n^\infty)$ называется **решением системы уравнений** (5.24), а каждая компонента α_i^∞ — **языком, соответствующим переменной** ψ_i (или языком типа АЛГОЛ [19]).

Язык α_1^∞, соответствующий аксиоме $\psi_1 = \sigma$, называется **языком, порожденным системой уравнений** (5.24). Можно показать, что язык α_1^∞, порожденный системой уравнений (5.24), совпадает с языком $L(G)$, порожденным соответствующей грамматикой G, т. е. $\alpha_1^\infty = L(G)$ [95]. Кроме того, набор $\tilde{\alpha}^\infty = (\alpha_1^\infty, \ldots, \alpha_n^\infty)$ является **минимальным решением системы уравнений** (5.24), так что если $\tilde{L} = (L_1, L_2, \ldots, L_n)$ — произвольное решение системы уравнений (5.24), то $\alpha_i^\infty \subseteq L_i$ при любом $i = 1, 2, \ldots, n$. Действительно, $\alpha_i^0 \subset L_i$ $(i = 1, 2, \ldots, n)$. Рассуждая индуктивно, в результате применения описанной выше рекурсии получим $\alpha_i^\infty \subset L_i$.

Таким образом, справедлива следующая теорема.

Теорема 5.7. *Язык L тогда и только тогда контекстно-свободный, когда существует система уравнений (5.24), имеющая минимальное решение $\tilde{\alpha}^\infty = (\alpha_1^\infty, \alpha_2^\infty, \ldots, \alpha_n^\infty)$ и такая, что $\alpha_1^\infty = L$.*

Проиллюстрируем процесс порождения языка по заданной системе уравнений на приведенных выше примерах. Построим язык, порожденный системой (5.25) и состоящий из одного уравнения $\sigma := f(\sigma) = a\sigma a \cup b$. Тогда

$$\alpha^0 = f(\varnothing) = a\varnothing a \cup b = \varnothing \cup b = b,$$
$$\alpha^1 = f(\alpha^0) = a(\alpha^0)a \cup b = aba \cup b,$$
$$\alpha^2 = f(\alpha^1) = a(\alpha^1)a \cup b =$$
$$= a(aba \cup b)a \cup b = aabaa \cup aba \cup b,$$
$$\alpha^3 = f(\alpha^2) = a(\alpha^2)a \cup b = a^3ba^3 \cup a^2ba^2 \cup aba \cup b$$

и т. д. Таким образом, $\alpha^\infty = L(G_3) = \{a^nba^n \mid n = 0, 1, 2, \ldots\}$ (см. § 5.2, пример 3).

Язык α_1^∞, порожденный системой уравнений (5.26), строится следующим образом: $\tilde{\alpha}^0 = (\alpha_1^0, \alpha_2^0)$, где $\alpha_1^0 = f_1(\varnothing, \varnothing) = \varnothing \cup a \cup b = a \cup b$, $\alpha_2^0 = f_2(\varnothing, \varnothing) = f_2(\alpha_1^k, \alpha_2^k) = a \cup b$ при любом $k = 0, 1, 2, \ldots$; $\tilde{\alpha}^1 = (\alpha_1^1, \alpha_2^1)$, где $\alpha_1^1 = f_1(\alpha_1^0, \alpha_2^0) = \alpha_1^0\alpha_2^0 \cup a \cup b = (a \cup b) \cdot (a \cup b) \cup a \cup b = aa \cup ab \cup ba \cup bb \cup a \cup b$; если $k = 2$, то $\tilde{\alpha}^2 = (\alpha_1^2, \alpha_2^2)$, где $\alpha_1^2 = f_1(\alpha_1^1, \alpha_2^1) = \alpha_1^1\alpha_2^1 \cup a \cup b = \alpha_1^1(a \cup b) \cup a \cup b$ и т. д.

Таким образом, язык α_1^∞, порожденный системой уравнений (5.26), представляет собой универсальный язык над алфавитом $\{a, b\}$, содержащий все непустые цепочки в данном алфавите (см. § 5.2, пример 2).

Как отмечалось выше, существуют нс-языки, не являющиеся контекстно-свободными языками. При решении этой проблемы теории кс-языков важную роль играет установление критериев, с помощью которых можно выяснить, является ли данный язык кс-языком. Необходимое условие, которому удовлетворяет любой кс-язык, устанавливается следующей теоремой [19].

Теорема 5.8. *Для любого бесконечного кс-языка L существует натуральная константа p такая, что каждое слово $z \in L$, длина которого больше p, можно представить в виде $z = z_1uwvz_2$, где $|uwv| \leqslant p$, по крайней мере одна из подцепочек u, v непуста и при любом натуральном n*

$$z_1u^nwv^nz_2 \in L \quad (n = 1, 2, \ldots).$$

Основная идея доказательства данной теоремы состоит в следующем. Пусть $L(G^0)$ — произвольный кс-язык, заданный приведенной кс-грамматикой $G^0 = (\mathfrak{A}, V_н, \sigma, P)$, и r — число различных символов в нетерминальном алфавите $V_н$. Рассмотрим множество D всех деревьев вывода, построенных с помощью продукций схемы P и таких, что $T \in D$ тогда и только тогда, когда корень дерева T помечен некоторым нетерминальным символом из $V_н$, причем длина каждого пути в дереве T не превышает $r + 1$. Очевидно, множество D конечно. В качестве константы p выберем длину наибольшей из цепочек, порожденных деревьями выводов из D. Пусть теперь $z \in L(G^0)$ — произвольная цепочка языка, длина которой больше p, $|z| > p$. В силу выбора константы p цепочку z порождает дерево вывода T, имеющее по крайней мере один путь, длина которого больше $r + 1$. Из всех путей дерева T, обладающих этим свойством, выберем максимальный по длине: $\Pi = (q_1, q_2, \ldots, q_s)$, где q_1 — корень дерева T, помеченный аксиомой σ грамматики G^0; q_s — максимальная вершина дерева T, помеченная символом основного алфавита $\mathfrak{A}$.

Рассмотрим конечный участок Π' пути Π, имеющий длину $r + 1$, $\Pi' = (q_{s-r}, q_{s-r+1}, \ldots, q_s)$. В силу приведенности грамматики G^0 аксиома σ не входит в правые части продукций схемы P, поэтому любая нетерминальная пометка при вершине q_j на участке Π' отлична от σ, $s - r \leqslant j \leqslant s - 1$. Это означает, что среди вершин участка Π' найдутся по крайней мере две — q_i и q_j — с одинаковой пометкой $\psi \in V_н$, причем вершина q_j — продолжение вершины q_i. Пусть T_i и T_j — поддеревья дерева T, имеющие в качестве корней указанные вершины и содержащие все вершины, являющиеся продолжением вершин соответственно q_i, q_j в дереве T. Так как вершина q_j служит продолжением q_i-й вершины, дерево T_j является поддеревом дерева T_i, причем если z', $w \in F(\mathfrak{A})$ — цепочки, порожденные деревьями T_i, T_j соответственно, то $z' = uwv$. Из приведенности грамматики G_0 следует, что хотя бы одна из цепочек u, v непустая, кроме того, z' является подцепочкой цепочки z, так что $z = z_1uwvz_2$. Заметим, что длина любого пути в дереве T_i не превышает $r + 1$, в противном случае путь Π не был бы максимальным в дереве T. Это означает, что $T_i \in D$, и поэтому цепочка $z' = uwv$ имеет длину $|z'| \leqslant p$.

Покажем, что при любом $n = 1, 2, \ldots$ $z_n = z_1 u^n w v^n z_2 \in z(G^0)$. Рассмотрим дерево вывода $\tilde{T}_i$, полученное из дерева T_i исключением всех вершин дерева T_j (отличных от q_j), а также всех дуг, ведущих в эти вершины. Дерево $\tilde{T}_i$ соответствует выводу $\psi \overset{*}{\underset{G}{\Rightarrow}} u\psi v$. Вставим в дерево T $n-1$ экземпляр дерева T_i так, чтобы корень очередного дерева T_i был склеен с той вершиной в предшествующем дереве $\tilde{T}_i$, которая помечена выделенным вхождением символа ψ. Построенное дерево (рис. 26) соответствует выводу $\sigma \overset{*}{\Rightarrow} z_1 u^n w v^n z_2$ в грамматике G^0. Теорема доказана.

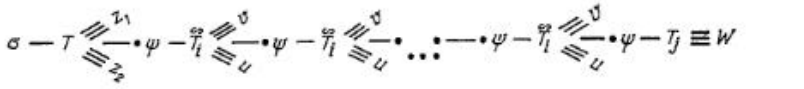


Рис. 26.

Следствие. Нс-язык $L_6 = \{xbx \mid x \in F(\mathfrak{A})\}$, где $\mathfrak{A} = \{a_1, a_2, \ldots, a_n\}$, $n > 1$ (см. § 5.2, пример 6), не является кс-языком.

Действительно, предположим противное. Пусть L_6 — кс-язык. Тогда в силу доказанной теоремы существует константа p такая, что каждую цепочку $z \in L_6$, где $|z| > p$, можно представить в виде $z = z_1 uwvz_2$, причем $|uwv| \leqslant p$, $uv \neq e$ и при любом n

$$z_1 u^n w v^n z_2 \in L_6. \tag{5.27}$$

В качестве z выберем цепочку $a_1^p a_2^p b a_1^p a_k^p \in L_6$. Очевидно, подцепочка uwv должна быть выбрана так, чтобы символ b входил в w, в противном случае соотношение (5.27) не выполняется в силу определения языка L_6. Но так как $|uwv| \leqslant p$, цепочка uwv входит в $a_2^p b a_1^p$. Отсюда следует, что и в этом случае соотношение (5.27) не выполняется. Таким образом, L_6 не является кс-языком.

Аналогично можно показать, что нс-языки $\{a^n b^n c^n \mid n = 1, 2, \ldots\}$ и $\{a^n b^n a^n \mid n = 1, 2, \ldots\}$ (см, § 5.1, пример 2; § 5.2, теорему 5.4 и следствие) не являются контекстно-свободными языками [19].

§ 5.4. Линейные и автоматные грамматики

Рассмотрим некоторые важные подклассы класса кс-грамматик.

Продукция $\psi \to z$ называется **линейной**, если z — некоторая терминальная цепочка, $z \in F(\mathfrak{A})$, либо $z = u\tau v$, где $u, v \in F(\mathfrak{A})$; $\psi, \tau \in V_н$. Иными словами, правые части линейных продукций либо являются терминальными цепочками, либо содержат лишь один нетерминальный символ.

Продукция $\psi \to z$ называется **праволинейной (леволинейной)**, если $z \in F(\mathfrak{A})$ либо $z = u\tau$ ($z = \tau u$), т. е. если единственный нетерминал τ входит в правые части праволинейных (леволинейных) продукций, то он всегда крайний справа (слева).

Грамматика $G = (\mathfrak{A}, V_н, \sigma, P)$, схема которой содержит только либо линейные, либо праволинейные, либо леволинейные продукции, называется соответственно **линейной, праволинейной, леволинейной.** Аналогично язык $L(G)$ называется **линейным, праволинейным, леволинейным,** если грамматика G относится к соответствующему классу грамматик.

Из приведенных определений следует справедливость включений $\mathfrak{L}_п \subseteq \mathfrak{L}$, $\mathfrak{L}_л \subseteq \mathfrak{L}$, где $\mathfrak{L}$, $\mathfrak{L}_п$, $\mathfrak{L}_л$ — множества соответственно всех линейных, праволинейных, леволинейных языков. Так, линейны грамматики и соответствующие им языки в примерах 1—5 § 5.2, праволинейны грамматики и языки в примерах 1,2 § 5.2.

Заметим, что приведенные определения линейных и праволинейных (леволинейных) грамматик отличаются от общеизвестных тем, что в правых частях продукций допускаются произвольные терминальные цепочки ($u, v \in F(\mathfrak{A})$), а не символы основного алфавита $\mathfrak{A}$, как обычно. Однако такое обобщение не приводит к существенному расширению соответствующих классов языков.

Справедливо следующее утверждение.

Теорема 5.9. *Каждой праволинейной грамматике соответствует эквивалентная ей леволинейная грамматика и наоборот, т. е.* $\mathfrak{L}_п = \mathfrak{L}_л$.

Действительно, праволинейной грамматике со схемой

$$\left.\begin{array}{ll} \sigma \to a\psi, & \psi \to b\psi, \\ \sigma \to b\psi, & \psi \to a, \\ \psi \to a\psi, & \psi \to b \end{array}\right\} \tag{5.28}$$

эквивалентна леволинейная грамматика со схемой

$$\left.\begin{array}{ll} \sigma \to \psi a, & \psi \to \psi b, \\ \sigma \to \psi b, & \psi \to a, \\ \psi \to \psi a, & \psi \to b, \end{array}\right\}$$

так как каждая из приведенных грамматик порождает универсальный язык над алфавитом $\mathfrak{A} = \{a, b\}$. Точно так же праволинейной грамматике со схемой

$$\left.\begin{array}{ll} \sigma \to a\psi, & \psi \to ac\tau, \\ \sigma \to ac\tau, & \tau \to b\tau, \\ \psi \to a\psi, & \tau \to b \end{array}\right\} \tag{5.29}$$

эквивалентна леволинейная грамматика со схемой

$$\left.\begin{array}{ll}\sigma' \to \tau b, & \psi \to \psi a,\\ \tau \to \tau b, & \tau \to ac,\\ \tau \to \psi ac, & \psi \to a,\end{array}\right\} \qquad (5.30)$$

так как каждая из этих грамматик порождает язык $\{a^n c b^m \mid n, m = 1, 2, \ldots\}$. В этих примерах грамматики представлены в приведенной форме (см. § 5.3).

Существует общий метод построения по приведенной форме праволинейной грамматики $G = (\mathfrak{A}, V_н, \sigma, P)$ эквивалентной ей леволинейной грамматики $G' = (\mathfrak{A}, V'_н, \sigma', P')$ и наоборот. Поясним суть этого метода на примере праволинейной грамматики со схемой (5.29). В качестве аксиомы грамматики G' выберем новый нетерминал $\sigma' \overline{\in} V_н$. Нетерминальный алфавит $V'_н$ грамматики G' содержит символ σ', а также все нетерминальные символы алфавита $V_н$, кроме аксиомы σ, $V'_н = (V_н \setminus \sigma) \cup \sigma'$. Построим леволинейную грамматику $G' = (\{a, b, c\}, \{\sigma', \psi, \tau\}, \sigma', P')$, эквивалентную грамматике со схемой (5.29). Схему P' левосторонней грамматики G' строим по схеме P. Первоначально в P выделяем все заключительные продукции вида $\sigma \to u$, каждой из которых соответствует продукция $\sigma' \to u \in P'$ (где $u \in F(\mathfrak{A})$). Затем каждой заключительной продукции вида $\xi \to u \in P$, где $\xi \neq \sigma$, сопоставляем леволинейную продукцию $\sigma' \to \xi u \in P'$. В грамматике со схемой (5.29) есть лишь одна заключительная продукция $\tau \to b$, которой соответствует леволинейная продукция $\sigma' \to \tau b \in P'$. Пусть $V_1 = \{\xi \mid \xi \neq \sigma\} \subseteq V_н$ — множество всех нетерминальных символов (отличных от аксиомы σ), составляющих левые части заключительных продукций грамматики G. Выберем совокупность продукций

$$\{\Delta \to u_1 \xi \mid \text{для всех } \xi \in V_1\} \subseteq P, \qquad (5.31)$$

где $u_1 \in F(\mathfrak{A})$. В данном примере $V_1 = \{\tau\}$, так что к выделенным продукциям относятся $\sigma \to ac\tau$, $\psi \to ac\tau$, $\tau \to b\tau$. Если среди указанных продукций есть продукции вида $\sigma \to u_1\xi$, то в грамматике G' им соответствуют заключительные продукции $\xi \to u_1$. Остальным продукциям вида $\Delta \to u_1\xi$ (где $\Delta \neq \sigma$) сопоставим леволинейные продукции, имеющие вид $\xi \to \Delta u_1 \in P'$. Продукции $\sigma \to ac\tau$ сопоставим продукцию $\tau \to ac \in P'$, заключительную в грамматике G', а продукциям $\psi \to ac\tau$ и $\tau \to b\tau$ — леволинейные продукции $\tau \to \psi ac$ и $\tau \to \tau b$ схемы P'. Далее образуем множество $V_2 = \{\Delta \mid \Delta \overline{\in} V_1 \cup \cup \sigma\} \subseteq V_н$ всех нетерминальных символов (отличных от аксиомы σ и не входящих в V_1), составляющих левые части продукций (5.31), выделим совокупность продукций $\{p \to u_2\Delta \mid \text{для всех } \Delta \in V_2\} \subseteq P$, где $u_2 \in F(\mathfrak{A})$, и выполним для нее построение, аналогичное предыдущему, т. е. каждой продукции вида $\sigma \to u_2\Delta$ сопоставим заключительную продукцию $\Delta \to u_2 \in P'$, а каждой продукции $\rho \to u_2\Delta$

(где $\rho \neq \sigma$) — леволинейную продукцию $\Delta \to \rho u_2 \in P'$ и т. д. В данном примере $V_2 = \{\psi\}$ и продукциям $\sigma \to a\psi$, $\psi = a\psi$ в грамматике G' соответствует заключительная продукция $\psi \to a$ и продукция $\psi \to \psi a$. В силу конечности множества $V_н$ и приведенности грамматики G этот процесс завершится на конечном шаге построения схемы P' леволинейной грамматики G'. Схема грамматики G', построенной с помощью изложенного метода, имеет вид (5.30).

Таким образом, используя схему P грамматики G, удается построить леволинейную грамматику G', причем каждому левостороннему выводу цепочки $x \in L(G)$ соответствует правосторонний вывод этой же цепочки в грамматике G' и наоборот. Следовательно, $L(G) = L(G')$. Теорема доказана.

Теперь достаточно рассмотреть лишь праволинейные грамматики, так как их основные свойства легко переносятся на множество леволинейных грамматик.

Пусть $G = (\mathfrak{A}, V_н, \sigma, P)$ — некоторая праволинейная грамматика с терминальным алфавитом $\mathfrak{A} = \{a_1, \ldots, a_n\}$. Применяя к грамматике G приемы, изложенные в § 5.3, исключаем продукции вида $\psi \to e$ и $\psi \to \tau$. Полученная в результате такого преобразования грамматика G^* порождает (с точностью до пустого слова e) тот же язык, что и грамматики $G: L(G^*) = L(G) \setminus \{e\}$. Затем каждой продукции $\psi_i \to u_i\tau_i \in P$, где $u_i = a_{i_1}a_{i_2} \ldots a_{i_k} \in F(\mathfrak{A})$, можно сопоставить, вводя множество $V(i) = \{\psi_{i_1}, \psi_{i_2}, \ldots, \psi_{i_{k-1}}\}$ дополнительных нетерминалов $\psi_{i_r} \overline{\in} V_н$ $(r = 1, 2, \ldots, k-1)$, совокупность продукций

$$P(i) = \begin{cases} \psi_i \to a_{i_1}\psi_{i_1}, \\ \psi_{i_1} \to a_{i_2}\psi_{i_2}, \\ \ldots\ldots\ldots\ldots \\ \psi_{i_{k-2}} \to a_{i_{k-1}}\psi_{i_{k-1}}, \\ \psi_{i_{k-1}} \to a_{i_k}\tau_i. \end{cases} \qquad (5.32)$$

Совокупность продукций, соответствующая заключительной продукции вида $\psi_i \to u_i$, может быть получена из (5.32), если заменить продукцию $\psi_{i_{k-1}} \to a_{i_k}\tau_i$ продукцией $\psi_{i_{k-1}} \to a_{i_k}$. Предполагается, что в различных продукциях схемы P множества дополнительных нетерминалов не пересекаются: $V(i) \cap V(j) = \varnothing$ при $i \neq j$. Очевидно, грамматика $G^a = (\mathfrak{A}, \tilde{V}_н, \sigma, \tilde{P})$, $\tilde{V}_н = V_н \cup (\bigcup_i V(i))$, схема которой получена в результате объединения продукций, входящих в $P(i)$, $\tilde{P} = \bigcup_i P(i)$, эквивалентна исходной грамматике G.

Грамматика G^a, каждая продукция которой имеет вид $\psi \to z$, где $z = \begin{Bmatrix} a\tau \\ a \end{Bmatrix}$, $(a \in \mathfrak{A}, \tau \in \tilde{V}_н)$, называется **автоматной** или **грамма-**

тикой с **конечным числом состояний** [110]. Так, автоматны грамматики в примерах 1, 2 § 5.2. Рассмотренные выше грамматики со схемами (5.29), (5.30), очевидно, не автоматны. Им эквивалентна, однако, автоматная грамматика со схемой

$$\left.\begin{array}{l}\sigma \to a\psi,\ \ \tau \to b\tau,\\ \psi \to a\psi,\ \ \tau \to b.\\ \psi \to c\tau,\end{array}\right\} \qquad (5.33)$$

Таким образом, из приведенных выше рассуждений следует, что для любой праволинейной грамматики G найдется автоматная грамматика G^a, порождающая (с точностью до пустого слова e) тот же язык, что и грамматика $G: L(G^a) = L(G) \setminus \{e\}$.

Существует тесная связь между автоматными грамматиками и **конечными автоматами** [100] (чем и объясняется термин «автоматная грамматика»). С каждой автоматной грамматикой $G^a = (\mathfrak{A}, \tilde{V}_н, \sigma, \tilde{P})$ ассоциирован конечный автомат $A = (\mathfrak{A}, S, \sigma, \varphi, \Gamma)$, где $\mathfrak{A}$ — входной алфавит автомата A, совпадающий с терминальным алфавитом грамматики G^a; $S = \tilde{V}_н \cup \{\varphi\}$ $(\varphi \overline{\in} \tilde{V}_н)$ — множество состояний; σ и φ соответственно начальное и заключительное состояния; Γ — граф переходов, определяющий функционирование автомата.

Каждому нетерминалу $\psi \in \tilde{V}_н$ грамматики G^a соответствует вершина графа Γ, помеченная символом ψ; в графе есть еще одна вершина, помеченная символом заключительного состояния φ. Единственному начальному состоянию σ автомата A соответствует вершина, помеченная аксиомой σ. Далее, для каждой незаключительной продукции $\psi \to a\tau$ проводится ребро, помеченное терминальным символом $a \in \mathfrak{A}$, из вершины ψ в вершину τ; для каждой заключительной продукции $\psi \to a$ проводится ребро a из вершины ψ в вершину φ. Автомат функционирует следующим образом: на вход последовательно (слева направо) подаются символы цепочки $x = a_{i_1}a_{i_2} \ldots a_{i_k} \in F(\mathfrak{A})$. Движение по графу начинается с вершины σ, что соответствует начальному состоянию автомата A, в котором он воспринимает символ a_{i_1}. Затем осуществляется переход из вершины σ по любому из ребер a_{i_1} к некоторой вершине ψ_1, после чего автомат воспринимает очередной символ a_{i_2} и т. д. Если на некотором шаге осуществляется переход в вершину ψ, у которой отсутствует ребро с пометкой, соответствующей очередному символу на входе, то автомат останавливается безрезультатно (аналог тупикового вывода в грамматике). В противном случае входная цепочка $x \in F(\mathfrak{A})$ переведет автомат A из начального состояния σ в некоторое состояние $\tau \in S$.

Путь в графе Γ, соответствующий такому переходу, называется **путем из вершины σ в вершину τ, несущим цепочку x**. Цепочка x воспринимается автоматом A, если для нее в графе Γ существует по крайней мере один путь из начальной вершины σ в заключительную вершину φ, несущий данную цепочку. Множество всех цепочек, воспринимаемых автоматов A, образует **язык $L(A)$, представимый данным автоматом.**

Очевидно, цепочка x воспринимается автоматом A тогда и только тогда, когда она выводима в соответствующей автоматной грамматике G^a так, что $L(A) = L(G^a)$. Например, универсальный язык над алфавитом $\mathfrak{A} = \{a, b\}$, порожденный приведенной выше грамматикой со схемой (5.28), представим автоматом с графом, приведенным на рис. 27, а язык, порожденный грамматикой со схемой (5.33), представим автоматом с графом переходов, приведенным на рис. 28.

Детерминированным автоматом называется автомат $A^д = (\mathfrak{A}, S, \sigma, \Omega, \Gamma)$ с множеством заключительных состояний $\Omega \subseteq S$, удовлетворяющий следующим условиям:

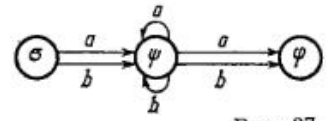


Рис. 27.

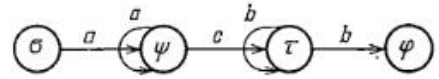


Рис. 28.

1) из каждой вершины графа Γ может исходить не более одного ребра, помеченного символом $a \in \mathfrak{A}$ **(условие однозначности)**;

2) из каждой вершины графа Γ исходит ребро, помеченное символом a, для любого $a \in \mathfrak{A}$ **(условие полной определенности)**;

3) каждая вершина ψ графа Γ достижима из начальной вершины σ, т. е. существует хотя бы один путь, связывающий вершины σ и ψ **(условие связности)**.

Из условий 1 и 2 следует, в частности, что произвольная цепочка $x \in F(\mathfrak{A})$ переводит автомат $A^д$ из начального состояния σ в некоторое состояние $\psi \in S$, причем в графе Γ существует лишь один путь из σ в ψ, несущий данную цепочку. Цепочка воспринимается автоматом $A^д$, если она переводит автомат из начального состояния σ в одно из заключительных состояний, принадлежащих множеству Ω. Множество всех цепочек, воспринимаемых автоматом $A^д$, образует язык $L(A^д)$, представимый данным автоматом. Такие языки называются **регулярными языками** или **событиями** [24]. Если множество Ω пусто, то автомат $A^д$ по определению представляет пустой язык.

Приведенные выше примеры автоматов с графами переходов (см. рис. 27 и 28), очевидно, недетерминированы. Так, граф, представленный на рис. 27, не удовлетворяет условию однозначности (из вершины ψ исходят два ребра, помеченных одним и тем же символом a); граф, представленный на рис. 28, не удовлетворяет ни условию однозначности (из вершины τ исходят два ребра,

помеченных символом b), ни условию полной определенности (отсутствуют ребра с пометками b и c, исходящие из вершины σ). Тем не менее автомату с графом, приведенным на рис. 27, эквивалентен детерминированный автомат[3] с заключительной вершиной ψ и графом переходов, представленным на рис. 29.

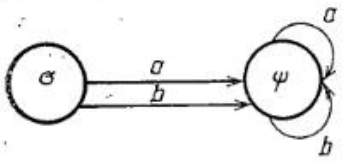


Рис. 29.

Автомату с графом, представленным на рис. 28, эквивалентен детерминированный автомат с заключительной вершиной τ и графом переходов, приведенным на рис. 30 (α — тупиковое состояние). Следовательно, языки, представимые указанными автоматами (см. рис. 27 и 28), регулярны.

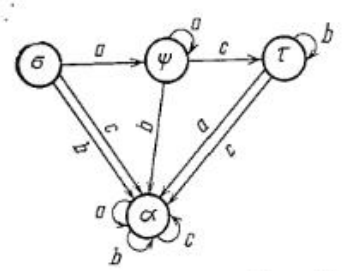


Рис. 30.

Примером регулярного языка является также язык

$$\{a^n \mid n = 1, 2, \ldots\} \cup \{b^m \mid m = 1, 2, \ldots\} \tag{5.34}$$

над алфавитом $\mathfrak{A} = \{a, b\}$, состоящий из любых цепочек символов a и любых цепочек символов b. Этот язык представим детерминированным автоматом с множеством заключительных состояний $\{\psi, \tau\}$, граф переходов которого представлен на рис. 31.

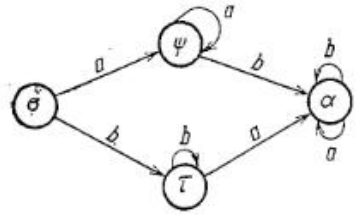


Рис. 31.

Известно [100], что каждому недетерминированному конечному автомату A эквивалентен некоторый детерминированный автомат $A^д$ (возможно, с бо́льшим числом состояний), более того, существует алгоритм процесса детерминизации — построения по автомату A эквивалентного ему детерминированного автомата $A^д$.

Таким образом, несмотря на то что автомат A, ассоциированный с грамматикой G^a, вообще говоря, недетерминированный, существует эквивалентный ему детерминированный автомат $A^д$.

Из приведенных выше рассуждений следует такая теорема[4].

[3] Автоматы эквивалентны, если они представляют один и тот же язык.

[4] Здесь и далее рассматриваются регулярные языки без пустого слова. Аналогичные результаты получены и в случае, когда $e \in L$. В частности, язык L порождается приведенной грамматикой, схема которой содержит наряду с автоматными продукциями продукцию $\sigma \to e$ (σ — аксиома грамматики). Этот язык представим соответствующим детерминированным автоматом, начальное состояние которого принадлежит множеству его заключительных состояний Q.

Теорема 5.10. *Множество всех языков, порожденных автоматными грамматиками, совпадает с множеством $\mathfrak{S}$ всех регулярных языков, представимых конечными детерминированными автоматами. Множество $\mathfrak{S}$ образует собственное подмножество множества всех кс-языков, более того, $\mathfrak{S}$ — собственное подмножество множества всех линейных языков.*

Действительно, рассмотрим язык

$$L = \{a^n c a^n \mid n = 1, 2, \ldots\}, \tag{5.35}$$

порожденный линейной грамматикой со схемой $\sigma \to a\sigma a$, $\sigma \to c$. Покажем, что не существует автоматной грамматики, порождающей данный язык.

Предположим противное. Пусть $G^a = (\{a, c\}, V_н, \sigma, P)$ — некоторая автоматная грамматика, порождающая язык L (5.35). Допустим, что r — число символов, входящих в нетерминальный алфавит $V_н$ данной грамматики. Выберем цепочку $x = a^m c a^m \in L$, где $m > r$. Тогда для любого вывода цепочки x в грамматике G^a можно указать нетерминал $\psi \in V_н$, встречающийся в данном выводе дважды:

$$\sigma \overset{*}{\Rightarrow} a^k\psi \overset{*}{\Rightarrow} a^{k+s}\psi \overset{*}{\Rightarrow} a^m c a^m, \tag{5.36}$$

где $k + s \leqslant m$. Это означает, что в грамматике G^a из символа ψ выводимы цепочки $\psi \overset{*}{\Rightarrow} a^s\psi$ и $\psi \overset{*}{\Rightarrow} a^{m-(k+s)}ca^m$. Устранив в выводе (5.36) продукции, примененные на выделенном фрагменте, получим вывод $\sigma \overset{*}{\Rightarrow} a^k\psi \overset{*}{\Rightarrow} a^{m-s}ca^m \in L$ в грамматике G^a. Следовательно, $L(G^a) \neq L$, что противоречит первоначальному предположению. Таким образом, язык L нерегулярный и справедлива следующая теорема.

Теорема 5.11. *Множество $\mathfrak{S}$ всех регулярных языков является собственным подмножеством множества всех линейных языков.*

Примером нерегулярного языка служит множество всех непустых цепочек в алфавите $\mathfrak{A} = \{a_1, a_2, \ldots, a_m\}$, длины которых являются точными квадратами [24]. Нетрудно показать, что множество $\mathfrak{S}$ всех регулярных языков над алфавитом $\mathfrak{A}$ замкнуто относительно теоретико-множественных операций $\cup$, $\cap$, $\neg$ (см. § 3.1, пример 5).

Действительно, пусть $L_1, L_2 \in \mathfrak{S}$ — регулярные языки, порожденные автоматными грамматиками соответственно $G_1^a = (\mathfrak{A}, V_н^1, \sigma_1, P_1)$ и $G_2^a = (\mathfrak{A}, V_н^2, \sigma_2, P_2)$ с непересекающимися нетерминальными алфавитами $V_н^1 \cap V_н^2 = \varnothing$. Построим автоматную грамматику $G^a = (\mathfrak{A}, V_н, \sigma, P)$, где σ — дополнительный нетерминал, не входящий в $V_н^1 \cup V_н^2$, $V_н = V_н^1 \cup V_н^2 \cup \{\sigma\}$; P содержит все продукции, входящие в $P_1 \cup P_2$, а также продукции $\sigma \to \sigma_1$ и $\sigma \to \sigma_2$. Очевидно, язык $L(G^a)$

регулярен, $L(G^a) = L_1 \cup L_2$, так что согласно теореме (5.10) объединение регулярных языков также регулярно.

Пусть $L \in \mathfrak{S}$ — произвольный регулярный язык. Рассмотрим язык $\neg L = F(\mathfrak{A}) \setminus L$ — дополнение языка L до свободной полугруппы $F(\mathfrak{A})$. Покажем, что язык $\neg L$ также регулярен. По определению из регулярности языка L следует существование автомата $A^{\text{д}} = (\mathfrak{A}, S, \sigma, \Omega, \Gamma)$, представляющего данный язык $L(A^{\text{д}}) = L$. Тогда автомат $B^{\text{д}} = (\mathfrak{A}, S, \sigma, \overline{\Omega}, \Gamma)$, где $\overline{\Omega} = S \setminus \Omega$, также детерминирован. Следовательно, для любой цепочки $x \in L$ в графе Γ существует единственный путь, несущий данную цепочку из начальной вершины σ в некоторую вершину $\psi \in \Omega$. Но тогда $\psi \overline{\in} \overline{\Omega}$ и поэтому $x \overline{\in} L(B^{\text{д}})$. С другой стороны, любая цепочка $y \in \neg L$ не воспринимается автоматом $A^{\text{д}}$. Это означает, что в графе Γ найдется единственный путь, несущий цепочку y из σ в некоторую вершину $\tau \overline{\in} \Omega$. Но тогда $\tau \in \overline{\Omega}$ и поэтому $y \in L(B^{\text{д}})$. Отсюда следует, что язык $\neg L$ представим детерминированным автоматом $B^{\text{д}}$, $L(B^{\text{д}}) = \neg L$. Таким образом, язык $\neg L$ регулярен и множество $\mathfrak{S}$ замкнуто относительно операции дополнения.

Замкнутость множества $\mathfrak{S}$ относительно объединения и дополнения влечет за собой замкнутость данного множества относительно пересечения, так как $L_1 \cap L_2 = \neg(\neg L_1 \cup \neg L_2)$.

Таким образом, справедлива следующая теорема.

Теорема 5.12. *Множество $\mathfrak{S}$ всех регулярных языков с определенными на нем теоретико-множественными операциями $\neg$, $\cup$, $\cap$ образует булеву алгебру.*

В гл. 3 (§ 3.1, пример 5) рассматривалась полугруппа $F(\mathfrak{A})$ с операцией конкатенации (соединения) слов. Операция конкатенации позволяет ввести такие важные операции над языками, как умножение языков и итерация [19].

Под **умножением языков** L_1 и L_2 понимается язык $L = L_1 \cdot L_2$, состоящий из всевозможных конкатенаций xy слов $x \in L_1$ и $y \in L_2$, т. е. $L = \{xy \mid (x, y) \in L_1 \times L_2\}$, где $L_1 \times L_2$ — декартово произведение языков (см. § 1.2). Из ассоциативности конкатенаций следует ассоциативность операции умножения языков, так что последняя относится к полугрупповым операциям. В то же время умножение языков не коммутативно, $L_1 \cdot L_2 \neq L_2 \cdot L_1$, кроме того $L\varnothing = \varnothing L = \varnothing$ для любого $L \in \mathfrak{S}$. С помощью операции умножения можно возводить данный язык L в степень:

$$L^1 = L,\ L^2 = L \cdot L,\ L^3 = L \cdot L \cdot L,\ \ldots,\ L^n = \underbrace{L \cdot L \cdot \ldots \cdot L}_{n \text{ раз}}.$$

Объединение всех последовательных степеней языка L, $L^* = \bigcup_{n=1}^{\infty} L^n$, называется **итерацией** L^* языка L. Можно показать, что множество $\mathfrak{S}$ замкнуто относительно операций умножения и итерации[5].

Действительно, пусть языки L_1, L_2 регулярны. Тогда найдутся автоматные грамматики $G_1^a = (\mathfrak{A}, V_{\text{н}}^1, \sigma_1, P_1)$ и $G_2^a = (\mathfrak{A}, V_{\text{н}}^2, \sigma_2, P_2)$, порождающие соответственно языки L_1, L_2, причем предполагается, что нетерминальные алфавиты этих грамматик не пересекаются, $V_{\text{н}}^1 \cap V_{\text{н}}^2 = \varnothing$. Рассмотрим автоматную грамматику $G^a = (\mathfrak{A}, V_{\text{н}}, \sigma_1, P)$, где σ_1 — аксиома грамматики G_1^a; $V_{\text{н}} = V_{\text{н}}^1 \cup V_{\text{н}}^2$; P состоит из продукций, входящих в схемы P_1 и P_2, причем все заключительные продукции вида $\psi \to a$ грамматики G_1^a заменяются продукциями $\psi \to a\sigma_2$ (σ_2 — аксиома грамматики G_2^a). Нетрудно видеть, что $L(G^a) = L_1 \cdot L_2$, откуда следует регулярность языка $L_1 \cdot L_2$

Для доказательства замкнутости множества $\mathfrak{S}$ относительно итерации достаточно по грамматике $G^a = (\mathfrak{A}, V_{\text{н}}, \sigma, P)$, порождающей произвольный регулярный язык $L(G) = L \in \mathfrak{S}$, построить грамматику $G_1^a = (\mathfrak{A}, V_{\text{н}}, \sigma, P^1)$, в схему которой входят все продукции схемы P и продукции вида $\psi \to a\sigma$, если $\psi \to a \in P$. Очевидно, $L(G_1^a) = L^*$, так что язык L^* регулярен, откуда и следует замкнутость множества $\mathfrak{S}$ относительно итерации.

Множество $\mathfrak{S}$ всех регулярных языков с определенными операциями объединения, умножения и итерации образует **алгебру регулярных событий**, играющую важную роль в теории конечных автоматов и в автоматных грамматиках [19, 24]. В частности, существуют алгоритмы, позволяющие по любому конечному автомату (автоматной грамматике) построить для языка, представимого данным автоматом (порожденного грамматикой), выражение алгебры $\mathfrak{S}$, составленной из элементарных однобуквенных языков $\{a\}$ (для любого $a \in A$) и операций алгебры $\mathfrak{S}$ (проблема анализа конечных автоматов). С другой стороны, можно для любого такого выражения сконструировать автомат, представляющий соответствующий язык (проблема синтеза конечных автоматов). Это означает, что множество элементарных языков образует базис алгебры $\mathfrak{S}$.

Одной из основных проблем в алгебре $\mathfrak{S}$ является проблема тождественных преобразований формул с целью их оптимизации по определенным критериям [7, 23, 94, 99, 137, 138]. К основным тождествам алгебры $\mathfrak{S}$, играющим в ней роль аксиом, относятся

$$(\alpha \cdot \beta)\gamma = \alpha(\beta \cdot \gamma), \tag{5.37}$$

$$(\alpha \cup \beta) \cup \gamma = \alpha \cup (\beta \cup \gamma), \tag{5.38}$$

$$\alpha \cup \beta = \beta \cup \alpha, \tag{5.39}$$

$$\alpha \cup \alpha = \alpha, \tag{5.40}$$

[5] Иногда итерация определяется как объединение L^*, включающее в себя нулевую степень языка L, $L^0 = e$.

$$(\alpha \cup \beta)\gamma = \alpha\gamma \cup \beta\gamma, \tag{5.41}$$

$$\alpha(\beta \cup \gamma) = \alpha\beta \cup \alpha\gamma, \tag{5.42}$$

$$\alpha^* = \alpha^*\alpha \cup \alpha. \tag{5.43}$$

(Здесь и далее α, β, γ, δ — произвольные выражения алгебры $\mathfrak{S}$.) Если скобок нет, то операции выполняются в следующем порядке: итерации, умножения, объединения. Используются также правила, позволяющие из аксиом получать новые тождества — теоремы в данном исчислении:

1) R_1 — правила замены; пусть $\alpha = \beta$ и $\gamma = \delta$ — произвольные тождества в алгебре $\mathfrak{S}$, причем α — подформула, входящая в γ; тогда выводимы тождества $\gamma' = \gamma$ и $\gamma' = \delta$, где γ' — формула, полученная в результате замены вхождения α в γ вхождением β;

2) R_2 — решение уравнений; из тождества $\alpha = \alpha\beta \cup \gamma$ выводимо тождество $\alpha = \gamma\beta^* \cup \gamma$.

Выводом в данном исчислении называется конечная последовательность тождеств, каждое из которых либо аксиома, либо следует из предыдущих тождеств на основании приведенных выше правил вывода. Заключительное тождество вывода называется **теоремой данного исчисления.** Как и в других исчислениях, основными проблемами в данном исчислении являются непротиворечивость (любая теорема должна быть тождеством, выполнимым в алгебее $\mathfrak{S}$) и полнота (любое выполнимое в $\mathfrak{S}$ тождество является теоремой в данном исчислении, т. е. может быть выведено из аксиом (5.37)—(5.43) с помощью правил R_1, R_2).

Нетрудно показать, что построенное исчисление непротиворечиво. Каждая из аксиом (5.37)—(5.43) является тождеством, выполнимым в алгебре $\mathfrak{S}$. Точно так же выполнимы все тождества, полученные из аксиом в результате применения правила замены. Покажем, что правило R_2 сохраняет выполнимость тождеств в алгебре $\mathfrak{S}$.

Пусть $\alpha = \alpha\beta \cup \gamma$ — тождество, выполнимое в $\mathfrak{S}$. Применяя последовательно правило замены, получаем совокупность тождеств, выполнимых в $\mathfrak{S}$:

$$\alpha = \alpha\beta^{k+1} \cup \gamma\beta^k \cup \cdots \cup \gamma\beta \cup \gamma, \tag{5.44}$$

где $k = 0, 1, 2, \ldots$ Следовательно, любая цепочка, входящая в $\gamma\beta^* \cup \gamma$, принадлежит и выражению $\alpha : \gamma\beta^* \cup \gamma \subseteq \alpha$. С другой стороны, справедливо и обратное включение $\alpha \subseteq \gamma\beta^* \cup \gamma$. Действительно, выберем произвольную цепочку $x \in \alpha$, где $|x| = r$ — длина цепочки x. Тогда в совокупности (5.44) найдется тождество $\alpha = \alpha\beta^{r+1} \cup \gamma\beta^r \cup \gamma\beta^{r-1} \cdots \cup \gamma\beta \cup \gamma$, из которого следует, что $x \in \alpha\beta^{r+1}$. Это в свою очередь означает, что $x \in \gamma\beta^r \cup \cdots \cup \gamma\beta \cup \gamma$. Таким образом, $\alpha = \gamma\beta^* \cup \gamma$ — тождество, выполнимое в $\mathfrak{S}$, так что правило R_2 сохраняет выполнимость тождеств в алгебре $\mathfrak{S}$. Из приведенных рассуждений следует теорема.

Теорема 5.13. *Аксиоматическая система* (5.37)—(5.43) *с правилами вывода* R_1, R_2 *непротиворечива.*

В работе [99] доказана полнота данного исчисления. В то же время для алгебры $\mathfrak{S}$ не существует конечной полной системы аксиом, если правило замены — единственное правило вывода [94].

В теории формальных языков и ее приложениях важное место занимают **преобразователи,** осуществляющие перевод (трансляцию) цепочек с одного языка на другой (см. гл. 6).

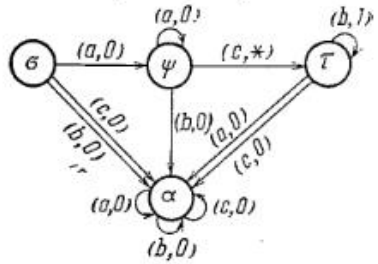


Рис. 32.

Пусть $A^{\text{п}} = (\mathfrak{A}, S, \sigma, \Omega, \Gamma)$ — конечный детерминированный автомат. Снабдим его выходным алфавитом $\mathfrak{B} = \{b_1, \ldots, b_r\}$, $r \geqslant 2$, и потребуем, чтобы автомат на каждом такте работы при подаче входного символа $a \in \mathfrak{A}$ одновременно выдавал некоторый выходной символ $b \in \mathfrak{B}$. Функционирование такого автомата может быть описано графом Γ', который совпадает (с точностью до пометок над ребрами) с графом Γ. Если в графе Γ из вершины ψ в вершину τ ведет ребро, помеченное символом $a \in \mathfrak{A}$, то в графе Γ' из вершины ψ в вершину τ ведет ребро, помеченное парой (a, b), где $b \in \mathfrak{B}$ — выходной символ, который выдает данный автомат, находясь в состоянии ψ при подаче на вход символа a. Описанный таким образом автомат $A^{\text{п}} = (\mathfrak{A}, \mathfrak{B}, S, \sigma, \Omega, \Gamma')$ называется **конечным преобразователем.**

Конечный преобразователь $A^{\text{п}}$ каждой цепочке $x \in F(\mathfrak{A})$, воспринимаемой автоматом $A^{\text{п}}$, ставит в соответствие выходную цепочку $A^{\text{п}}(x) \in F(\mathfrak{B})$, порожденную автоматом $A^{\text{п}}$. Множество всех таких цепочек образует язык, порожденный автоматом $A^{\text{п}}$:

$$L(A^{\text{п}}) = \{A^{\text{п}}(x) \mid x \in L(A^{\text{п}})\},$$

где $L(A^{\text{п}}) = L(A^{\text{п}})$ — язык, представимый автоматом $A^{\text{п}}$. Нетрудно показать, что язык $L(A^{\text{п}})$, порожденный конечным преобразователем $A^{\text{п}}$, регулярен. Для этого достаточно в пометках над ребрами устранить символы входного алфавита и детерминизировать полученный автомат, если он окажется недетерминированным. Например, конечным преобразователем является автомат с заключительной вершиной τ и выходным алфавитом $\{0, \times, 1\}$, заданный графом, приведенным на рис. 32. Этот автомат, воспринимая цепочку $a^n c b^m$, одновременно порождает цепочку $0^n \times 1^m$, так что язык $\{a^n c b^m \mid n,$

$m = 1, 2, \ldots\}$ представим, а язык $\{0^n \barwedge 1^m \mid n, m = 1, 2, \ldots\}$ порождаем данным автоматом.

Заметим, что функционирование конечного преобразователя можно определить также двумя функциями — функцией переходов φ: $\mathfrak{A} \times S \to S$ и функцией выходов $\Delta : \mathfrak{A} \times S \to \mathfrak{B}$, которые обычно задаются таблицами соответственно переходов (табл. 7) и выходов (табл. 8) [24].

Таблица 7

φ	σ	ψ	τ	α
a	ψ	ψ	a	α
b	a	a	τ	α
c	a	τ	a	α

Таблица 8

Δ	σ	ψ	τ	α
a	0	0	0	0
b	0	0	1	0
c	0	$\barwedge$	0	$\cdot$ 0

§ 5.5. Алгебры контекстно-свободных языков

Развитие теории кс-языков тесно связано с применением алгебраических методов [19, 48, 70, 95]. В частности, в теории кс-языков изучаются алгебраические свойства некоторых важных операций над кс-языками, находятся в соответствующих алгебрах системы образующих, устанавливаются критерии полноты, выясняются некоторые свойства структуры подалгебр (см. § 3.1—3.3). При этом в сигнатуру алгебр кс-языков обычно включаются операции, относительно которых множество $\mathfrak{K}$ всех кс-языков замкнуто (множество $\mathfrak{K}$ замкнуто относительно n-арной операции Ω, если $\Omega(L_1, L_2, \ldots, L_n) \in \mathfrak{K}$ для любых $L_1, \ldots, L_n \in \mathfrak{K}$). К операциям, относительно которых множество $\mathfrak{K}$ всех кс-языков замкнуто, относится прежде всего **суперпозиция**

$$S^{n+1}\begin{pmatrix} q_1, & \ldots, & q_n \\ L_1, & \ldots, & L_n \end{pmatrix} L$$

(см. § 3.1, примеры 5—7).

Пусть L_i, $L \in \mathfrak{K}$ — произвольные кс-языки, порожденные кс-грамматиками $G_i = (\mathfrak{A}, V_н^i, \sigma_i, P_i)$, $G = (\mathfrak{A}, V_н, \sigma, P)$ соответственно $(i = 1, 2, \ldots, n)$; причем нетерминальные алфавиты $V_н^1, V_н^2, \ldots, V_н^n, V_н$ — попарно не пересекаются, $\mathfrak{A} = \{a_1, \ldots, a_m\}$, где $m \geqslant n$. Каждому из выделенных терминальных символов $q_1, q_2, \ldots, q_n \in \mathfrak{A}$ сопоставим соответственно дополнительный нетерминал $\bar{q}_1, \bar{q}_2, \ldots, \bar{q}_n$ так, что

$\bar{q}_i \in V_н \cup \bigcup_{i=1}^{n} V_н^i$ при любом $i = 1, 2, \ldots, n$. Пусть, далее, P^0 включает в себя все продукции

$$\bar{q}_i \to \sigma_i \tag{5.45}$$

$(i = 1, 2, \ldots, n)$, а также все продукции схемы P, в правых частях которых все вхождения терминалов $q_1, \ldots, q_n$ заменены нетерминалами $\bar{q}_1, \ldots, \bar{q}_n$. Тогда язык

$$\bar{L} = S^{n+1}\begin{pmatrix} q_1, & \ldots, & q_n \\ L_1, & \ldots, & L_n \end{pmatrix} L$$

порождается кс-грамматикой $\bar{G} = (\mathfrak{A}, \bar{V}_н, \sigma, \bar{P})$, где $\bar{V}_н = V_н \cup (\bigcup_{i=1}^{n} V_н^i) \cup \{\bar{q}_1, \ldots, \bar{q}_n\}$; σ — аксиома грамматики G; $\bar{P} = P^0 \cup (\bigcup_{i=1}^{n} P_i)$.

Действительно, выберем произвольную цепочку $x \in \bar{L}$. По определению суперпозиции цепочку x можно представить в виде $x = x_1 y_1 x_2 y_2 \ldots x_n y_n x_{n+1}$, где $y_1 \in L_1, \ldots, y_n \in L_n$; $x' = x_1 q_1 \ldots \ldots x_n q_n x_{n+1} \in L$. Это означает, что для каждой цепочки y_i существует вывод

$$\sigma_i \overset{*}{\underset{G_i}{\Rightarrow}} y_i \tag{5.46}$$

в соответствующей грамматике G_i при любом $i = 1, 2, \ldots, n$; кроме того, для цепочки x' существует вывод

$$\sigma \overset{*}{\underset{G}{\Rightarrow}} x' \tag{5.47}$$

в грамматике G. Следовательно, используя продукции, входящие в P^0, по выводу (5.47) для цепочки $x'' = x_1 \bar{q}_1 x_2 \bar{q}_2 \ldots x_n \bar{q}_n x_{n+1}$ можно построить вывод $\sigma \overset{*}{\Rightarrow} x''$ в грамматике $\bar{G}$. Затем, применяя продукции (5.45), получаем вывод $\sigma \overset{*}{\Rightarrow} x_1 \sigma_1 x_2 \sigma_2 \ldots x_n \sigma_n x_{n+1}$ в грамматике $\bar{G}$. Наконец, используя (5.46), получаем вывод цепочки x в грамматике $\bar{G}$:

$$\sigma \overset{*}{\Rightarrow} x_1 \sigma_1 x_2 \sigma_2 \ldots x_n \sigma_n x_{n+1} \overset{*}{\Rightarrow} x_1 y_1 x_2 \sigma_2 \ldots x_n \sigma_n x_{n+1} \overset{*}{\Rightarrow} x_1 y_1 x_2 y_2 x_3 \sigma_3 \ldots$$
$$\ldots x_n \sigma_n x_{n+1} \overset{*}{\Rightarrow} \ldots \overset{*}{\Rightarrow} x_1 y_1 x_2 y_2 \ldots x_n y_n x_{n+1} = x.$$

Таким образом, справедливо включение

$$\bar{L} \subseteq L(\bar{G}). \tag{5.48}$$

Докажем теперь справедливость обратного включения

$$L(\overline{G}) \subseteq \overline{L}. \tag{5.49}$$

Выберем произвольную цепочку $z \in L(\overline{G})$ и рассмотрим некоторый вывод

$$\sigma \overset{*}{\Rightarrow} z \tag{5.50}$$

этой цепочки в грамматике $\overline{G}$. Поскольку нетерминальные алфавиты грамматик G_i, G попарно не пересекаются, вывод (5.50) можно перестроить так, что вначале применяются продукции из P^0, соответствующие продукциям грамматики G, затем — продукции вида (5.45) и, наконец, — группы продукций, входящие в схемы P_i $(i = 1, 2, \ldots, n)$ в соответствующем порядке. Следовательно, существуют цепочки $z' = z_1 q_1 z_2 q_2 \ldots z_n q_n z_{n+1} \in L$, $u_1 \in L_1$, $u_2 \in L_2$, $\ldots$, $u_n \in L_n$ такие, что $z = z_1 u_1 z_2 u_2 \ldots z_n u_n z_{n+1}$. Это означает, что $z \in \overline{L}$ и справедливо включение (5.49). Из включений (5.48), (5.49) вытекает, что $\overline{L} = L(\overline{G})$. Таким образом, справедливо следующее утверждение.

Теорема 5.14. *Множество $\mathfrak{K}$ всех кс-языков замкнуто относительно суперпозиции.*

Анализируя проведенное доказательство, замечаем, что если грамматики $G_1, G_2, \ldots, G_n, G$ автоматны, то грамматику $\overline{G}$ также нетрудно свести к эквивалентной автоматной грамматике, исключая известным способом (см. § 5.3) продукции вида (5.45). Поэтому согласно теореме 5.10 справедливо такое следствие.

Следствие 1. Множество $\mathfrak{S}$ всех регулярных языков замкнуто относительно суперпозиции.

Пусть $\mathfrak{A} = \{a_1, \ldots, a_m\}$, $m \geqslant 2$,— произвольный терминальный алфавит. Выделим в алфавите $\mathfrak{A}$ символы $a, b \in \mathfrak{A}$ и рассмотрим языки $\bigcup(\mathfrak{A}) = \{a, b\}$ (состоящий из двух однобуквенных цепочек a и b), $\Pi(\mathfrak{A}) = \{ab\}$ (содержащий единственную цепочку ab) и a^* (представляющий собой итерацию элементарного языка $\{a\}$). Очевидно, что языки $\bigcup(\mathfrak{A})$, $\Pi(\mathfrak{A})$ и a^* являются кс-языками. Сопоставим теперь выделенным символам $a, b \in \mathfrak{A}$ соответственно произвольные кс-языки L_1, L_2. Выполнив суперпозиции, получим

$$S^3\begin{pmatrix} a, & b \\ L_1, & L_2 \end{pmatrix} \bigcup(\mathfrak{A})\Big) = L_1 \cup L_2, \quad S^3\begin{pmatrix} a, & b \\ L_1, & L_2 \end{pmatrix} \Pi(\mathfrak{A})\Big) = L_1 L_2,$$

$$S^2\begin{pmatrix} a \\ L_1 \end{pmatrix} a^*\Big) = L_1^*.$$

Согласно доказанной теореме приведенные суперпозиции порождают кс-языки.

Произвольность выбора языков $L_1, L_2 \in \mathfrak{K}$ позволяет сделать следующий вывод.

Следствие 2. Множество $\mathfrak{K}$ всех кс-языков замкнуто относительно операций объединения, умножения и итерации.

В дальнейшем используется замкнутость множества $\mathfrak{K}$ также относительно унарной операции **исключения пустого слова** e,

$$\overline{L} = \begin{cases} L, & \text{если } e \overline{\in} L, \\ L \setminus e, & \text{если } e \in L, \end{cases}$$

для любого $L \in \mathfrak{K}$ (см. теорему 5.5), и относительно **операции обращения**, т. е. $L^{-1} = \{x^{-1} \mid x \in L\} \in \mathfrak{K}$ для любого $L \in \mathfrak{K}$ (цепочка $x^{-1} = a_{i_k} a_{i_{k-1}} \ldots a_{i_1}$ является обратной цепочке $x = a_{i_1} a_{i_2} \ldots a_{i_{k-1}} a_{i_k}$).

Действительно, если L — кс-язык, порожденный грамматикой G со схемой $\{\psi_i \to W_i\}$, то язык L^{-1} порождается кс-грамматикой G^{-1} со схемой $\{\psi_i \to W_i^{-1}\}$. Несмотря на замкнутость множества $\mathfrak{K}$ относительно объединения, можно показать, что другие теоретико-множественные операции (пересечение, дополнение) уже выводят из класса кс-языков. В качестве примера рассмотрим языки

$$L_1 = \{a^n b^m a^n \mid n, m = 1, 2, \ldots\}, \quad L_2 = \{a^n b^n a^m \mid n, m = 1, 2, \ldots\},$$

порождаемые кс-грамматиками соответственно G_1 со схемой

$$\left.\begin{array}{l} \sigma \to a\tau a, \\ \sigma \to a\psi a, \\ \tau \to a\tau a, \\ \tau \to a\psi a, \\ \psi \to b\psi, \\ \psi \to b \end{array}\right\} \tag{5.51}$$

и G_2 со схемой

$$\left.\begin{array}{l} \sigma \to \tau a, \\ \sigma \to \psi a, \\ \tau \to \tau a, \\ \tau \to \psi a, \\ \psi \to a\psi b, \\ \psi \to ab, \end{array}\right\} \tag{5.52}$$

так, что $L_1 = L(G_1)$ и $L_2 = L(G_2)$. Следовательно, $L_1, L_2 \in \mathfrak{K}$; более того, языки L_1, L_2 линейны. В то же время $L_1 \cap L_2 = \{a^n b^n a^n \mid n = 1, 2, \ldots\} \overline{\in} \mathfrak{K}$ (см. § 5.3). Таким образом, множество $\mathfrak{K}$ всех кс-языков и множество $\mathfrak{L}$ всех линейных языков не замкнуты относительно пересечения. Из соотношения

$$L_1 \cap L_2 = \neg(\neg L_1 \cup \neg L_2) \tag{5.53}$$

следует, что операция дополнения выводит за пределы множества кс-языков (линейных), так как в противном случае согласно формуле (5.53) и следствию 2 доказанной выше теоремы множество $\mathfrak{K}$ (соответственно $\mathfrak{L}$) оказалось бы замкнутым относительно пересечения.

Заметим, однако, что множество $\mathfrak{K}$ замкнуто относительно **суженной операции пересечения.** Пересечение произвольного кс-языка с любым регулярным языком снова образует кс-язык [19]. Это следствие приведенной ниже теоремы подтверждает основополагающую роль регулярных множеств в общей теории кс-языков.

Рассмотрим произвольную кс-грамматику $G = (\mathfrak{A}, V_н, \sigma, P)$ и конечный преобразователь $T = (\mathfrak{A}, \mathfrak{B}, S, s_0, \Omega, \varphi, \Delta)$, где $\mathfrak{A}$ и $\mathfrak{B}$ — соответственно входной и выходной алфавиты; S — множество состояний; $s_0 \in S$ — начальное состояние; $\Omega \subseteq S$ — множество заключительных состояний; φ — функция переходов; Δ — функция выходов данного преобразователя (см. § 5.4). Пусть $L(G)$ — кс-язык, порожденный грамматикой G; $L(T)$ — язык, представимый преобразователем T.

Теорема 5.15. *Язык* $T(L) = \{T(x) \mid x \in L(G) \bigcap L(T)\}$, *состоящий из всех цепочек, порожденных преобразователем* T, *при подаче на вход преобразователя воспринимаемых им цепочек из* $L(\mathfrak{S})$ *является также кс-языком.*

Доказательство. Построим кс-грамматику $G' = (\mathfrak{B}, V'_н, \sigma', P')$, порождающую язык $T(L)$. Для этого рассмотрим объединенный алфавит $V = \mathfrak{A} \bigcup V_н$ грамматики G и множество $S = (s_0, s_1, \ldots, s_k)$ состояний преобразователя T. Аксиомой грамматики G' является новый нетерминал $\sigma' \overline{\in} V$, кроме него в нетерминальный алфавит $V'_н$ входят символы $\Delta_{(s_i, s_j)}$ для любых терминальных или нетерминальных символов $\Delta \in V$ грамматики G и пар состояний $s_i, s_j \in S$ преобразователя T.

Схема P' грамматики G' состоит из следующих трех групп продукции:

1) $\sigma' \to \sigma_{(s_0, q)}$ для любых $q \in \Omega$, где σ — аксиома грамматики G; s_0 и $q \in \Omega$ — соответственно начальное и заключительное состояния преобразователя T;

2) $\psi_{(s_i, s_j)} \to \overset{1}{\psi}_{(s_i, q_1)} \overset{2}{\psi}_{(q_1, q_2)} \ldots \overset{n}{\psi}_{(q_{n-1}, s_j)}$ для произвольной продукции $\psi \to \psi^1 \psi^2 \ldots \psi^n \in P$ грамматики G и любых состояний $s_i, s_j, q_1, \ldots, q_{n-1} \in S$ преобразователя T;

3) заключительные продукции $a_{(s_i, s_j)} \to b$ такие, что $\varphi(a, s_i) = s_j$ и $\Delta(a, s_i) = b$, где $a \in \mathfrak{A}$; $b \in \mathfrak{B}$; $s_i, s_j \in S$; φ и Δ — соответственно функции переходов и выходов преобразователя T.

Нетрудно видеть, что для всякой цепочки $x = a^1 a^2 \ldots a^r \in L(G)$ с помощью продукций, принадлежащих группам 1, 2, в грамматике G' выводима цепочка $x' = \overset{1}{a}_{(s_0, q_1)} \overset{2}{a}_{(q_1, q_2)} \ldots \overset{r}{a}_{(q_{r-1}, q)}$ при любых

$q_1, q_2, \ldots, q_{r-1} \in S$, где s_0 и $q \in \Omega$ — соответственно начальное и заключительное состояния преобразователя T. В то же время к цепочке x' применимы продукции группы 3, если $(s_0, q_1, \ldots, q_{r-1}, q)$ — путь в T, несущий цепочку x из начального состояния s_0 в заключительное $q \in \Omega$ так, что $\Delta(a^1, s_0) = b_1$, $\Delta(a^2, q_1) = b_2, \ldots, \Delta(a^r, q_{r-1}) = b_r$ и $b_1 b_2 \ldots b_r \in F(\mathfrak{B})$ — цепочка, порожденная преобразователем T при подаче на его вход цепочки $x \in L(G) \bigcap L(T)$. Таким образом, $L(G') = T(L)$. Теорема доказана.

Заметим, что число нетерминалов и продукций грамматики G' обычно избыточно. Пусть, например, даны кс-грамматика G_1 со схемой

$$\begin{cases} \sigma \to a\sigma a, \\ \sigma \to b, \end{cases}$$

порождающая язык

$$L(G_1) = \{a^n b a^n \mid n = 0, 1, 2, \ldots\}; \tag{5.54}$$

конечный преобразователь T_1 с начальным состоянием s и заключительным q, заданный графом (рис. 33) так, что $L(T_1) = \{a^n b F(\mathfrak{A}) \mid n = 1, 2, \ldots\}$ — язык, представимый в T_1; порожденный преобразователем T_1 язык $\tilde{L} = \{0^n \times F(\mathfrak{B}) \mid n = 1, 2, \ldots\}$, где $\mathfrak{A} = \{a, b\}$ и $\mathfrak{B} = \{0, \times, 1\}$ — соответственно входной и выходной алфавиты преобразователя T_1. Тогда грамматика $G' = (\mathfrak{B}, V'_н, \sigma_{(s, q)}, P')$ с нетерминальным алфавитом $V'_н = \{\sigma_{(s, q)}, a_{(s, s)}, a_{(q, q)}, b_{(s, q)}, b_{(q, q)}\}$, схема которой имеет вид

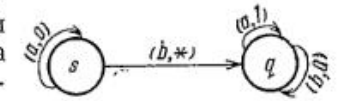


Рис. 33.

$$P' = \begin{cases} \sigma_{(s, q)} \to a_{(s, s)} \sigma_{(s, q)} a_{(q, q)}, \\ \sigma_{(s, q)} \to b_{(s, q)} \\ a_{(s, s)} \to 0, \\ b_{(s, q)} \to \times, \\ a_{(q, q)} \to 1, \end{cases}$$

порождает язык $L(G') = \{0^n \times 1^n \mid n = 1, 2, \ldots\}$, совпадающий с множеством всех цепочек, порожденных преобразователем T_1 при подаче на его вход цепочек из $L(G_1) \bigcap L(T_1)$.

Пусть $L(D) \in \mathfrak{S}$ — произвольный регулярный язык, представимый детерминированным автоматом D. Построим по автомату D конечный преобразователь T (представляющий язык $L(T) = L(D)$), функция выходов которого имеет вид $\Delta(a, s) = a$ для любых $(a, s) \in \mathfrak{A} \times S$, где $\mathfrak{A}$ — входной алфавит; S — множество состояний автомата D. Построенный таким образом преобразователь T копирует воспринимаемую им цепочку $T(x) = x$ для любого $x \in L(T)$.

Пусть $L(G) \in \mathfrak{K}$ — произвольный кс-язык, порожденный грамматикой G. Тогда $T(L) = L(G) \cap L(T) = L(G) \cap L(D)$ и из доказанной теоремы вытекает такое следствие.

Следствие 1. Пересечение $L(G) \cap L(D)$ принадлежит $\mathfrak{K}$ для любых $L(G) \in \mathfrak{K}$ и $L(D) \in \mathfrak{S}$, т. е. множество $\mathfrak{K}$ всех кс-языков замкнуто относительно суженной операции пересечения произвольных контекстно-свободного и регулярного языков.

Таким образом, можно отбрасывать «лишние» (не имеющие нужной формы) цепочки данного кс-языка, пересекая его с подходящим регулярным языком, при этом оставшиеся цепочки снова образуют кс-язык.

Левым (правым) делением $L \backslash\!\backslash u (L /\!/ u)$ языка L на одноэлементный язык $\{u\}$ [19] называется операция, порождающая новый язык:

$$L \backslash\!\backslash u = \{x \mid ux \in L\} \quad (L /\!/ u = \{x \mid xu \in L\}).$$

Левое деление удовлетворяет следующим свойствам:

$$\{u\} \backslash\!\backslash u = \{e\}, \tag{5.55}$$

где e — пустое слово;

$$(L_1 \cup L_2) \backslash\!\backslash u = (L_1 \backslash\!\backslash u) \cup (L_2 \backslash\!\backslash u), \tag{5.56}$$

$$(L_1 L_2) \backslash\!\backslash u = (L_1 \backslash\!\backslash u)\, L_2 \cup \xi(L_1)(L_2 \backslash\!\backslash u), \tag{5.57}$$

где

$$\xi(L_1) = \begin{cases} e, & \text{если } e \in L_1, \\ \varnothing, & \text{если } e \overline{\in} L_1; \end{cases} \quad \varnothing \text{ — пустой язык.}$$

Аналогичным свойствам удовлетворяет операция правого деления.

Следствие 2. Множество $\mathfrak{K}$ всех кс-языков замкнуто относительно операции левого (правого) деления.

Действительно, пусть $L \in \mathfrak{K}$ — произвольный кс-язык над алфавитом $\mathfrak{A}$ и $u = a_1 a_2 \ldots a_k$ — некоторая цепочка над этим же алфавитом. Покажем, что язык $L \backslash\!\backslash u$ также является кс-языком. Для этого рассмотрим конечный преобразователь T, в котором представим регулярный язык $M = (u\mathfrak{A}^*) \in \mathfrak{S}$, т. е. $L(T) = M$, причем, воспринимая цепочку $x \in M$, преобразователь T выдает символы e при подаче на вход символов начальной подцепочки u, копируя остальную часть цепочки x так, что $T(x) = x'$, где $x = ux'$ — произвольная цепочка из M. Подавая на вход преобразователя T цепочки из пересечения $L \cap M$, убеждаемся в том, что $L \backslash\!\backslash u = T(L)$. Но согласно доказанной выше теореме язык $T(L) = \{T(x) \mid x \in L \cap M\}$ является кс-языком.

Таким образом, множество $\mathfrak{K}$ замкнуто относительно операции левого деления. Замкнутость множества $\mathfrak{K}$ относительно операции правого деления следует из замкнутости этого множества по операциям обращения и левого деления.

В теории кс-языков важное место занимают рекурсивные процедуры, порождающие из конечных языков бесконечные [95]. Пусть

$$U^n = \{\psi_i ::= f_i(\psi_1, \psi_2, \ldots, \psi_n) \mid i = 1, 2, \ldots, n\} \tag{5.58}$$

является системой уравнений (см. § 5.3) с выделенным нетерминалом ψ_1, где $f_i(\psi_1, \psi_2, \ldots, \psi_n) = \psi_i \cup z_{i_1} \cup z_{i_2} \cup \cdots \cup z_{i_{k_i}}$ — такой полином, что каждая цепочка z_{i_j} $(1 \leqslant j \leqslant k_i)$ содержит хотя бы один нетерминал из алфавита $V_н = \{\psi_1, \ldots, \psi_n\}$ при любом $i = 1, 2, \ldots, n$. Рассмотрим языки $L_1, L_2, \ldots, L_n \subseteq F(\mathfrak{A})$ над терминальным алфавитом $\mathfrak{A}$. Аналогично рекурсивному процессу, изложенному в § 5.3, подставим языки $L_1, L_2, \ldots, L_n$ вместо переменных соответственно $\psi_1, \psi_2, \ldots, \psi_n$ в каждую из правых частей системы уравнений (5.58). В результате получим новые языки над алфавитом $\mathfrak{A}$:

$$\left.\begin{aligned} L_1^1 &= f_1(L_1, \ldots, L_n), \\ L_2^1 &= f_2(L_1, \ldots, L_n), \\ &\ldots\ldots\ldots \\ L_n^1 &= f_n(L_1, \ldots, L_n). \end{aligned}\right\} \tag{5.59}$$

Затем, вновь подставляя языки $L_1^1, L_2^1, \ldots, L_n^1$ (5.59) вместо переменных $\psi_1, \ldots, \psi_n$ в правые части уравнений (5.58), получаем

$$\begin{aligned} L_1^2 &= f_1(L_1^1, \ldots, L_n^1), \\ L_2^2 &= f_2(L_1^1, \ldots, L_n^1), \\ &\ldots\ldots\ldots \\ L_n^2 &= f_n(L_1^1, \ldots, L_n^1). \end{aligned}$$

Продолжив рекурсивно этот процесс, получим

$$\begin{aligned} L_1^k &= f_1(L_1^{k-1}, \ldots, L_n^{k-1}), \\ L_2^k &= f_2(L_1^{k-1}, \ldots, L_n^{k-1}), \\ &\ldots\ldots\ldots \\ L_n^k &= f_n(L_1^{k-1}, \ldots, L_n^{k-1}) \end{aligned}$$

при любом $k = 1, 2, \ldots$

Пусть $L_i^\infty = \bigcup_{k=1}^{\infty} L_i^k$ при любом $i = 1, 2, \ldots, n$. Операцию рекурсии, заданную системой уравнений U^n (5.58), можно рассматривать как $(n+1)$-арную операцию $R^{n+1}(L_1, L_2, \ldots, L_n; \tilde{L})$ над языками $L_1, \ldots, L_n, \tilde{L} \subseteq F(\mathfrak{A})$, где $\tilde{L}$ — язык, полученный в результате исключения всех нетерминальных символов, входящих

в правые части уравнений системы U^n. Операция рекурсии порождает новый язык $L_1^\infty = R^{n+1}(L_1, \ldots, L_n; \tilde{L})$, причем схема такого порождения определяется, как было показано выше, системой уравнений U^n.

Каждому кс-языку можно поставить в соответствие операцию рекурсии, порождающую данный язык из конечных языков. Например, рассмотренный выше (см. (5.54)) кс-язык $L(G_1) = \{a^n b a^n \mid n = 0, 1, 2, \ldots\}$ порождается операцией рекурсии $R^2(L_1; \tilde{L})$, схема которой определяется единственным уравнением $\{\psi :: = \psi \cup a\psi a\}$, где $L_1 = \{b\}$, $\tilde{L} = \{aa\}$. В то же время языки

$$L_1 = \{a^n b^m a^n \mid n, m = 1, 2, \ldots\}, \qquad L_2 = \{a^n b^n a^m \mid n, m = 1, 2, \ldots\},$$

которые порождаются кс-грамматиками соответственно со схемами (5.51), (5.52), могут быть порождены также операциями рекурсии $L_1 = R_1^4(\varnothing, \varnothing, \{b\}; \{aa, b\})$, $L_2 = R_2^4(\varnothing, \varnothing, \{ab\}; \{a, ab\})$, заданными системами уравнений

$$U_1^3 = \begin{cases} \psi_1 :: = \psi_1 \cup a\psi_2 a \cup a\psi_3 a, \\ \psi_2 :: = \psi_2 \cup a\psi_2 a \cup a\psi_3 a, \\ \psi_3 :: = \psi_3 \cup b\psi_3, \end{cases}$$

$$U_2^3 = \begin{cases} \psi_1 :: = \psi_1 \cup \psi_2 a \cup \psi_3 a, \\ \psi_2 :: = \psi_2 \cup \psi_2 a \cup \psi_3 a, \\ \psi_3 :: = \psi_3 \cup a\psi_3 b. \end{cases}$$

С другой стороны, любая рекурсия кс-языков образует снова кс-язык. Таким образом, множество $\mathfrak{K}$ замкнуто относительно операции рекурсии, причем совокупность всех конечных языков является системой образующих в алгебрах кс-языков, сигнатуры которых содержат всевозможные рекурсии.

Рассмотрим алгебру $\mathfrak{K}$ всех кс-языков над алфавитом $\mathfrak{A} = \{a_1, \ldots, a_n\}$, сигнатура которой содержит наряду с операциями $^+$ (исключение пустого слова), $\cup$ (объединение языков), $\cdot$ (умножение языков) также всевозможные суперпозиции и рекурсии [118]. Пусть $\mathfrak{K}_0 = \mathfrak{K} \setminus \{\{e\}, \varnothing\}$; $\mathfrak{C} \subseteq \mathfrak{K}_0$ — множества всех конечных языков над алфавитом $\mathfrak{A}$; $\mathfrak{C}^r \subseteq \mathfrak{C}$ — множество всех конечных языков L таких, что для цепочки $x \in L$ выполняется неравенство $|x| \leqslant r$, где $|x|$ — длина цепочки x, $r \geqslant 1$.

Заметим, что $\mathfrak{K}_0$ образует подалгебру алгебры $\mathfrak{K}$, а множество $\mathfrak{C}$ замкнуто по операциям $^+$, $\cup$, $\cdot$, S^{n+1}.

Справедлив следующий тривиальный критерий полноты в подалгебре $\mathfrak{K}_0$.

Теорема 5.16. *Система языков $\Sigma \subset \mathfrak{K}_0$ является системой образующих подалгебры $\mathfrak{K}_0$ тогда и только тогда, когда выполняется включение $\Sigma \supseteq \mathfrak{C}^1$, где $\mathfrak{C}^1$ — множество всех элементарных одно-буквенных языков $\{a_1\}, \ldots, \{a_m\}$.*

Необходимость сформулированного условия очевидна, так как множество $\mathfrak{K}_0 \setminus \{a\}$ является подалгеброй, максимальной относительно $\mathfrak{K}_0$ для каждого $a \in \mathfrak{A}$, а *достаточность* — так как система языков $\mathfrak{C}^1$ порождает подалгебру $\mathfrak{K}_0$ (см. § 3.2, пример 8).

Перейдем к построению всех максимальных подалгебр алгебры $\mathfrak{K}$. К ним относятся, в частности, множества $\mathfrak{K}_e = \mathfrak{K}_0 \cup \{e\}$ и $\mathfrak{K}_\varnothing = \mathfrak{K}_0 \cup \varnothing$, где $\{e\}$ — язык, состоящий из одного пустого слова e; $\varnothing$ — пустой язык. Рассмотрим отображение ψ, которое произвольному слову x над алфавитом $\mathfrak{A} = \{a_1, \ldots, a_m\}$ ставит в соответствие m-мерный набор $\psi(x) = \langle n_1, \ldots, n_m \rangle$, где n_i — число вхождений символа $a_i \in \mathfrak{A}$ в слово x, причем $\psi(e) = \langle 0, \ldots, 0 \rangle$. Распространим отображение ψ на произвольные подмножества свободной полугруппы $F(\mathfrak{A})$ так, что для любого $L \subseteq F(\mathfrak{A})$ $\psi(L) = \{\psi(x) \mid x \in L\}$. Пусть $\mathfrak{K}_{a_i}$ — множество кс-языков, содержащее все языки L такие, что либо $\langle n_1, \ldots, n_{i-1}, 1, n_{i+1}, \ldots n_m \rangle \overline{\in} \overline{\in} \psi(L)$ при любых $n_j \geqslant 0$ $(j \neq i)$, либо для любого $\langle n_1, \ldots, n_{i-1}, 1, n_{i+1}, \ldots, n_m \rangle \in \psi(L)$ найдется набор $\langle n_1', \ldots, n_m' \rangle \in \psi(L)$, где $n_i' > 1$, $n_j' = 0$ при любом $n_j = 0$ $(j \neq i)$. Иными словами, $L \in \mathfrak{K}_{a_i}$ тогда и только тогда, когда L вообще не содержит слов с единственным вхождением символа a_i, либо, если такое слово есть $(x \in L)$, для него всегда найдется по крайней мере одно слово $x' \in L$, содержащее несколько вхождений символа a_i и не содержащее символов, не входящих в слово x. Нетрудно проверить замкнутость множества $\mathfrak{K}_{a_i}$ относительно операций, входящих в сигнатуру алгебры $\mathfrak{K}$ так, что $\mathfrak{K}_{a_i}$ образует подалгебру алгебры $\mathfrak{K}$ для любого $a_i \in \mathfrak{A}$.

Теорема 5.17. *Система языков $\Sigma \subset \mathfrak{K}$ порождает алгебру $\mathfrak{K}$ тогда и только тогда, когда для каждой подалгебры $\mathfrak{K}_e$, $\mathfrak{K}_\varnothing$, $\mathfrak{K}_{a_i}$ $(i = 1, 2, \ldots, m)$ в системе Σ найдется хотя бы один язык, не принадлежащий данной подалгебре.*

Необходимость очевидна. *Достаточность.* Из определений множеств $\mathfrak{K}_e$ и $\mathfrak{K}_\varnothing$ следует, что система Σ содержит языки $\{e\}$ и $\varnothing$. Пусть $\{L_1, \ldots, L_m\} \subset \Sigma$, где $L_i \overline{\in} \mathfrak{K}_{a_i}$ $(i = 1, 2, \ldots, m)$. Тогда в язык L_i входит слово x, имеющее единственное вхождение символа a_i, причем любое слово с несколькими вхождениями символа a_i содержит по крайней мере один символ, не входящий в слово x. Пусть в слово x кроме символа a_i входят также символы $a_{i_1}, a_{i_2}, \ldots, a_{i_k} \in \mathfrak{A}$. Каждому символу a_{i_j} сопоставим язык $\{e\}$ $(j = 1, 2, \ldots, k)$, всем остальным символам алфавита $\mathfrak{A}$, кроме символа a_i, сопоставим язык $\varnothing$. Тогда

$$\overline{S}^m \begin{pmatrix} a_{i_j}, & r \\ \{e\}, & \varnothing \end{pmatrix} L_i = \{a_i\}, \quad j = 1, 2, \ldots, k,$$

а r пробегает множество $\{\mathfrak{A} \setminus \{a_i, a_{i_1}, \ldots, a_{i_k}\}\}$. Таким образом,

систему Σ удалось свести к системе $\Sigma^1 = \{\varnothing, \{e\}, \{a_i\} \mid i = 1, 2, \ldots, m\}$, которая образует базис алгебры $\mathfrak{K}$.

Операции сигнатуры алгебры $\mathfrak{K}$ имеют общее свойство: в результате применения к непустым словам они порождают новые слова, не убывающие по длине. Процесс порождения становится более гибким и разнообразным, если к операциям алгебры $\mathfrak{K}$ присоединить операцию деления на одноэлементный язык.

Рассмотрим алгебру $\tilde{\mathfrak{K}}$ всех кс-языков, сигнатура операций которой содержит все операции, входящие в сигнатуру алгебры $\mathfrak{K}$, а также операцию $L \backslash\!\!\backslash u$ левого деления, и множество ее максимальных подалгебр. Пусть $\mathfrak{T}^p_{a_i}$ (где $a_i \in \mathfrak{A}$; $p \in P$) — множество кс-языков такое, что $L \in \mathfrak{T}^p_{a_i}$ тогда и только тогда, когда $\psi(x) = \langle n_1, n_2, \ldots, n_i, \ldots, n_m \rangle$, где $n_i = 0$ или $n_i \vdots p$ для любого слова $x \in L$ (здесь и далее $n \vdots m$ означает, что число n кратно числу m). Иными словами, либо каждое слово $x \in L$ вообще не содержит вхождений символа a_i, либо число таких вхождений кратно данному простому числу $p \in P$. Нетрудно убедиться в том, что множество $\mathfrak{T}^p_{a_i}$ замкнуто относительно сигнатуры операций алгебры $\tilde{\mathfrak{K}}$. Действительно, последние не нарушают свойства слов, принадлежащих языкам из $\mathfrak{T}^p_{a_i}$, которое состоит в том, что число вхождений символа a_i кратно p. Следовательно, $\mathfrak{T}^p_{a_i}$ образует подалгебру алгебры $\tilde{\mathfrak{K}}$. Покажем, чо $\mathfrak{T}^p_{a_i}$ является максимальной подалгеброй алгебры $\tilde{\mathfrak{K}}$.

Пусть L — произвольный язык, не принадлежащий подалгебре $\mathfrak{T}^p_{a_i}$. Тогда найдется по крайней мере одно слово $x \in L$, такое, что $\psi(x) = \langle n_1, \ldots, n_m \rangle$, где n_i не делится на p. По определению $\{e\} \in \mathfrak{T}^p_{a_i}$, и можно выполнить подстановку $S^m\left(\begin{matrix} a_j \\ \{e\} \end{matrix} L\right) = L' \subseteq a_i^*$ при любом $j \neq i$. Пусть $x_1 \in L'$ — слово наименьшей длины такое, что $|x_1|$ не делится на p. Выберем одноэлементный язык $\{x_1' a_j x_1''\} \in \mathfrak{T}^p_{a_i}$ (при $j \neq i$), где $x_1'; x_1'' \in a_i^*$; $|x_1'| + |x_1''| = p$; $(|x_1| + |x_1'|) \vdots p$. Отсюда следует, что $|x_1''|$ не делится на p. Используя операцию умножения языков, получаем $L'' = L'\{x_1' a_j x_1''\}$. Далее, разделив L'' на одноэлементный язык $\{x_2 a_j\} \in \mathfrak{T}^p_{a_i}$, где $x_2 \in a_i^*$ и $|x_2| = |x_1| + |x_1'|$, получаем $L'' \backslash\!\!\backslash \{x_2 a_j\} = \{x_1''\}$, а применив $(p-2)$ раз операцию подстановки, —

$$S^2\left(\begin{matrix} a_i \\ \{x_1''\} \end{matrix} S^2\left(\begin{matrix} a_i \\ \{x_1''\} \end{matrix} \ldots S^2\left(\begin{matrix} a_i \\ \{x_1''\} \end{matrix} \{x_1''\}\right) \ldots\right)\right) = \{u\} \in a_i^*,$$

где $|u| = |x_1''|^{p-1}$. Однако для простого p и для a, которое не делится на p, $a^{p-1} \equiv 1 \pmod p$ (теорема Ферма [17]). Следовательно,

$|x_1''|^{p-1} \equiv 1 \pmod p$, и, разделив $\{u\}$ на одноэлементный язык $\{v\} \in \mathfrak{T}^p_{a_i}$, где $|v| = |x_1''|^{p-1} - 1$, получим однобуквенный язык $\{a_i\}$. Заметим, что $\{a_j\} \in \mathfrak{T}^p_{a_i}$ при любом $j \neq i$. Таким образом, присоединение произвольного языка $L \overline{\in} \mathfrak{T}^p_{a_i}$ к множеству $\mathfrak{T}^p_{a_i}$ порождает систему образующих алгебры $\tilde{\mathfrak{K}}$. В силу бесконечности множества P всех простых чисел справедлива следующая теорема [118].

Теорема 5.18. *Множество всех максимальных подалгебр алгебры $\tilde{\mathfrak{K}}$ бесконечно.*

Эта теорема справедлива, несмотря на конечную порожденность алгебры $\tilde{\mathfrak{K}}$.

Полученные результаты переносятся на алгебры регулярных языков, в сигнатурах которых операции рекурсии заменены итерацией.

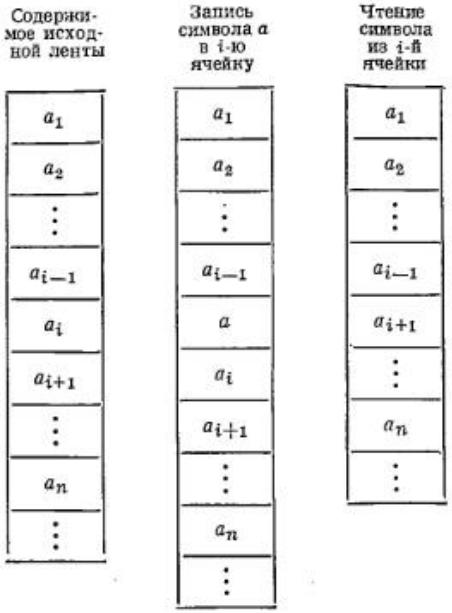
ГЛАВА 6 ПАРАМЕТРИЧЕСКИЕ СИСТЕМЫ ПРОГРАММИРОВАНИЯ

§ 6.1. Автоматы над внутренней памятью

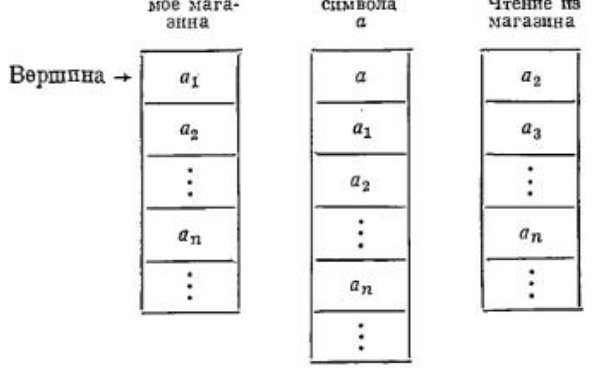
В связи с синтезом логических структур многих важных системных процессов представляет интерес применение в абстрактной модели ЭВМ, рассмотренной в § 4.1, в качестве управляющих, наряду с конечными, бесконечных автоматов с различной организацией внутренней памяти.

Обобщая известные структуры памяти, рассмотрим **эластичную ленту** [21], или, кратко, **ленту**, разделенную на ячейки, которые последовательно занумерованы натуральными числами из множества $N = \{1, 2, \ldots\}$. Номер ячейки играет роль адреса при обращении к памяти. Ячейки ленты расположены вертикально так, что ячейка с номером 1 является самой верхней. Любая ячейка может содержать один символ (букву) из множества $\mathfrak{A}$ (необязательно конечного), названного алфавитом данной ленты. В каждый момент времени содержимым ленты является цепочка произвольной длины $x = a_1 a_2 \ldots a_n \in F(\mathfrak{A})$. При этом цепочка x располагается в n верхних ячейках, остальные ячейки ленты остаются пустыми. К ленте допускаются обращения в режиме записи и чтения. В режиме записи символа $a \in \mathfrak{A}$ в i-ю ячейку «хвост» $a_i a_{i+1} \ldots a_n$ цепочки x смещается вдоль ленты на одну ячейку вниз и символ a помещается в освободившуюся i-ю ячейку так, что содержимое ленты составляет цепочку $x' = a_1 \ldots a_{i-1} a a_i a_{i+1} \ldots a_n$; наоборот, в режиме чтения из i-й ячейки воспринимается, а затем стирается, символ a_i (ее содержимое), причем подцепочка $a_{i+1} \ldots a_n$ смещается на одну ячейку вверх, образуя новое содержимое $x'' = a_1 \ldots a_{i-1} a_{i+1} \ldots a_n$ данной ленты (см. стр. 223).

Рассмотренная лента является естественным обобщением структур памяти, положенных в основу известных концепций бесконечных автоматов. Понятие ленты служит также абстрактной моделью памяти стековых ЭВМ, отражая ее наиболее характерные свойства.



При реализации некоторых важных системных процессов широкое распространение получила **магазинная память**. **Магазин** представляет собой ленту, в которой для записи и чтения доступна лишь верхняя ячейка, называемая **вершиной магазина**; в режиме записи содержимое магазина сдвигается на одну ячейку вниз и в освободившуюся вершину записывается некоторый символ из алфавита данного магазина; при чтении содержимое вершины магазина стирается, а оставшаяся часть цепочки смещается на одну ячейку вверх; таким образом, символ, записанный в магазин последним, воспринимается при чтении как первый:



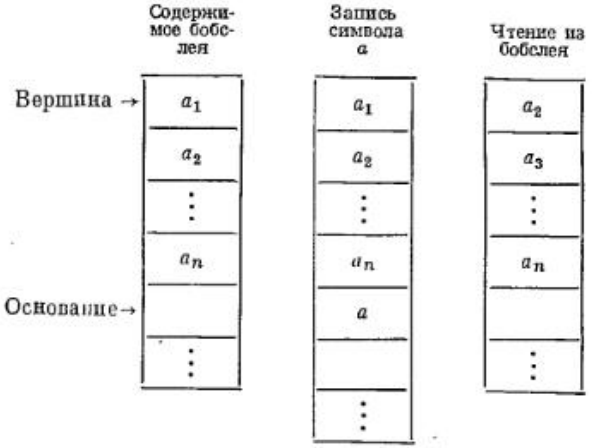
Наряду с двусторонними магазинами, работающими как в режиме записи, так и в режиме чтения, можно рассматривать односторонние магазины, работающие лишь в одном из указанных режимов. Магазин, работающий лишь в режиме чтения, используется в качестве входной ленты для автоматных структур. Функция выходов автомата состоит в накоплении результирующей цепочки на выходной ленте, которая представляет собой односторонний магазин, функционирующий в режиме записи.

Модификацией магазина является **стековая память.** Стек представляет собой магазин, любая ячейка которого доступна просмотру (чтению с сохранением содержимого ячейки).

Другой важной разновидностью магазина является **магазин-счетчик,** алфавит которого содержит лишь один символ. Накопление этого символа на ленте используется в качестве количественной характеристики в процессе переработки информации.

Стеки и счетчики широко используются в системном программировании при реализации рекурсивных процедур, циклов и других важных алгоритмических конструкций.

Важным ограничением режимов обращения к ленте является введение естественной очередности на порядок считывания символов, записываемых на ленту. Для организации функционирования ленты по принципу очереди режимы чтения и записи необходимо разделить по месту обращения. Чтение, как и в случае магазина, производится из самой верхней ячейки ленты — ее вершины, а запись — снизу в свободную ячейку с наименьшим номером — основание ленты. Лента, образованная таким образом, названа **циклической очередью или памятью типа «бобслей»** (кратко — бс-памятью) [123].



Как и в случае магазина, бобслей, алфавит которого содержит лишь один символ, является счетчиком. Для практических приложений представляет интерес изучение вспомогательной памяти, сочетающей свойства магазина и бобслея. К такой памяти относится прежде всего память типа «бс-магазин» [89, 90]. **бс-Магазином** является лента, допускающая обращение к своей вершине в режиме записи и чтения, а к основанию — только в режиме записи. Таким образом, бс-магазин моделирует как магазин, так и бобслей. Память подобного типа может быть использована при параллельной обработке скобочных структур (см. § 6.4).

Определение автомата с n вспомогательными лентами является естественным обобщением известных концепций бесконечных автоматных структур. Рассмотрим конечный автомат, который кроме входного и выходного каналов имеет каналы для работы с n внутренними лентами ($n \geqslant 0$). На входной и выходной лентах размещаются соответственно входные и накапливаются выходные цепочки.

Автомат A с n внутренними лентами представляет собой объект

$$A = (S, X, Y, Z_1, \ldots, Z_n, \varphi, \varphi_i, \psi_i, \delta, s_0, F),$$

где S — конечное множество состояний автомата A; X и Y — соответственно входной и выходной алфавиты; Z_i — алфавит i-й внутренней ленты; $\varphi : S \times \{X \cup \wedge\} \to S$ — функция переходов, связанная с чтением входной цепочки (символ $\wedge$ используется для переключений автомата в новое состояние без обращения к входной ленте); $\varphi_i : S \times Z_i \times N \to S$ — функция переходов, связанная с чтением из r-й ячейки ($r \in N$) i-й внутренней ленты; $\psi_i : S \to S \times Z_i \times N$ — функция записи в r-ю ячейку i-й внутренней ленты ($i = 1, 2, \ldots, n$); $\delta : S \to Y \times S$ — функция выходов; $s_0 \in S$ — начальное состояние; $F \subset S$ — множество заключительных состояний автомата. Конфигурацией автомата A назовем набор $(s, p, p_1, p_2, \ldots, p_n, q)$, где $s \in S$ — состояние автомата; $p(q)$ — содержимое входной (выходной) ленты; p_i — содержимое i-й внутренней ленты автомата ($i = 1, 2, \ldots, n$).

В каждый момент дискретного автоматного времени детерминированный автомат A с n внутренними лентами совершает одно и только одно элементарное действие из следующих пяти типов действий.

1. Чтение очередного символа из входной ленты и переход в новое состояние, зависящее в соответствии с функцией φ от прочитанного символа и состояния автомата перед этой операцией.

2. Чтение из некоторой ячейки i-й внутренней ленты согласно выбранному типу памяти и переход в следующее состояние, зависящее в соответствии с функцией φ_i от прочитанного символа и текущего состояния автомата.

3. Запись в некоторую ячейку i-й внутренней ленты некоторого символа из алфавита Z_i, зависящего в соответствии с функцией ψ_i

от текущего состояния автомата, и однозначный переход в следующее состояние.

4. Запись на выходную ленту некоторого символа (из алфавита Y), зависящего в соответствии с функцией δ от текущего состояния автомата, и однозначный переход в следующее состояние.

5. Переключение автомата A в очередное состояние в соответствии с функцией φ без обращения к его лентам.

В первом и втором случаях при исчерпании содержимого соответствующих лент (когда считывается пустой символ) переход в новое состояние считается неопределенным и автомат безрезультатно останавливается.

В недетерминированном автомате с n лентами функции, определяющие переходы, могут быть многозначными. Кроме того, в некоторых состояниях автомат может выполнять одно из некоторой совокупности допустимых элементарных действий.

Детерминированный и недетерминированный автоматы могут быть заданы двувходовой таблицей, столбцы которой соответствуют состояниям данного автомата, а строки сгруппированы в $(n+2)$ секции. Первая секция — секция чтения из входной ленты. Составляющие ее строки соответствуют различным символам входного алфавита и заполняются в соответствии с функцией φ (действие 1). В следующих n секциях (по числу внутренних лент) согласно функциям φ_i (действие 2) фиксируются переходы автомата, связанные с чтением из соответствующих лент, причем строки секции, сопоставленной i-й ленте, отмечены разными символами алфавита Z_i данной ленты ($i = 1, 2, \ldots, n$). Последняя секция состоит из единственной строки, в которой отражены действия 3—5 в соответствии с функциями ψ_i, δ, φ.

Автоматы с n вспомогательными лентами допускают параллельный режим работы благодаря выполнению групповых операций обращения к нескольким лентам. При групповой операции в некотором состоянии последняя секция содержит выражение $[x, z_{r_1}^{q_1}, z_{r_2}^{q_2}, \ldots, z_{r_n}^{q_n}, y]$, где $q_i = \left\{\begin{matrix}-1\\0\\1\end{matrix}\right\}$ $(i = 1, 2, \ldots, n)$; x — символ, считываемый с входной ленты; $z_{r_i}^{-1}$ $(z_{r_i}^{1})$ — символ, считываемый (записываемый) с r_i-й ячейки i-й внутренней ленты, $z_{r_i}^{0} = \wedge$; y — выходной символ; каждый из перечисленных символов может совпадать с $\wedge$ ($q = 0$, $z^0 = \wedge$), если соответствующая лента не участвует в данной групповой операции; отсутствие нижнего индекса означает обращение к вершине магазина или, в зависимости от режима обращения, к вершине или основанию бобслея.

В качестве примера рассмотрим две таблицы, задающие автоматы с одной внутренней лентой: детерминированный (табл. 9) и недетерминированный (табл. 10) автоматы с входным алфавитом $\{a, b\}$ и алфавитом внутренней ленты $\{p, q\}$. Второй автомат недетерминирован в трех состояниях. В состоянии 1 это происходит из-за неоднозначности перехода в состояния 2 и 3 при считывании символа a из входной ленты и двух типов элементарных действий — считывания из входной ленты и записи во внутреннюю ленту. В состоянии 2 автомату представляется возможность выбора одного из двух элементарных действий — считывания из входной или внутренней ленты. В состоянии 4 существует неоднозначность перехода (не сопровождающегося никаким другим действием) в состояния 1 и 3. В состоянии 5 выполняется групповая операция чтения на выходе символа a и записи в вершину внутренней ленты символа q с последующим переходом в состояние 4[1].

Таблица 9

	1	2	3	4	5
a b	2 3		5 4		
p q				2, p^{-1} 1, q^{-1}	
		1, p			4, q

Таблица 10

	1	2	3	4	5
a b	2, 3 4	3			
p q		4, p^{-1} 2, q^{-1}			
	1, p		4, q	1, 3	4, $[a, q]$

Многотактный переход из начального состояния s_0 с пустым наполнением внутренних лент в некоторое заключительное состояние $s \in F$ считается правильным, если наполнение внутренних лент в состоянии s также пусто. Автомат A воспринимает входную цепочку $x \in F(X)$, если возможен правильный переход из начального состояния s_0 в некоторое заключительное состояние $s \in F$, при котором происходит полное считывание цепочки x из входной ленты. Множество всех цепочек, воспринимаемых автоматом A, образует язык $L(A) \subseteq F(X)$, представимый данным автоматом. При изучении вопросов, связанных с представимостью языков, достаточно

[1] В секциях считывания из внутренней ленты (см. табл. 9, 10) для наглядности указывается операция вычеркивания считываемой буквы (p^{-1} и q^{-1})

рассматривать автоматы с n внутренними лентами без выхода. Конкретизация памяти, используемой в автоматах с n внутренними лентами, приводит к известным концепциям бесконечных автоматов.

Автомат порождает выходную цепочку $y \in F(Y)$, если при некотором заполнении входной ленты возможен правильный переход из начального в заключительное состояние, при котором на выходной ленте оказывается записанной цепочка y. Язык M порождается автоматом A, если последним порождаются все цепочки языка и только они. Автомат отображает (или транслирует) язык L на язык M, если при начальных заполнениях входной ленты цепочками языка L автомат порождает все цепочки языка M и только такие цепочки, одновременно воспринимая соответствующую цепочку языка L.

Проиллюстрируем сущность приведенных выше определений на некоторых характерных примерах, отражающих особенности соответствующих автоматных структур.

Автоматы, внутренними лентами которых являются магазины, составляют класс **n-магазинных автоматов** [28]. В системном программировании для реализации многих важных процессов широко применяются одномагазинные автоматы (см. § 6.2).

Пример 1. Построим одномагазинный автомат A_1, представляющий зеркальный кс-язык $L_1 = \{x \divideontimes x^r \mid x \in F(\mathfrak{A})\}$, где $\mathfrak{A} = \{a, b\}$, x^r — зеркальное отображение цепочки x. Автомат имеет входной алфавит $X = \{a, b, \divideontimes\}$ и алфавит внутреннего магазина $Z = \{a', b'\}$. Функционирование автомата определяется табл. 11. Начальное состояние —1, заключительные — 6, 7. Автомат считывает с входной ленты начальную подцепочку $x \in F(\mathfrak{A})$ вплоть до появления на входе разделителя $\divideontimes$. Одновременно во внутреннем магазине накапливается копия зеркального образа прочитанной

Таблица 11

	1	2	3	4	5	6	7
a	2			2	6		
b	3			3	7		
$\divideontimes$				5			
a'						5	
b'							5
		4, a'	4, b'				

подцепочки. Затем происходит опустошение магазина при считывании с входа подцепочки x^r. Так, воспринимая цепочку $abaab \divideontimes baaba \in L_1$, автомат функционирует следующим образом. Вначале считывает на входе подцепочку $abaab$, проходя последовательность состояний 1, 2, 4, 3, 4, 2, 4, 2, 4, 3; при этом в состояниях 2, 3 в магазине накапливается обращенная копия $b'a'a'b'a'$ считанной подцепочки. Затем переходит в состояние 4, в котором с входной ленты считывается разделитель $\divideontimes$, и воспринимает «хвост» $baaba$, вырабатывая последовательность состояний 5, 7, 5, 6, 5, 6, 5, 7, 5, 6 так, что в состояниях 6, 7 происходит опустошение магазина.

Рассмотренный пример отражает соответствие структуры цепочек зеркального языка основному принципу работы магазина: «последняя запись считывается первой» (представление языков магазинными автоматами рассмотрено в § 6.2).

бс(n)-Автоматом называется автомат с n внутренними бобслеями. Изучению класса автоматных структур данного типа посвящена работа [123]. В частности, представляет интерес рассмотрение класса бс-автоматов с одним бобслеем без выходов.

Пример 2. Построим бс-автомат A_2, представляющий нс-язык $L_2 = \{x \divideontimes x \mid x \in F(\mathfrak{A})\}$, где $\mathfrak{A} = \{a, b\}$. Автомат A_2 имеет входной алфавит $X = \{a, b, \divideontimes\}$ и алфавит бобслея $Z = \{a', b'\}$. Функционирование бс-автомата A_2 определяется также табл. 11 с учетом замены магазина бобслеем. Начальное состояние — 1, заключительные — 6, 7. Автомат считывает с входной ленты начальную подцепочку $x \in F(\mathfrak{A})$, копируя ее в бобслее цепочкой $x' \in F(Z)$, и после появления на входе разделителя $\divideontimes$ производит опустошение бобслея при считывании с входа подцепочки x. Так, воспринимая цепочку $abba \divideontimes abba \in L$, автомат A_2 функционирует следующим образом. В состояниях 1, 4 считывает с входной ленты начальную подцепочку $abba$, наполняя (в состояниях 2, 3) бобслей ее копией $a'b'b'a'$; при этом вырабатывается последовательность состояний 1, 2, 4, 3, 4, 3, 4, 2. Далее, в состоянии 4 считывает на входе разделитель $\divideontimes$, переходя в состояние 5 с последующим чтением «хвоста» $abba$ входной цепочки и опустошением бобслея. Воспринимая цепочку $abba \divideontimes abba \in L_2$, автомат осуществляет правильный переход (1, 2, 4, 3, 4, 3, 4, 2, 4, 5, 6, 5, 7, 5, 7, 5, 6) из начального состояния 1 в заключительное 6.

Справедливо следующее утверждение [123].

Теорема 6.1. *Для любого бс-автомата A существует эквивалентная ему машина Тьюринга M, так что $L(A) = L(M)$, где $L(A)$ и $L(M)$ — языки, представимые соответственно автоматом A и машиной M.*

Следствие. В автоматах с одной вспомогательной лентой представимы произвольные рекурсивно-перечислимые множества.

Счетчиковым автоматом или счетчиковой машиной [105] называется автомат A с n вспомогательными счетчиковыми лентами.

Класс $C(n)$ счетчиковых автоматов образует собственный подкласс в классах $M(n)$ n-магазинных и $B(n)$ бс(n)-автоматов, так что $C(n) \subset (M(n) \cap B(n))$.

Пример 3. Построим недетерминированный счетчиковый автомат A_3, осуществляющий трансляцию $T: a^n b^n c^n \to I^n$ $(n = 1, 2, \ldots)$. Языком, представимым в A_3, является нс-язык $L_3 = \{a^n b^n c^n \mid n = 1, 2, \ldots\}$, а язык $L_3' = \{I^n \mid n = 1, 2, \ldots\}$ порождается данным автоматом. Трансляция состоит в чтении на входе цепочки $a^n b^n c^n$, воспринимая которую автомат осуществляет правильный переход из начального состояния в заключительное, порождая на выходной ленте цепочку I^n, фиксирующую длину согласованных подцепочек на входной ленте.

Автомат A_3 имеет входной алфавит $X = \{a, b, c\}$, два счетчика с алфавитом $Z = \{p\}$ и выходной алфавит $Y = \{I\}$. Секция записи, содержащая групповые операции, определяющие функционирование автомата, представлена в табл. 12. Начальное состояние — 1, заключительное — 3. В состоянии 1 выполняется групповая операция чтения символа a на входе с одновременной записью в оба счетчика символа p. Затем автомат может оставаться в состоянии 1, повторяя указанные действия, либо перейти в состояние 2, в котором

Таблица 12

1	2	3
1, 2 [a, p, p, $\wedge$]	2, 3 [b, p^{-1}, $\wedge$, I]	3 [c, $\wedge$, p^{-1}, $\wedge$]

осуществляется одновременно чтение символов b с входной ленты и опустошение первого счетчика и запись на выходную ленту символов I. Переходя в состояние 3, автомат читает на входе символы c и опустошает второй счетчик. Автомат недетерминирован в состояниях 1, 2.

Автомат A_3 воспринимает лишь цепочки языка L_3. Действительно, пусть на вход автомата A_3 поступает цепочка $a^m b^n c^k$, причем $m \neq n$. Если $m < n$, то автомат считывает в состоянии 1 на входе подцепочку a^m, одновременно наполняя оба счетчика содержимым p^m. В состоянии 2 автомат воспринимает подцепочку b^m, опустошая первый счетчик. При этом на входной ленте остается «хвост» $b^{n-m}c^k$. Однако чтение в состоянии 2 очередного входного символа b приводит к считыванию из пустого счетчика, что влечет за собой безрезультатный останов автомата. Если $m > n$, то автомат считывает в состояниях 1, 2 входную подцепочку $a^m b^n$ и переходит в состояние 3, читая на входе символы c. При этом содержимым первого счетчика является цепочка p^{m-n} и его дальнейшее опустошение невозможно, что исключает правильный переход из начального в заключительное состояние. Следовательно, $m = n$. Равенство $m = n = k$ устанавливается аналогично.

Таким образом, язык L_3 представим автоматом A_3, осуществляющим трансляцию цепочек $x \in L_3$ в цепочки $x' \in L_3'$.

Рассмотренные концепции автоматов над внутренней памятью и соответствующие формы их представлений применяются при разработке систем автоматизации программирования как транслирующего, так и интерпретирующего типа. Основным назначением трансляторов является существенное ускорение процесса получения программ на машинных языках и обеспечение более полного использования машинных ресурсов. Помимо перевода алгоритмов с одного языка (входного) на другой (промежуточный или язык ЭВМ) транслирующая система позволяет выявлять синтаксические и семантические ошибки в алгоритмах, чем частично решается задача отладки программ. Кроме того, благодаря соответствующим оптимизационным блокам системы, обрабатывающим алгоритмы посредством применения эквивалентных преобразований на уровне входного или выходного языка, можно получить программы, удовлетворяющие определенным требованиям качества.

Наряду с транслирующими системами все большее значение приобретают системы интерпретирующие, которые (как и транслирующие) могут быть реализованы как программными, так и схемными средствами. Интерпретатор осуществляет последовательную обработку (перевод) операторов программы, записанной в его входном языке, и одновременно их интерпретацию — выполнение машинных команд, реализующих данный оператор. Процессы перевода выполняемого алгоритма и его интерпретация при использовании этих систем тесно взаимосвязаны, что облегчает отработку и отладку программ, в частности при постановке на ЭЦВМ задач исследовательского характера. Однако повторное выполнение операторов в таких системах связано с повторным переводом их на язык интерпретации (как правило, язык машины), поэтому скорость выполнения готовых программ в системах интерпретации ниже, чем в системах трансляции.

Проблема трансляции и интерпретации данного языка программирования представляет собой по существу две проблемы — анализа и синтеза. В основу построения первых трансляторов положена идея компоновки рабочей программы из программ, соответствующих отдельным операторам исходного алгоритма. Трансляторы такого рода многопроходовые, т. е. при их работе запись обрабатываемого алгоритма или его эквивалента просматривается несколько раз. Так, при первом просмотре обрабатываются все описательные части программы, в которых приводятся характеристики данных, при втором — переводятся в промежуточные коды арифметические операторы и т. д. Затем осуществляется общее распределение памяти и присвоение истинных адресов. Если входной язык системы

программирования оказывается пригодным для описания алгоритмов трансляции, то созданием таких систем в значительной мере решается проблема автоматизации процесса конструирования трансляторов [128].

Появление языков для описания грамматик языков программирования (так называемых метаязыков) создало предпосылки для построения алгоритмов синтаксического контроля и анализа, которые по заданному описанию грамматики одного из языков данного класса и программе в этом языке могут выдавать синтаксическое дерево программы. Существенно, что процесс синтаксического контроля или анализа языка во многих случаях не зависит от конкретных особенностей ЭВМ. Благодаря этому блок анализа транслятора становится универсальным, применимым для трансляции с других языков данного класса. Однако метод построения многих систем программирования жестко привязан к конкретным входному и выходному языкам. Вместе с тем опыт эксплуатации ЭВМ, снабженных соответствующими системами программирования, подсказывает, что пара «входной и выходной языки системы» не может оставаться постоянной. Быстрое расширение сферы применения ЭВМ требует большой свободы при создании языков программирования, оперативной разработки новых и совершенствования существующих систем программирования. Кроме того, конструирование систем программирования связано с исследованием класса языков, которые получаются в результате несущественных изменений входного языка с целью достижения разумного компромисса между изобразительной мощью языка и сложностью строящейся системы. В то же время даже незначительная модификация входного или выходного языка в системе программирования, жестко привязанной к последним, может привести к глобальной перестройке всей системы в целом. Это обусловило интенсивное развитие нового подхода к разработке систем программирования, ориентированных на класс входных и выходных языков, — параметрических систем программирования [95—98].

Развитие параметрических систем программирования тесно связано с разработкой методов их конструирования, оставляющих неизменной значительную часть создаваемой системы, независимо от выбора входного и выходного языков из некоторых классов. Такой подход к организации структуры систем программирования определяет ее качество, трудоемкость и степень автоматизации разработки.

Структура синтаксически управляемых трансляторов представляет собой совокупность управляющей программы и семантических подпрограмм, осуществляющих замену синтаксического разложения программы ее семантическим эквивалентом в выходном языке транслирующей системы, в частности в машинном. Транслятор обрабатывает исходную информацию, посимвольно просматривая ее в порядке записи. При этом программа обработки каждого символа определяется, вообще говоря, не только этим символом, но и содержащей его синтаксической структурой. Некоторые символы на данном уровне анализа не могут быть обработаны и засылаются в специально организованную вспомогательную память: магазинную, бобслей, счетчиковую и другие, в зависимости от соответствующей синтаксической структуры.

Таким образом, перевод с одного языка на другой делится на два этапа, которые часто выполняются одновременно: 1) синтаксическое разложение исходной записи программы, т. е. перевод ее на язык с тривиальным синтаксическим анализом (проблема анализа); 2) замена элементов разложения их семантическими эквивалентами на выходном языке, в частности машинном. Как правило, при этом перевод распадается на цепочку указанных этапов, связанную с использованием одного или нескольких промежуточных языков (посредников) [128], откуда возникают некоторые общие проблемы, безотносительно к входным — выходным языкам, так как трудности перевода обусловливаются не столько избранными языками, сколько грамматиками, задающими эти языки.

Проблема конструирования синтаксически управляемых трансляторов, ориентированных на широкий класс грамматик, содержащих аппарат для своего расширения, стимулировала новые исследования по формализации синтаксиса и семантики языков программирования. Процессы трансляции и интерпретации достаточно широкого класса языков программирования происходят в одних и тех же условиях и могут быть описаны алгоритмами выполнения заданной переработки информации в зависимости от значений некоторых параметров, определяющих конкретный язык. Если запись в исходном языке транслируется в некоторый универсальный машинно-независимый язык, достаточно близкий к внутренним языкам определенного класса ЭВМ, формальное описание синтаксиса и семантики языка приобретает машинно-независимую форму, что позволяет автоматизировать процесс построения трансляторов по их описанию для пары из множества входных и выходных языков. Основные трудности задачи в данном случае заключаются в создании удобных метаязыков, на базе которых могут быть разработаны параметрические системы (транслирующего или интерпретирующего типа), ориентированные на класс языков.

В работах [15, 129, 96—98] предложен метод построения параметрических систем, состоящий в эффективном решении проблемы синтаксического анализа для широкого класса языков программирования. Этот метод положен в основу параметрической системы транслирующего типа [88], ориентированной на класс языков. При разработке использован метод задания семантики по синтаксической структуре предложения языка посредством индукции. В связи с этим к проблеме формализации языка программирования добав-

ляется проблема формализации его синтаксического и семантического описания, которая решается путем создания метаязыка для описания синтаксиса и на его основе — метаязыка для описания семантики языков программирования. В качестве последнего предложен ряд языков, в частности язык СЁМОЛ [88], построенный на базе адресного языка [128] (см. § 6.4, 6.5).

Дальнейшее развитие идей автоматизации программирования привело к созданию операционных систем, в которых автоматизируется не только процесс программирования (включая хранение, документирование и размножение программ), но и труд оператора [87].

В связи с расширением использования ЭВМ возникли задачи, компактное описание которых выходит за рамки одного процедурно-ориентированного языка [36]. Для решения таких задач были созданы языки общего назначения: ПЛ/1 [102], Симула-67 [42] и АЛГОЛ-68 [11], которые можно отнести к языкам третьего поколения. При описании синтаксиса и семантики языков, подобных АЛГОЛу-68, может быть использован, в частности, аппарат многоосновных алгебр (см. § 3.5, 6.7, а также [71]).

§ 6.2. Синтез магазинных автоматов

Одной из важных проблем в разработке современных систем программирования является проблема синтаксического анализа программ. Процесс синтаксического анализа программы состоит в распознавании правильности данной программы, т. е. ее принадлежности к рассматриваемому алгоритмическому языку[2]. Этот этап называется этапом **синтаксического контроля** программы. Одновременно с контролем осуществляется описание синтаксической структуры правильных программ, подобно тому как производится грамматический разбор предложений в естественных языках.

Для формального описания синтаксиса алгоритмических языков широко применяется так называемая бэкусовская нормальная форма. Пусть, например, требуется описать синтаксическую структуру формул, строящихся с помощью сложения и умножения переменных, идентификаторы которых представляют собой произвольные слова в двухбуквенном алфавите (x, y). Применяя бэкусовскую нотацию, определение произвольной формулы указанного вида можно задать двумя формулами:

$$\langle\text{формула}\rangle ::= (\langle\text{формула}\rangle) \times (\langle\text{формула}\rangle) \mid \langle\text{переменная}\rangle \mid (\langle\text{формула}\rangle) + (\langle\text{формула}\rangle)$$
$$\langle\text{переменная}\rangle ::= x \mid y \mid \langle\text{переменная}\rangle x \mid \langle\text{переменная}\rangle y.$$

[2] Программа здесь рассматривается как цепочка символов, а алгоритмический язык — как множество таких цепочек.

Существует более сжатая форма записи этих соотношений в виде системы уравнений в языках (см. § 5.3). Для осуществления такой записи прежде всего введем буквенные обозначения всех понятий и символов из приведенных бэкусовских строчек: a, b — соответственно открывающая и закрывающая скобки, c, d — знаки сложения и умножения. Тогда, согласно приведенному определению, понятие ⟨формула⟩ означает некоторое множество слов (язык) в алфавите $\{a, b, c, d, x, y\}$. Обозначим это множество буквой S, а множество слов, представленное понятием ⟨переменная⟩, — R. Тогда вместо бэкусовских строк может быть записана эквивалентная им система уравнений[3]

$$\left.\begin{aligned} S &:= R \vee aSbcaSb \vee cSbdaSb, \\ R &:= x \vee y \vee Rx \vee Ry. \end{aligned}\right\} \qquad (6.1$$

Операция дизъюнкции здесь понимается как объединение соответствующих множеств, произведение множеств слов PQ обозначает множество, составленное из всех слов вида pq, где $p \in P$, $q \in Q$. Используя описанную в § 5.3 методику решения систем уравнений, обозначим через $S^{(0)}$ и $R^{(0)}$ множества слов, получаемые подстановкой в правые части уравнений (6.1) вместо S и R пустого множества $\varnothing$. Тогда $S^{(0)} = \varnothing$ и $R^{(0)} = x \vee y$. При любом $i = 0, 1, 2, \ldots$

$$S^{(i+1)} = R^{(i)} \vee aS^{(i)}bcaS^{(i)}b \vee aS^{(i)}bdaS^{(i)}b,$$
$$R^{(i+1)} = x \vee y \vee R^{(i)}x \vee R^{(i)}y.$$

Нетрудно видеть, что при всех i $S^{(i)} \subseteq S^{(i+1)}$ и $R^{(i)} \subseteq R^{(i+1)}$. Объединения $\bigcup_{i=0}^{\infty} S^{(i)} = S$ и $\bigcup_{i=0}^{\infty} R^{(i)} = R$ представляют собой минимальное решение системы уравнений (6.1). Легко убедиться, что при этом множество S состоит из всех формул, а множество R — из всех переменных в смысле приведенного выше определения этих понятий.

Указанным приемом любое описание синтаксиса в виде бэкусовских нормальных форм может быть представлено в виде системы уравнениях в языках. Преимуществом последней формы записи наряду с краткостью является также возможность выполнения различных тождественных преобразований. Для подобных преобразований используется коммутативность дизъюнкции (объединения) языков и дистрибутивность умножения по отношению к дизъюнкции. Это позволяет раскрыть скобки и выносить за скобки общие множители так же, как в обычной алгебре (учитывая лишь

[3] Здесь и до конца настоящего параграфа синтаксические понятия (не терминалы) обозначаются прописными латинскими буквами, а состояния в соответствующих автоматах — греческими.

что умножение языков в общем случае некоммутативно). Используя подобные преобразования, систему (6.1) можно привести к виду

$$\left.\begin{aligned} S &:: = R \vee aSb(c \vee d)aSb, \\ R &:: = x \vee y \vee R(x \vee y). \end{aligned}\right\} \qquad (6.2)$$

При построении трансляторов с входных языков на машинные в традиционных ЭВМ синтаксический анализ выражений на входных языках осуществляется программными средствами. Программы получаются весьма громоздкими, а их выполнение занимает много машинного времени. В машинах со структурной интерпретацией языков синтаксический анализ выражений может осуществляться схемными методами, с помощью которых алгоритмы анализа реализуются экономнее и гораздо быстрее [18]. Основным методом как программного, так и схемного решения является построение магазинных автоматов, представляющих выбранные языки. Для решения вопросов, связанных с представимостью языков автоматами над внутренней памятью, особую роль играют правильные переходы. Напомним, что переход автомата из состояния α в состояние β называется правильным, если в обоих этих состояниях (но, вообще говоря, не в промежуточных) внутренние ленты автомата пусты. Можно дать и другое определение правильности перехода магазинного автомата из одного состояния в другое. Если в начальном состоянии α во внутреннем магазине хранилось какое-то слово p, то переход из состояния α в состояние β называется правильным тогда и только тогда, когда в состоянии β содержимое внутреннего магазина равно p, а во всех промежуточных состояниях оно не меньше p. Иными словами, стирание букв первоначального заполнения внутреннего магазина во время всех промежуточных переходов запрещено. В силу предположения о неопределенности переходов при считывании с лент пустой буквы приведенные определения, очевидно, эквивалентны: из состояния α в состояние β правильный переход в смысле первого определения возможен тогда и только тогда, когда переход правильный в соответствии со вторым определением. В остальном на случай магазинных автоматов полностью распространяются приведенные в § 6.1 основные понятия, связанные с представимостью языков, их порождением, отображением (трансляцией) одного языка на другой и др.

Задача построения магазинного автомата, представляющего заданный язык, называется **задачей синтеза** соответствующего автомата. Оказывается, что для всех языков, которые задаются уравнениями вида (6.1), получаемыми из произвольных бэкусовских нормальных форм, задача синтеза всегда разрешима. Опишем один простой и естественный алгоритм синтеза [28, 18].

Рассмотрим задачу синтеза автомата, представляющего язык, задаваемый системой уравнений (6.2). Первый шаг алгоритма состоит

в том, что все вхождения неизвестных (S и R) в правые части уравнений индивидуализируются путем приписывания разным вхождениям одной и той же переменной разных индексов. В нашем случае это приводит к выражениям

$$\left.\begin{aligned} S &= R_1 \vee aS_1b(c \vee d)aS_2b, \\ R &= x \vee y \vee R_2(x \vee y). \end{aligned}\right\} \qquad (6.3)$$

На втором шаге правые части уравнений, состоящие из двух или более дизъюнктивных членов, заключаются в скобки, после чего все символы в них (включая знаки дизъюнкции и скобки, но не введенные индексы у переменных) разделяются вертикальными черточками так, чтобы каждый из них оказался между двумя последовательными черточками. Выражения (6.3) принимают при этом вид

$$\left.\begin{aligned} S &= |(|R_1| \vee |a|S_1|b|(|c| \vee |d|)|a|S_2|b|)|, \\ R &= |(|x| \vee |y| \vee |R_2|(|x| \vee |y|)|)|. \end{aligned}\right\} \qquad (6.4)$$

На третьем шаге все черточки нумеруются последовательно целыми числами. При этом черточки, которые стоят в начале всех дизъюнктивных членов, заключенных в скобки, получают тот же номер, что и черточка перед скобками, а те черточки, которые стоят в конце этих дизъюнктивных членов, нумеруются так же, как черточка после соответствующих скобок. Остальные черточки получают различные номера. После выполнения указанной нумерации

$$S = \underset{1}{|}\underset{1}{(}\underset{1}{|}R_1\underset{2}{|}\vee\underset{1}{|}a\underset{3}{|}S_1\underset{4}{|}b\underset{5}{|}\underset{5}{(}\underset{5}{|}c\underset{6}{|}\vee\underset{5}{|}d\underset{6}{|}\underset{6}{)}\underset{6}{|}a\underset{7}{|}S_2\underset{8}{|}b\underset{2}{|}\underset{2}{)}\underset{2}{|},$$

$$R = \underset{9}{|}\underset{9}{(}\underset{9}{|}x\underset{10}{|}\vee\underset{9}{|}y\underset{10}{|}\vee\underset{9}{|}R_2\underset{11}{|}\underset{11}{(}\underset{11}{|}x\underset{10}{|}\vee\underset{11}{|}y\underset{10}{|}\underset{10}{)}\underset{10}{|}\underset{10}{)}\underset{10}{|}.$$

Все введенные номера (назовем их местами в данных выражениях) отождествляются с состояниями синтезируемого автомата. Все переменные с индексами, входящие в правые части построенных выражений (S_1, S_2, R_1, R_2), составляют алфавит внутреннего магазина. Буквы a, b, c, d, x, y — входной алфавит (алфавит входного магазина).

Следующий шаг состоит в построении таблицы синтезируемого автомата по правилам, определяющим элементарные действия автомата в каждом состоянии.

1. Если состояние нумерует место в выражении непосредственно слева от какой-нибудь буквы входного алфавита (в нашем примере места 1, 4, 5, 6, 8, 9, 11), то осуществляется считывание этой буквы из входного магазина и переход к месту (состоянию) непосредственно справа от соответствующей буквы.

2. Если состояние нумерует место непосредственно слева от какой-нибудь буквы алфавита внутреннего магазина (места 1, 3,

7, 9), то осуществляется запись этой буквы в магазин и переход к самому левому начальному месту выражения, равному этой букве без индекса. Например, из места 1 переходим в место 9, записывая в магазин R_1, или из места 7 переходим в место 1, записывая в магазин S_2.

3. В самых правых (конечных) местах выражений (2 и 10) осуществляется считывание (верхней) буквы из внутреннего магазина и переход к месту, расположенному непосредственно справа от считанной буквы (благодаря индексации это место определяется однозначно). Если магазин пуст, соответствующий переход не определен и автомат должен остановиться. Заметим, что, как будет ясно из дальнейшего, в конечных местах выражения, равного какой-либо букве алфавита магазина (например, S), может считываться только та же буква с различными индексами. Для других букв алфавита магазина (в нашем примере R_1, R_2) соответствующий переход считается неопределенным.

Таблица 13

	1	2	3	4	5	6	7	8	9	10	11
a	3					7					
b				5				2			
c					6						
d					6						
x									10		10
y									10		10
S_1		4, S_1^{-1}									
S_2		8, S_2^{-1}									
R_1										2, R_1^{-1}	
R_2										11, R_2^{-1}	
	9, R_1		1, S_1				1, S_2		9, R_2		

Построим таблицу автомата для рассматриваемого примера (табл. 13). Автомат, задаваемый этой таблицей, будет, как легко видеть, недетерминированным, ибо в состояниях 1 и 9 не выполняется условие наличия лишь одного элементарного действия.

Во всех незаполненных местах таблицы соответствующие элементарные действия не определены.

Исходная система уравнений (6.2), по которой построена табл. 13, задает два языка: S и R. Автомат, заданный табл. 13, представляет оба языка. Для представления языка S в качестве начального состояния должно быть выбрано начальное место первого выражения (1), а в качестве заключительного состояния — конечное место этого выражения (2). Аналогично для представления языка R начальное состояние автомата должно быть отождествлено с начальным местом второго выражения, а заключительное — с его конечным местом (места соответственно 9 и 10). Это правило действует и при любых других системах уравнений.

Описанный алгоритм применим к любой конечной системе языков, заданных бэкусовскими нормальными формами. Нам осталось лишь доказать, что строящиеся с помощью этого алгоритма автоматы действительно представляют данные языки. Чтобы не усложнить систему обозначений, проведем доказательство на рассмотренном примере (метод доказательства непосредственно переносится и на общий случай).

Введем следующие обозначения: $S^{(0)}$- и $R^{(0)}$-события, $S^{(0)} = \varnothing \vee a \varnothing b (c \vee d) a \varnothing b = \varnothing$ и $R^{(0)} = x \vee y \vee \varnothing (x \vee y) = x \vee y$; при любом $i = 0, 1, 2, \ldots$

$$S^{(i+1)} = R^{(i)} \vee a S^{(i)} b (c \vee d) a S^{(i)} b,$$
$$R^{(i+1)} = x \vee y \vee R^{(i)} (x \vee y);$$

$P^{(k)}_{i,j}$ — язык, состоящий из всех слов, при восприятии которых построенный нами автомат (задаваемый табл. 13) осуществляет правильный переход из i-го состояния в j-е так, что максимальное число букв, дописываемых во внутренний магазин, ни в каком из промежуточных состояний не превышает k (в j-м состоянии в силу определения правильного перехода $k = 0$). Очевидно, что $P^{(0)}_{1,2} = S^{(0)}$ и $P^{(0)}_{9,10} = R^{(0)}$. Предположим, что уже доказана справедливость равенств $P^{(i)}_{1,2} = S^{(i)}$ и $P^{(i)}_{9,10} = R^{(i)}$ при всех $i = 0, 1, \ldots, \ldots, k-1, k$. Тогда

$$S^{k+1} = R^{(k)} \vee a S^{(k)} b (c \vee d) a S^{(k)} b = P^{(k)}_{9,10} \vee a P^{(k)}_{1,2} b (c \vee d) a P^{(k)}_{1,2} b,$$
$$R^{(k+1)} = x \vee y \vee R^{(k)} (x \vee y) = x \vee y \vee P^{(k)}_{9,10} (x \vee y).$$

Из описанных выше правил синтеза автомата непосредственно вытекает, что при восприятии любой цепочки p из события $P^{(k)}_{9,10} \vee \vee a P^{(k)}_{1,2} b (c \vee d) a P^{(k)}_{1,2} b$ автомат совершит правильный переход из состояния 1 в состояние 2, а в промежуточных состояниях заполнение магазина не превысит $k+1$ букв. Действительно, если $p \in P^{(k)}_{9,10}$, то при считывании этого слова в состоянии 1 автомат

должен избрать путь записи R_1 в магазин и перехода в состояние 9. После этого, согласно определению множества $P^{(k)}_{ij}$, автомат осуществит правильный переход в состояние 10, считав при этом слово p из входного магазина. Учитывая уже записанную в магазин букву R_1, заполнение магазина в промежуточных состояниях не превысит $k+1$, а в заключительное, 10-е, состояние автомат придет, имея в магазине лишь букву R_1. Считав эту букву, он перейдет в состояние 2 и, поскольку магазин окажется пустым, остановится. Таким образом, действительно осуществится требуемый правильный переход из состояния 1 в состояние 2.

Аналогично разбирается случай $p \in aP^{(k)}_{1,2}b\,(c \vee d)\,aP^{(k)}_{1,2}b$, т. е. показывается, что $P^{(k)}_{9,10} \vee aP^{(k)}_{1,2}b\,(c \vee d)\,aP^{(k)}_{1,2}b \subset P^{(k+1)}_{1,2}$. Точно таким же способом доказывается, что

$$x \vee y \vee P^{(k)}_{9,10}(x \vee y) \subset P^{(k+1)}_{9,10}.$$

Пусть p — любое слово из $P^{(k+1)}_{1,2}$. Если $p \overline{\in} P^{(k)}_{1,2}$, то по предположению индукции $p \in S^{(k)}$, а так как $S^{(k)} \subset S^{(k+1)}$, то $p \in S^{(k+1)}$. Если же $p \overline{\in} P^{(k)}_{1,2}$, то, поскольку $k+1 \geqslant 1$, при восприятии слова p автомат обязательно использует магазин. Выделим первую запись в магазин при считывании слова p. Из табл. 13 следует, что для этого есть две возможности: запись в магазин может произойти сразу в состоянии 1 (запишется R_1) или в состоянии 3 после предварительного считывания из входного магазина буквы a (во внутренний магазин запишется при этом буква S_1). Рассмотрим второй случай (первый случай рассматривается совершенно аналогично), т. е. $p = ap_1$. Так как при этом в магазин запишется буква S_1, а автомат, перейдя из состояния 3 в состояние 1, должен осуществить правильный переход (из состояния 1 в состояние 2), то в некотором состоянии эта запись должна быть стерта. Согласно табл. 13 это возможно лишь в состоянии 2. Следовательно, $p_1 = q_1p_2$, где $q_1 \in P^{(k)}_{1,2}$. После стирания S_1 из магазина автомат переходит в состояние 4. В этом состоянии автомат должен обязательно считать из входного магазина сначала букву b, затем c или d, после чего, считывая букву a, он попадает в состояние 7, где в магазин будет записана буква S_2, а автомат перейдет в состояние 1. Таким образом, $p_2 = bcap_3$ или $p_2 = bdap_3$.

Буква S_2 снова должна быть где-то считана. Поскольку это может произойти только в состоянии 2, $p_3 = q_2p_4$, где $q_2 \in P^{(k)}_{1,2}$. Но после считывания буквы S_2 автомат попадает в состояние 8, где он может считать лишь букву b из входного магазина и попасть в заключительное (2) состояние с пустым магазином. Так как автомат остановится, то слово p должно быть на этом исчерпано (ведь $p \in P^{(k+1)}_{1,2}$!). Следовательно, $p_4 = b$ и $p = aq_1bjaq_2b$, где $j = c$ или d. Так как $q_1 \in P^{(k)}_{1,2}$ и $q_2 \in P^{(k)}_{1,2}$, то $p \in P^{(k)}_{9,10} \vee aP^{(k)}_{1,2}b\,(c \vee d) \times aP^{(k)}_{1,2}b = S^{(k+1)}$.

Таким образом, $P^{(k+1)}_{1,2} \subset S^{(k+1)}$, откуда, учитывая ранее выведенное обратное включение, получаем $P^{(k+1)}_{1,2} = S^{(k+1)}$. Аналогично доказывается, что $P^{(k+1)}_{9,10} = R^{(k+1)}$. Тем самым индукция проведена. А так как объединение всех множеств S^k ($k = 0, 1, 2, \ldots$) есть S, а объединение $\bigcup_{k=0}^{\infty} P^{(k)}_{1,2} = P_{1,2}$ — совокупность входных слов, осуществляющих правильные переходы автомата из состояния 1 в состояние 2, то мы доказали, что язык S действительно представлен построенным автоматом. Доказательство (при выборе в качестве начального 9-го, а в качестве заключительного 10-го состояний) справедливо и для языка R. Тем самым полностью обоснована справедливость описанного выше алгоритма синтеза магазинного автомата.

Рассмотрим некоторые упрощения, позволяющие уменьшить число состояний магазинного автомата или число букв в алфавите его внутреннего магазина. Пусть некоторая буква записывается в магазин только в начальном состоянии автомата, переход же, как требуется алгоритмом синтеза, совершается опять в это же состояние. Считывается эта буква в конечном состоянии автомата. Предположим, что автомат переходит в то же конечное состояние (т. е. не изменяет состояния). Эту букву, как и сопутствующие ее записи и считыванию переходы (сохраняющие прежнее состояние), можно исключить из таблицы автомата, не изменив представляемые автоматом языки. Действительно, записанная в начальном состоянии, она не оказывает никакого влияния на выполняемые операции до тех пор, пока автомат не попадает в заключительное состояние. Теперь буква может оказать влияние только тогда, когда она находится в верхней ячейке магазина, но тогда она обязательно должна быть в этом состоянии считана. Поскольку же состояние при этом не изменяется, то все происходит так, как если бы эта буква не была записана в магазин.

Если произвести операцию объединения начальных состояний, то, как легко видеть, места справа от букв магазинного алфавита, первых в дизъюнктивных членах правых частей уравнения вида (6.3) или (6.4), можно отождествить с конечными местами выражений для соответствующих букв (без индексов). Если эти места, как, например, место справа от буквы R_1 в рассмотренном нами примере, были ранее отождествлены с другими состояниями, то их следует предварительно «разъединить» и затем объединить с новыми местами. При этом добавляется переход (без выполнения каких-либо других действий) из нового объединенного состояния в состояние, с которым рассматриваемое место было объединено

до сих пор. В нашем примере это приводит к следующей новой нумерации мест:

$$S = |_1(|_1 R_1|_9 \vee |_1 a|_2 S_1 |_3 b|_4(|_4 c|_5 \vee |_4 d|_5)|_5 a|_6 S_2 |_7 b|_8)|_8,$$

$$R = |_1(|_1 x|_9 \vee |_1 y|_9 \vee |_1 R_2|_9(|_9 x|_9 \vee |_9 y|_9)|_9)|_9.$$

Теперь буквы R_1 и R_2 будут записываться в магазин в состоянии 1 с сохранением состояния и считываться в состоянии 9 опять-таки с сохранением состояния. Как отмечалось, эти буквы могут быть исключены из алфавита магазина. В результате получим таблицу автомата, представляющего языки S и R (табл. 14). Язык S представляется состоянием 8, а язык R — состоянием 9. Начальное состояние для обоих языков — 1.

Таблица 14

	1	2	3	4	5	6	7	8	9
a	2				6				
b			4				8		
c				5					
d				5					
x	9								9
y	9								9
S_1								$3, S_1^{-1}$	
S_2								$7, S_2^{-1}$	
		$1, S_1$				$1, S_2$			8

Легко видеть, что описанным способом всегда можно устранить из алфавита магазина все буквы, являющиеся первыми буквами дизъюнктивных членов уравнений, задающих рассматриваемые языки.

Заметим, что хотя автомат, задаваемый табл. 14, недетерминированный, его нетрудно детерминизировать, если учесть, что переход из состояния 9 в состояние 8 следует выполнять лишь тогда, когда во входном магазине очередная считываемая буква не x или y. В противном случае автомат после двух шагов перейдет в состояние 3 или 7 и остановится, поскольку дальнейшие переходы будут не определены.

Таким образом, справедлива следующая теорема.

Теорема 6.2. *Всякий кс-язык может быть представлен автоматом (вообще говоря, недетерминированным) с одним внутренним магазином.*

Справедливо и обратное утверждение: одномагазинные автоматы представляют лишь кс-языки. Проблема анализа состоит в построении системы уравнений, порождающей язык, представленный данным автоматом [28]. Несколько видоизменив идею, предложенную А. А. Летичевским, опишем один из возможных простых алгоритмов анализа одномагазинного автомата (вообще говоря, недетерминированного). Для любой пары состояний (α, β) автомата A обозначим через $L_{\alpha,\beta}$ язык, представимый данным автоматом при выборе α в качестве начального и β — в качестве заключительного состояний.

Пусть M_α — множество состояний, в которые автомат может перейти из состояния α за один такт, не выполняя никаких записей или считываний; $M_{\alpha(a)}$ — множество всех состояний, в которые можно перейти за один такт из состояния α, считывая с входной ленты непустой символ a; $M_{\alpha,R}$ — множество всех состояний, к которым возможен переход из α за один такт, сопровождаемый записью во внутренний магазин непустого символа R (принадлежащего алфавиту данного магазина); N_R — множество состояний, в которых возможно считывание символа R из внутреннего магазина, а $N_{\gamma,R}$ — множество состояний, в которые можно попасть за один такт из состояния $\gamma \in N_R$. Наконец,

$$E(x_{\alpha,\beta}) = \begin{cases} \{e\}, & \text{если } \alpha = \beta, \\ \varnothing, & \text{если } \alpha \neq \beta, \end{cases}$$

где e — пустое слово; $\varnothing$ — пустой язык. На основании второго определения правильности переходов для любой пары (α, β) состояний рассматриваемого автомата A можно записать уравнение

$$x_{\alpha,\beta} = \bigvee_{\gamma \in M_\alpha} x_{\gamma,\beta} \vee \bigvee_{a \in \mathfrak{A}} \bigvee_{\gamma \in M_{\alpha(a)}} a x_{\gamma,\beta} \vee \bigvee_{R \in \mathfrak{M}} \bigvee_{\gamma \in M_{\alpha,R}} \vee$$

$$\bigvee_{\Delta \in N_R} \bigvee_{\delta \in N_{\Delta,R}} x_{\gamma,\Delta} x_{\delta,\beta} \vee E(x_{\alpha,\beta}),$$

где $\mathfrak{A}$ — входной алфавит автомата A, $\mathfrak{M}$ — алфавит внутреннего магазина (предполагается, что дизъюнкция пустого множества членов представляет собой пустое множество). Рассмотрим систему уравнений Q указанного типа, взятых по всевозможным парам состояний (α, β) автомата A. Тогда справедливо следующее утверждение.

Теорема 6.3. *Минимальное решение системы уравнений Q совпадает с системой языков $L_{\alpha,\beta}$ для любых пар состояний (α, β) автомата A.*

Для задания языка L, представимого автоматом A, посредством системы уравнений достаточно к системе Q присоединить еще одно уравнение: $x = \bigvee_{\beta \in F} x_{S_0,\beta}$, где S_0 — начальное состояние автомата A, F — множество его заключительных состояний. Рассмотренный алгоритм анализа одномагазинных автоматов может быть улучшен с помощью минимизации числа уравнений, входящих в систему Q.

Остановимся кратко на представлении языков n-магазинными автоматами ($n \geqslant 1$). Заметим, что представимость и порождение языков n-магазинными автоматами тесно взаимосвязаны: если существует n-магазинный автомат A (детерминированный или недетерминированный), порождающий некоторый язык L, то может быть построен n-магазинный автомат B (вообще говоря, недетерминированный), представляющий язык L. Действительно, переместим в табл. 14, задающей функционирование автомата A, все переходы из секции чтения входной ленты в секцию записи, а все переходы, связанные с записью на выходную ленту, — в секцию чтения входной ленты. Если переход из состояния α в состояние β сопровождался записью символа y на выходную ленту, то в секции чтения этот переход будет осуществляться лишь при условии чтения символа y на входе автомата B. При таких перемещениях, очевидно, не нарушится правильность переходов автомата A. В результате автомат B будет воспринимать те и только те цепочки, которые ранее порождались автоматом A.

В заключение приведем некоторые примеры представления языков n-магазинными автоматами.

Пример 1. Построим двухмагазинный автомат, представляющий нс-язык $L_1 = \{x \times x\}$ для любого $x \in F(\mathfrak{A})$, где $\mathfrak{A} = \{a, b\}$ (см. § 5.2). Функционирование этого автомата определено табл. 15. В начальном состоянии 1 автомат считывает с входного магазина символы a или b и переходит в состояние 2 или 3 соответственно, одновременно записывая во внутренний магазин I символ ε. В состояниях 2 и 3 записывает в магазин I символы A и B соответственно, переходя в состояние 4, повторяет цикл, связанный с записью в магазин I символов A и B при считывании с входного магазина соответствующих символов a или b и переходит в состояние 5 при считывании $\times$. Так, если во входной магазин записано слово $abb \times abb \in L_1$, то, переходя в состояние 5, автомат считает с входного магазина подцепочку $abb \times$, одновременно записав в магазин I цепочку $BBAE$. В состояниях 5—7 осуществляется перезапись из магазина I в магазин II, так что при считывании из магазина I символа ε автомат, опустошив магазин I

Таблица 15

		1	2	3	4	5	6	7	8	[9]	[10]
	a	2			2					8	
	b	3			3						8
	$\times$				5						
I	A					6, A^{-1}					
I	B					7, B^{-1}					
I	ε					8, ε^{-1}					
II	A								9, A^{-1}		
II	B								10, A^{-1}		
		ε (I)	4, A (I)	4, B (I)			5. A(II)	5. B (II)			

и записав в магазин II цепочку ABB, переходит в состояние 8. В состоянии 8 и заключительных состояниях 9 и 10 автомат читает из магазина II символы A и B и затем с входа — символы a и b соответственно, так что цепочка ABB стирается из магазина II, если на вход поступает abb. Построенный автомат детерминированный во всех состояниях, кроме начального 1, в котором одновременно выполняется чтение из входного магазина и запись в магазин I символа ε.

Пример 2. Автомат, функционирование которого определено табл. 16, представляет нс-язык (см. § 5.2) $L_2 = \{a^n b^n c^n \mid n = 1, 2, \ldots\}$. В начальном состоянии 1 автомат читает с входа символ a и, записывая в магазины I и II символ A, переходит в состояние 4. Затем этот цикл повторяется при чтении с входа символа a. При подаче на вход символа b автомат переходит в состояние 5, стирая из магазина I символ A. В состоянии 6 при поступлении b вновь происходит стирание A из магазина I, а при подаче c в заключительном состоянии 7 стирается A из магазина II. В состоянии 8 повторяется обращение к магазину II при чтении с входа символа c. Таким образом, правильный переход из состояния 1 в заключительное состояние 7 (связанный с опустошением магазинов I и II) осуществляется лишь при подаче на вход цепочки вида $a^n b^n c^n \in L_2$. Из табл. 16 следует, что построенный автомат детерминированный.

Пример 3. Снабдим автомат, построенный в примере 2, выходной лентой с алфавитом $Y = \{b\}$. Заменив в состоянии 5

Таблица 16

	1	2	3	4	5	6	7	8
a	2			2				
b				5		5		
c						7		7
I—A					6, A^{-1}			
II—B							8, A^{-1}	
		3, A (I)	4, A (II)					

(см. табл. 16) операцию чтения символа a из магазина I групповой операцией $[\wedge, A^{-1}, \wedge, b]$, помещенной в секцию записи с последующим переходом в состояние 6, получим автомат, реализующий трансляцию T: $a^nb^nc^n \to b^n$ при любом $n = 1, 2, \ldots$. Действительно, как и в примере 2, этот автомат представляет язык $\{a^nb^nc^n \mid n = 1, 2, \ldots\}$. В то же время он порождает язык $\{b^n \mid n = 1, 2, \ldots\}$, причем при заполнении входной ленты цепочкой $a^nb^nc^n$ осуществляет правильный переход из начального состояния 1 в заключительное 7, порождая выходную цепочку b^n.

Класс языков, представляемых n-магазинными автоматами ($n \geqslant 2$), совпадает с классом рекурсивно-перечислимых множеств. Это утверждение следует из возможности представления в автоматах с двумя внутренними магазинами средств известных алгоритмических систем, в частности машин Тьюринга [76, 154].

§ 6.3. Методы синтаксического анализа в системах программирования

Одним из основных блоков в современных системах программирования является анализатор — блок, осуществляющий синтаксический анализ входной программы.

Классификация методов синтаксического анализа, используемых при разработке анализаторов, проводится исходя из стратегии анализа, положенной в основу того или иного метода [104].

Наибольшее распространение получили методы синтаксического анализа, основанные на стратегии восходящего анализа, которая состоит в сведении анализируемой цепочки с помощью продукций данной грамматики к ее аксиоме. Методы этого класса называются **методами анализа сверткой.**

Рассмотрим, например, язык арифметических выражений, порожденный грамматикой с системой продукций

$$\sigma \to (\sigma \times \rho), \tag{6.5}$$

$$\sigma \to (a + b), \tag{6.6}$$

$$\rho \to b, \tag{6.7}$$

где σ — аксиома, $+$, $\times$, $(,)$, a, b — терминальные символы, σ, ρ — нетерминальные символы. С помощью стратегии свертки проанализируем цепочку

$$(((a + b) \times b) \times b). \tag{6.8}$$

Выделим в ней подцепочку $(a + b)$ и заменим ее как правую часть продукции (6.6) нетерминалом σ, который составляет левую часть этой продукции. Получим цепочку $((\sigma \times b) \times b)$, в которой на основании продукции (6.7) первое вхождение символа b заменяется нетерминалом ρ. Применив к цепочке $((\sigma \times \rho) \times b)$ продукцию (6.5), получим цепочку $(\sigma \times b)$. Наконец, вновь применив к полученной цепочке продукцию (6.7), а затем (6.5), свернем анализируемую цепочку в аксиому σ.

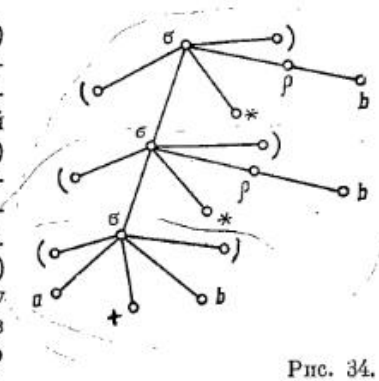


Рис. 34.

Таким образом, цепочка (6.8) правильна и ей соответствует дерево синтаксического анализа, представленное на рис. 34.

Рассмотренному классу двойственны методы **анализа развертокой**, основанные на стратегии нисходящего анализа — развертки. В противоположность анализу сверткой процесс развертки состоит в получении анализируемой цепочки из аксиомы данной грамматики, что соответствует построению вывода указанной цепочки. Так, в рассмотренном выше примере цепочка (6.8) может быть получена в результате развертки аксиомы σ с использованием продукций (6.5)—(6.7) данной грамматики:

$$\sigma \Rightarrow (\sigma \times \rho) \Rightarrow ((\sigma \times \rho) \times \rho) \Rightarrow (((a+b) \times \rho) \times \rho) \Rightarrow (((a+b) \times b) \times \rho) \Rightarrow$$
$$\Rightarrow (((a + b) \times b) \times b).$$

Методам анализа разверткой посвящены работы [296, 126, 142, 151, 152, 157, 160].

В остальных методах анализа обычно сочетаются элементы свертки и развертки. Заметим, что и для свертки, и для раз-

вертки существуют языки и грамматики, синтаксический анализ которых производится эффективнее методами, основанными на данной стратегии, чем двойственными методами.

Описанные выше стратегии анализа **левосторонние**, так как обработка символов цепочки производится слева направо. Однако, в отличие от свертки, развертка представляет собой целенаправленную стратегию, поскольку на каждом шаге анализа преследуется цель, состоящая в таком выборе применяемых продукций, чтобы к уже полученной начальной подцепочке можно было присоединить очередной символ или группу символов анализируемой цепочки. При анализе сверткой на каждом шаге обрабатывается самая левая подцепочка, которую можно свернуть, и более глубокие цели не ставятся.

Обычно «чистый» анализатор, основанный лишь на свертке или развертке, может использовать продукции, которые впоследствии приводят к тупикам, т. е. цепочка анализируется до конца и дальнейший анализ ее невозможен. Для преодоления тупиков, как правило, предусматривают возврат к некоторой метке, откуда можно выбрать новый вариант анализа. Такой возврат сопровождается восстановлением отдельных подцепочек анализируемой цепочки (при свертке) и стиранием некоторых присоединенных групп символов (при развертке). Другой метод преодоления тупиков связан с параллельным проведением всевозможных анализов, из которых исключаются тупиковые.

В связи с левосторонней обработкой анализируемых цепочек с точки зрения практических приложений значительный интерес представляют формализмы для описания языков, допускающие **беступиковый (беспереборный)** синтаксический анализ. При беступиковом анализе цепочки x появление хотя бы одного тупика свидетельствует о том, что $x \overline{\in} L(G)$, где $L(G)$ — язык, порожденный заданной грамматикой G. Однако в общем случае, как отмечалось выше, методы синтаксического анализа могут сопровождаться значительным перебором, так как из наличия одного тупика еще нельзя заключить, что $x \overline{\in} L(G)$. (В частности, нижняя оценка сложности алгоритмов синтаксического анализа кс-языков, как известно, равна n^2, где n — длина анализируемой цепочки [5]).

Стремление избежать попадания в тупик привело к созданию **модифицированных** анализаторов, в которых находят отражение элементы как свертки, так и развертки, вследствие чего происходит слияние обоих направлений в решении проблемы анализа, основанном на указанных стратегиях. Применение продукций на очередном шаге анализа модифицированные анализаторы связывают с проверкой некоторых контекстных условий, относящихся к предыстории анализа, а также к дальнейшим путям его продолжения.

кс-Грамматика $G(m, n)$ называется **грамматикой с (m, n) ограниченным контекстом** [144] тогда и только тогда, когда применение каждой ее продукции однозначно определено левым контекстом, состоящим из m символов, и правым контекстом из n символов. Кроме того, при левостороннем анализе цепочки на каждом шаге осуществляется поиск самой левой подцепочки, которая является правой частью некоторой продукции грамматики $G(m, n)$. Затем анализатор осуществляет свертку при условии, что m символов слева и n символов справа от выделенной подцепочки удовлетворяют соответствующим контекстам. В частности, грамматиками с (1.1) ограниченным контекстом являются грамматики предшествования (или старшинства разделителей), впервые рассмотренные Р. Флойдом [143].

кс-Грамматика $G = (\mathfrak{A}, V_н, \sigma, P)$ называется **операторной** или **грамматикой с разделителями**, если выполняется ограничение, состоящее в том, что в схеме P отсутствуют продукции вида $\psi \to z_1 \rho \tau z_2$, где $\psi, \rho, \tau \in V_н$; z_1, z_2 — произвольные цепочки (возможно, пустые) в алфавите $V = \mathfrak{A} \cup V_н$. Отсюда следует, что ни одна цепочка в алфавите V, выводимая из любого нетерминала $\psi \in V_н$, не может содержать двух рядом стоящих нетерминалов.

Следует заметить, что многие языки программирования имеют операторные грамматики или приводятся к ним путем несущественного изменения структуры языка.

На множестве всех терминалов операторной грамматики вводятся отношения предшествования $\doteq$, $\gtrdot$, $\lessdot$:

1) $a_i \doteq a_j$, если существует продукция $\psi \to z_1 a_i a_j z_2$ или $\psi \to z_1 a_i \tau a_j z_2$;

2) $a_i \gtrdot a_j$, если существует продукция $\psi \to z_1 \tau a_j z_2$, причем $\tau \overset{*}{\Rightarrow} z a_i$ или $\tau \overset{*}{\Rightarrow} z a_i \rho$;

3) $a_i \lessdot a_j$, если существует продукция $\psi \to z_1 a_i \tau z_2$, причем $\tau \overset{*}{\Rightarrow} a_j z$ или $\tau \overset{*}{\Rightarrow} \rho a_j z$; здесь $\psi, \tau, \rho \in V_н$; z_1, z_2, z — произвольные цепочки в алфавите V (возможно, пустые).

Между каждой упорядоченной парой терминальных символов может существовать от одного до трех отношений предшествования или не существовать ни одного (когда эти символы не встречаются рядом в цепочках языка, порождаемого данной грамматикой).

Операторная грамматика G называется **грамматикой предшествования** или **грамматикой старшинства разделителей**, если для каждой упорядоченной пары терминальных символов a_i, $a_j \in \mathfrak{A}$ существует не более одного отношения предшествования, а язык, порожденный такой грамматикой, — **языком предшествования** или **языком старшинства разделителей**.

Основная идея предложенного Р. Флойдом алгоритма анализа по отношениям предшествования состоит в следующем.

Рассмотрим цепочку $x = z_1 a_r z a_s z_2$ в объединенном алфавите V, причем терминальные символы в подцепочке z расположены в

порядке $a_{r_1}, a_{r_2}, \ldots, a_{r_n}$ $(n \geqslant 1)$. Пусть, далее, в цепочке выполняются следующие отношения предшествования:

$$a_r \lessdot a_{r_1} \doteq a_{r_2} \doteq \cdots \doteq a_{r_n} \gtrdot a_s$$

(нетерминальные символы, возможно встречающиеся в подцепочке z, здесь роли не играют). Тогда подцепочку z можно свернуть в некоторый символ ψ_k и перейти к цепочке $x' = z_1 a_r \psi_k a_s z_2$.

В алгоритме Флойда обычно используется магазин, в который записываются символы анализируемой цепочки в порядке их следования (слева направо) до тех пор, пока между a_{r_n} — верхним терминалом магазина и a_s — очередным терминалом данной цепочки не выполнится отношение $a_{r_n} \gtrdot a_s$. Затем с помощью отношений предшествования из верхних символов магазина выделяется подцепочка z, подлежащая свертке. Заменяя в магазине подцепочку z некоторым произвольно выбранным символом ψ_k, продолжаем анализ, сравнивая a_r с a_s. Всякая правильная цепочка языка $L(G)$ может быть свернута к произвольно выбранному нетерминальному символу ψ_r.

Отношения предшествования можно задавать с помощью квадратной матрицы размерности $m \times m$, где m — число терминалов грамматики L. Предшествование проверяется по матрице для метки столбца, соответствующей очередному символу анализируемой цепочки, и метки строки, соответствующей верхнему терминальному символу магазина.

На практике при составлении матрицы предшествования часто используются специальные множества левых $L(\psi)$ и правых $R(\psi)$ терминалов, соответствующие некоторому нетерминалу ψ данной грамматики, которые определяются рекурсивно: $L(\psi)$ — множество $\{a_i\}$ тех терминалов $a_i \in \mathfrak{A}$, для которых существует продукция $(\psi \to a_i x) \in P$ или $(\psi \to \rho a_i x) \in P$, либо схема P содержит продукцию $\psi \to \rho x$ такую, что $a_i \in L(\rho)$; аналогично $R(\psi)$ — множество $\{a_j\}$ тех терминалов $a_j \in \mathfrak{A}$, для которых существует продукция $(\psi \to x a_j) \in P$ или $(\psi \to x a_j \rho) \in P$, либо схема P имеет продукцию $\psi \to x\rho$ такую, что $a_j \in R(\rho)$ (x — цепочка, возможно пустая, в объединенном алфавите $V = \mathfrak{A} \cup V_н$, $\rho \in V_н$).

Для предварительного контроля анализируемой цепочки часто используют также матрицы возможных сочетаний терминальных символов в цепочках языка.

Пример 1. Пусть

$$G_1 = (\mathfrak{A}, V_н, \sigma, P_1),$$

$$\mathfrak{A} = \{\#, (, I, \times, +,)\},\ V_н = \{\sigma, \alpha, \beta, \gamma\},$$

$$P_1 = \{\sigma \to \# \alpha \#,\ \alpha \to \alpha + \beta,\ \alpha \to \beta,\ \beta \to \beta \times \gamma,$$
$$\beta \to \gamma,\ \gamma \to (\alpha),\ \gamma \to I\}.$$

С помощью вспомогательных множеств левых L и правых R терминальных символов для каждого нетерминала грамматики

$$\begin{aligned} L(\gamma) &= \{(, I\}, & R(\gamma) &= \{), I\}, \\ L(\beta) &= \{(, I, \times\}, & R(\beta) &= \{), I, \times\}, \\ L(\alpha) &= \{(, I, \times, +\}, & R(\alpha) &= \{), I, \times, +\}, \\ L(\sigma) &= \{\#\}, & R(\sigma) &= \{\#\} \end{aligned}$$

и схемы P_1 данной грамматики составим матрицу отношений предшествования. Так как $\beta \to \beta \times \gamma \in P_1$ и $) \in R(\beta)$, то $) \gtrdot \times$. Далее, из продукций $\alpha \to \alpha + \beta$, $\gamma \to (\alpha)$, $\sigma \to \# \alpha \#$, а также из $) \in R(\alpha)$ заключаем, что $) \gtrdot +$, $) \gtrdot)$, $) \gtrdot \#$. Аналогично из приведенных выше продукций, поскольку $I \in R(\beta)$ и $I \in R(\alpha)$, следует заполнение второй строки матрицы. Кроме того, $\times \in R(\beta)$ и $\times \in R(\alpha)$, значит, $\times \gtrdot \times$, $\times \gtrdot +$, $\times \gtrdot)$ и $\times \gtrdot \#$. В то же время из $\beta \to \beta \times \gamma$ и $L(\gamma) = \{(, I\}$ следует, что $\times \lessdot ($ и $\times \lessdot I$. Точно так же из $\alpha \to \alpha + \beta$ и $L(\beta) = \{(, I, \times\}$ заключаем, что $+ \lessdot ($, $+ \lessdot I$, $+ \lessdot \times$. А из $+ \in R(\alpha)$ и $\gamma \to (\alpha)$, $\sigma \to \# \alpha \#$ следует $+ \gtrdot +$. Далее, $(\doteq)$ на основании $\gamma \to (\alpha)$, кроме того, из $L(\alpha) = \{(, I, \times, +\}$ вытекает, что $(\lessdot ($, $(\lessdot I$, $(\lessdot \times$, $(\lessdot +$. Наконец, $\# \doteq \#$ в силу $\sigma \to \# \alpha \#$. Учитывая также $L(\alpha)$, получаем $\# \lessdot ($, $\# \lessdot I$, $\# \lessdot \times$ и $\# \lessdot +$. Матрица отношений предшествования имеет вид

	(	I	$\times$	$+$	)	#
)			$\gtrdot$	$\gtrdot$	$\gtrdot$	$\gtrdot$
I			$\gtrdot$	$\gtrdot$	$\gtrdot$	$\gtrdot$
$\times$	$\lessdot$	$\lessdot$	$\gtrdot$	$\gtrdot$	$\gtrdot$	$\gtrdot$
$+$	$\lessdot$	$\lessdot$	$\lessdot$	$\gtrdot$	$\gtrdot$	$\gtrdot$
(	$\lessdot$	$\lessdot$	$\lessdot$	$\lessdot$	$\doteq$	
#	$\lessdot$	$\lessdot$	$\lessdot$	$\lessdot$		$\doteq$

Проанализируем с помощью алгоритма Флойда цепочку $\# (I + I) \times I \#$, принадлежащую языку $L(G_1)$:

Шаг	Текущая цепочка	Выделенная подцепочка, подлежащая свертке
1	$\# (I + I) \divideontimes I \#$	I
2	$\# (\psi_1 + I) \divideontimes I \#$	I
3	$\# (\psi_1 + \psi_2) \divideontimes I \#$	$\psi_1 + \psi_2$
4	$\# (\psi_3) \divideontimes I \#$	(ψ_3)
5	$\# \psi_4 \divideontimes I \#$	I
6	$\# \psi_4 \divideontimes \psi_5 \#$	$\psi_4 \divideontimes \psi_5$
7	$\# \psi_6 \#$	$\# \psi_6 \#$
8	ψ_7	

При анализе цепочки используем магазин. Вначале левый символ # исходной цепочки заносим в магазин. По матрице предшествования $\# \lessdot ($, где # — терминал в вершине магазина, (— очередной символ анализируемой цепочки. Это означает, что символ (необходимо записать в магазин. Далее, $(\lessdot I$, где I — очередной символ цепочки, поэтому его также заносим в магазин. Наконец, $I \gtrdot +$, так что $(\lessdot I \gtrdot +$. Таким образом, левое вхождение символа I в исходную цепочку подлежит свертке на первом шаге.

Заменяя I в вершине магазина произвольно выбранным нетерминалом ψ_1, продолжаем процесс анализа. Сравнивая по матрице предшествования (— верхний, терминальный символ в магазине и + — очередной символ цепочки, находим $(\lessdot +$, поэтому символ + должен быть записан в магазин. По тем же соображениям очередной символ I также заносим в магазин. Далее, $I \gtrdot)$, так что выделенное вхождение символа I подлежит свертке на втором шаге. Для очередного символа) анализируемой цепочки выполняется отношение $+ \gtrdot)$, где + — верхний терминал магазина, так что на третьем шаге подлежит свертке подцепочка $\psi_1 + \psi_2$ и т. д. Наконец, на последнем шаге сворачиваем цепочку $\# \psi_6 \#$ в нетерминал ψ_7.

Таким образом, выбранная цепочка языка $L(G_1)$ может быть проанализирована с помощью алгоритма Флойда.

В то же время алгоритм Флойда из-за отсутствия механизма, устанавливающего связь между символами ψ_k, вырабатываемыми в процессе свертки, и нетерминалами исходной грамматики, может осуществлять свертку неправильных цепочек. Так, в приведенном примере алгоритм успешно сворачивает цепочку $(+I)$, которая не принадлежит языку $L(G_1)$. Ненадежность синтаксического контроля, который составляет часть синтаксического анализа, является основным недостатком алгоритма Флойда. Однако метод предшествования — достаточно гибкое, эффективное и простое средство анализа. Он применяется при разработке трансляторов, в частности двухпроходовых, которые при первом проходе осуществляют синтаксический контроль входной программы и собирают сведения о переменных, блоках и других объектах, а при втором, эффективно используя отношения предшествования и собранные при первом проходе сведения, производят синтаксический анализ и одновременно составляют машинную программу.

Метод Флойда получил дальнейшее развитие в работах Н. Вирта и Х. Вебера [159], в которых отношения предшествования распространены не только на терминальные, но и на нетерминальные символы кс-грамматики. Тем самым снимается требование, состоящее в том, чтобы анализируемые грамматики были операторными. По Н. Вирту и Х. Веберу отношения предшествования $\doteq$, $\gtrdot$, $\lessdot$ определяются следующим образом:

1) $S_i \doteq S_j$, если существует продукция $\psi \to z_1 S_i S_j z_2$;

2) $S_i \gtrdot S_j$, если существует продукция $\psi \to z_1 \tau_1 S_j z_2$ или $\psi \to z_1 \tau_1 \tau_2 z_2$ и вывод $\tau_1 \overset{*}{\Rightarrow} z S_i$ (и $\tau_2 \overset{*}{\Rightarrow} S_j w$);

3) $S_i \lessdot S_j$, если существуют продукция $\psi \to z_1 S_i \tau_1 z_2$ и вывод $\tau_1 \overset{*}{\Rightarrow} S_j z$.

Здесь $S_i, S_j \in V = \mathfrak{A} \cup V_н$; z, z_1, z_2, w — некоторые цепочки в объединенном алфавите V; $\psi, \tau_1, \tau_2 \in V_н$.

кс-Грамматика $G = (\mathfrak{A}, V_н, \sigma, P)$ называется **грамматикой предшествования по Вирту и Веберу** или **грамматикой старшинства**, если для любой упорядоченной пары символов из объединенного алфавита $V = \mathfrak{A} \cup V_н$ выполняется не более одного отношения предшествования типа $\doteq$, $\lessdot$, $\gtrdot$, причем никакие продукции, принадлежащие схеме P данной грамматики, не имеют одинаковых правых частей. Каждая цепочка языка такой грамматики имеет единственное дерево синтаксического анализа, и поэтому грамматики данного класса однозначны. При анализе, как и в алгоритме Флойда, символы анализируемой цепочки в порядке их обработки (слева направо) записываются в магазин до тех пор, пока не выполнится отношение $S_{r_n} \gtrdot S_j$, где S_{r_n} — символ, записанный в вершине магазина, S_j — очередной символ данной цепочки. После этого среди символов $S_{r_n}, S_{r_{n-1}}, \ldots, S_{r_1}, S_i$, записанных в верхней части магазина, выделяется подцепочка $S_{r_n} S_{r_{n-1}} \ldots S_{r_1}$, подлежащая свертке, где $S_{r_n} \doteq S_{r_{n-1}} \doteq \cdots \doteq S_{r_1}$ и $S_i \lessdot S_{r_1}$. В этом случае выделенная подцепочка является правой частью единственной продукции $\psi \to S_{r_n} S_{r_{n-1}} \ldots S_{r_1}$ и может быть заменена левой частью данной продукции.

В алгоритме Вирта — Вебера, как и в алгоритме Флойда, используются матрица предшествования и матрицы возможных сочетаний, которые, однако, задают отношения предшествования не только для терминалов, но и для всех символов объединенного алфавита V и поэтому более громоздки. При этом, как и в алго-

ритме Флойда, используются множества всех левых $L(\psi)$ и правых $R(\psi)$ символов для некоторого нетерминала ψ данной грамматики, которые определяются следующим образом: $L(\psi) = \{D|$, если $D \in V$ и существует продукция $(\psi \to Dz) \in P$ или $(\psi \to \rho z) \in P$, причем $D \in L(\rho)\}$, $R(\psi) = \{D|$, если $D \in V$ и существует продукция $(\psi \to \to zD) \in P$ или $(\psi \to z\rho) \in P$, причем $D \in R(\rho)\}$ (z — цепочка, возможно пустая, в объединенном алфавите $V = \mathfrak{A} \cup V_н$; $\psi, \rho \in V_н$).

Пример 2. Пусть

$$G_2 = (\mathfrak{A}, V_н, \sigma, P_2),$$

$$\mathfrak{A} = \{\#, a, b, c\},\ V_н = \{\sigma, \alpha, \beta, \gamma\},$$

$$P_2 = \{\sigma \to \# \alpha \#,\ \alpha \to \beta,\ \beta \to \beta\gamma,\ \beta \to \gamma,\ \gamma \to a,$$
$$\gamma \to b,\ \gamma \to c\}.$$

Язык $L(G_2)$, порожденный этой грамматикой, содержит любую цепочку, состоящую из терминальных символов a, b, c и ограниченную символами $\#$. Множества левых L и правых R символов имеют вид

$$L(\sigma) = \{\#\}, \qquad R(\sigma) = \{\#\},$$
$$L(\alpha) = \{\beta, \gamma, a, b, c\}, \qquad R(\alpha) = \{\beta, \gamma, a, b, c\},$$
$$L(\beta) = \{\beta, \gamma, a, b, c\}, \qquad R(\beta) = \{\gamma, a, b, c\},$$
$$L(\gamma) = \{a, b, c\}, \qquad R(\gamma) = \{a, b, c\}.$$

Как и в примере 1, используя множества левых и правых символов, по схеме P_2 данной грамматики заполняем матрицу отношений предшествования:

	α	β	γ	c b a	$\#$
α					$\doteq$
β			$\doteq$	$\lessdot$	$\gtrdot$
γ			$\gtrdot$	$\gtrdot$	$\gtrdot$
a, b, c			$\gtrdot$	$\gtrdot$	$\gtrdot$
$\#$	$\doteq$	$\lessdot$	$\lessdot$	$\lessdot$	

При синтаксическом анализе цепочки $\# cba \#$ согласно алгоритму Вирта — Вебера выполняем следующие шаги сворачивания:

Шаг	Текущая цепочка	Выделенная подцепочка, подлежащая свертке
1	$\# cba \#$	c
2	$\# \gamma ba \#$	γ
3	$\# \beta ba \#$	b
4	$\# \beta\gamma a \#$	$\beta\gamma$
5	$\# \beta a \#$	a
6	$\# \beta\gamma \#$	$\beta\gamma$
7	$\# \beta \#$	β
8	$\# \alpha \#$	$\# \alpha \#$
9	σ	—

Подобно алгоритму Флойда, подцепочки, подлежащие свертке, выделяем по матрице предшествования, используя магазин. Вначале в магазин засылаем символ $\#$, затем, так как $\# \lessdot c$, и символ c. Далее, $c \gtrdot b$, так что в вершине магазина выделено вхождение символа c, подлежащее свертке на первом шаге. По схеме P_2 данной грамматики находим единственную продукцию $\gamma \to c$, правая часть которой совпадает с выделенной подцепочкой, и заменяем в вершине магазина c символом γ. Далее, $\gamma \gtrdot b$, так что вхождение γ также подлежит свертке. В силу $(\beta \to \gamma) \in P_2$ заменяем в вершине магазина γ символом β. Затем, поскольку $\beta \lessdot b$, символ b засылаем в магазин и аналогично предыдущему заменяем его символом γ. Так как $\gamma \gtrdot a$ и $\beta \doteq \gamma$, в верхних ячейках магазина оказалась выделенной подцепочка $\beta\gamma$, подлежащая свертке на четвертом шаге. Продолжая этот процесс, сворачиваем цепочку $\# cba \# \in L(G_2)$ в аксиому σ данной грамматики, что подтверждает правильность исходной цепочки.

Нетрудно видеть, что грамматика G_2 не операторна и поэтому не может быть проанализирована методом Флойда.

С теоретической точки зрения алгоритм Вирта — Вебера позволяет проводить достаточно эффективный анализ и контроль программ. Однако требование о несовпадении правых частей различных продукций существенно сужает класс языков, которые могут быть проанализированы с помощью этого алгоритма.

Метод отношений предшествования, предложенный Р. Флойдом и получивший дальнейшее развитие в работах Н. Вирта и Х. Вебера, применим для анализа лишь кс-языков. В работе [101] предлагается обобщение этого метода на произвольные порождающие грамматики Хомского.

Пусть $G = (\mathfrak{A}, V_н, \sigma, P)$ — произвольная порождающая грамматика и B — некоторый символ из объединенного алфавита $V = \mathfrak{A} \cup V_н$. Для символа B определим множество самых левых $L(B)$ и самых правых $R(B)$ символов из V:

$L(B) = \{D \mid$, если $D \in V$ и существует продукция $(Bx \to Dz) \in P$ или продукция $(Bx \to B_1 z) \in P$, причем $D \in L(B_1)\}$, (6.9)

$R(B) = \{D \mid$, если $D \in V$ и существует продукция $(xB \to zD) \in P$ или продукция $(xB \to zB_1) \in P$, причем $D \in R(B_1)\}$ (6.10)

(x, z — некоторые цепочки, возможно, пустые, в алфавите V; B, B_1, $D \in V$). Из (6.9) и (6.10) следует, что терминалы и нетерминалы могут иметь непустое множество левых (правых) символов при условии, что они использовались как начальные (конечные) символы в левых частях продукций, входящих в схему P.

Отношения предшествования для порождающих грамматик общего типа определим следующим образом:

1) $S_i \doteq S_j$, если существует продукция $(z \to z_1 S_i S_j z_2) \in P$;

2) $S_i \lessdot S_j$, если существует продукция $(z \to z_1 S_i N z_2) \in P$ и $S_j \in L(N)$;

3) $S_i \gtrdot S_j$, если существует продукция $(z \to z_1 N S_j z_2) \in P$ и $S_i \in R(N)$ или найдется продукция $(z \to z_1 N_1 N_2 z_2) \in P$, причем $S_i \in R(N_1)$, а $S_j \in L(N_2)$ (z, z_1, z_2 — некоторые цепочки, возможно пустые, в алфавите V; $N, N_1, N_2 \in V$).

Порождающая грамматика $G = (\mathfrak{A}, V_н, \sigma, P)$ называется **грамматикой предшествования Вирта — Вебера**, если каждая упорядоченная пара символов объединенного алфавита V имеет не более одного отношения предшествования, а язык, порожденный такой грамматикой, — **языком предшествования Вирта — Вебера.**

Пример 3. Пусть

$$G_3 = (\mathfrak{A}, V_н, \sigma, P_3),$$

$$\mathfrak{A} = \{a, b\},\ V_н = \{\sigma, \alpha, \beta, \alpha_1, \beta_1\},$$

$$P_3 = \{p_1 : \sigma \to \alpha_1 \sigma \alpha,\ p_2 : \sigma \to \alpha_1 \beta_1 a,$$

$$p_3 : a\alpha \to \beta\alpha,\ p_4 : \beta \to \alpha,$$

$$p_5 : \beta_1 \to b,\ p_6 : b\alpha \to b\beta_1 a,\ p_7 : \alpha_1 \to a\}.$$

Грамматика G_3 порождает язык $L(G_3) = \{a^n b^n a^n \mid n = 1, 2, \ldots\}$, который не является кс-языком (см. § 5.3). Запишем множества самых левых и правых символов этой грамматики:

$$L(\sigma) = \{\alpha_1, a, \beta, \alpha\}, \qquad R(\sigma) = \{\alpha, a\},$$
$$L(\beta_1) = \{b\}, \qquad R(\beta_1) = \{b\},$$
$$L(\alpha_1) = \{a, \beta, \alpha\}, \qquad R(\alpha_1) = \{a\},$$
$$L(\beta) = \{\alpha\}, \qquad R(\beta) = \{\alpha, a\}$$
$$L(\alpha) = \{a, b, \alpha\}, \qquad R(\alpha) = \{a\},$$
$$L(a) = \{\alpha, \beta\}, \qquad R(a) = -,$$
$$L(b) = \{b\}, \qquad R(b) = -.$$

Матрица отношений предшествования имеет вид

	σ	β_1	α_1	β	α	a	b
σ					$\doteq$		
β_1				$\lessdot$	$\lessdot$	$\doteq$	
α_1	$\doteq$	$\doteq$	$\lessdot$	$\lessdot$	$\lessdot$	$\lessdot$	$\lessdot$
β				$\lessdot$	$\lessdot$	$\doteq$	
α				$\gtrdot$	$\gtrdot$	$\gtrdot$	
a	$\gtrdot$	$\gtrdot$	$\gtrdot$	$\gtrdot$	$\gtrdot$	$\gtrdot$	$\gtrdot$
b		$\doteq$		$\gtrdot$	$\gtrdot$	$\gtrdot$	$\lessdot$

Проиллюстрируем процесс свертки на примере выводимой цепочки $(aabbaa) \in L(G_3)$; при этом используем в приведенном порядке продукции $p_7, p_7, p_5, p_6, p_5, p_4, p_3, p_2, p_1$.

Шаг	Текущая цепочка	Выделенная подцепочка, подлежащая свертке
1	$aabbaa$	a
2	$\alpha_1 abbaa$	a
3	$\alpha_1\alpha_1 bbaa$	b
4	$\alpha_1\alpha_1 b\beta_1 aa$	$b\beta_1 a$
5	$\alpha_1\alpha_1 baa$	b
6	$\alpha_1\alpha_1\beta_1 aa$	a
7	$\alpha_1\alpha_1\beta_1\beta a$	βa
8	$\alpha_1\alpha_1\beta_1 aa$	$\alpha_1\beta_1 a$
9	$\alpha_1\sigma a$	$\alpha_1\sigma a$
10	σ	—

Подцепочки, подлежащие свертке, выделяем по матрице предшествования, используя магазин, как и по алгоритмам Флойда и Вирта — Вебера (см. примеры 1, 2). Единственное отличие состоит в том, что при свертке правых частей не контекстно-свободных продукций p_3 и p_6 в вершину магазина засылаются соответствующие левые части.

На основе описанного подхода в работе [101] предложены эффективные алгоритмы анализа, обобщающие алгоритмы Флойда и Вирта — Вебера на порождающие грамматики общего типа. Показано, что произвольная порождающая грамматика с помощью эквивалентных преобразований может быть приведена к грамматике предшествования, и разработан алгоритм такого перехода.

Более широкий класс кс-языков, допускающих эффективный анализ сверткой, описывают **$LR(k)$-грамматики**, предложенные Д. Кнутом [150].

кс-Грамматика $G = (\mathfrak{A}, V_н, \sigma, P)$ называется $LR(k)$-грамматикой, если применение каждой продукции из P в процессе свертки однозначно определяется всей левой частью анализируемой цепочки и правыми очередными k терминальными символами; язык, порожденный $LR(k)$-грамматикой, называется **$LR(k)$-языком**. Так, при анализе цепочки методом $LR(k)$-грамматик с использованием магазина для выделения цепочки, подлежащей свертке, необходимо просматривать весь магазин (а не только фиксированное число символов в нем, как, например, в случае грамматик предшествования) и следующие k терминальных символов анализируемой цепочки. Д. Кнут доказал, что для каждого кс-языка, воспринимаемого детерминированным магазинным автоматом, существует $LR(k)$-грамматика, порождающая этот же язык. Более того, для любой $LR(k)$-грамматики существует эквивалентная $LR(1)$-грамматика. Разработан также алгоритм для проверки, является ли данная грамматика $LR(k)$-грамматикой при фиксированном k, причем в положительном случае строится детерминированный магазинный автомат, воспринимающий данный язык.

Таким образом, наличие $LR(k)$-грамматики является наиболее общей характеристикой класса однозначных языков в терминах порождающих грамматик.

Переходя к методам анализа разверткой, заметим, что в этом направлении получены менее значительные результаты и его развитие было не столь интенсивным по сравнению с методами анализа сверткой.

К наиболее распространенным и эффективным методам беступикового анализа разверткой относится метод деления на слово с одноименной операцией над языком (см. § 5.5).

Пусть заданы некоторая цепочка $x \in F(\mathfrak{A})$ и исходная кс-грамматика $G = (\mathfrak{A}, V_н, \sigma, P)$, допускающая беступиковый синтаксический анализ. В качестве начальной рассмотрим пару (x, σ), где x — анализируемая цепочка, σ — аксиома грамматики G. Допустим, что на некотором шаге алгоритма анализа делением на слово была получена пара (x_i, z_i), где x_i — конечная подцепочка цепочки x, $z_i = \psi v_i$, причем v_i — цепочка (возможно, пустая) над объединенным алфавитом $V = \mathfrak{A} \cup V_н$. Тогда переход к паре (x_{i+1}, z_{i+1}) состоит в следующем. Среди продукций, входящих в схему P грамматики G, находим продукцию $\psi \to w$, применение которой к выделенному символу ψ цепочки z_i порождает цепочку $wv_i = u_{i+1}\tau v_{i+1}$ либо $wv_i = u_{i+1}$, где u_{i+1} — начальная подцепочка (возможно, пустая) цепочки x_i. Если указанная продукция не существует, то получаем тупик. Согласно выбору грамматики G это означает, что анализ цепочки x завершен, причем $x \overline{\in} L(G)$. Если

$wv_i = u_{i+1}\tau v_{i+1}$, то сокращая цепочки x_i и wv_i слева[4] на цепочку u_{i+1}, получаем очередную пару $(x_{i+1}, \tau v_{i+1})$. Если $x_{i+1} = e$, анализ вновь завершен и $x \overline{\in} L(G)$. Если $wv_i = u_{i+1}$ и в результате описанного выше сокращения получена пара (e, e), анализ завершен успешно, так что $x \in L(G)$. В противном случае вновь получаем тупик и $x \overline{\in} L(G)$.

Метод деления на слово допускает естественную магазинную интерпретацию с использованием двух магазинов, причем на i-м шаге алгоритма первый из них хранит компоненту x_i пары (x_i, z_i), полученной на данном шаге, а второй (магазин вывода) — компоненту z_i этой пары. На первом шаге алгоритма первый магазин содержит анализируемую цепочку, которая в процессе анализа как бы стирается по частям. Магазин вывода на первом шаге хранит σ-аксиому грамматики и в процессе анализа моделирует левый вывод данной цепочки.

Таким образом, возникает проблема выделения классов грамматик, ориентированных на беступиковый детерминированный анализ разверткой с применением метода деления на слово. Заметим, что левый вывод в таких грамматиках управляется остатком анализируемой цепочки, находящимся справа от текущего символа, и состоянием магазина вывода.

Важным классом однозначных кс-грамматик, допускающих беступиковый анализ разверткой, являются $LL(k)$-грамматики [152].

Пусть $G = (\mathfrak{A}, V_н, \sigma, P)$ — некоторая кс-грамматика и $\tilde{\sigma} \overline{\in} V_н$ — новый нетерминальный символ. Рассмотрим кс-грамматику $\tilde{G} = (\tilde{\mathfrak{A}}, \tilde{V}_н, \tilde{\sigma}, \tilde{P})$, где $\tilde{\mathfrak{A}} = \mathfrak{A} \cup \{\#\}$, $\tilde{V}_н = V_н \cup \{\tilde{\sigma}\}$, схема $\tilde{P}$ получается в результате присоединения к P дополнительной продукции $\tilde{\sigma} \to \sigma \#^k$; очевидно, $L(\tilde{G}) = L(G)\#^k$. Грамматика $\tilde{G}$ называется **$LL(k)$-грамматикой** ($k \geqslant 0$), если на любом шаге левостороннего вывода применение каждой продукции из $\tilde{P}$ однозначно определено терминальным началом строки, полученной на данном шаге вывода (история вывода), и k очередными терминалами, появляющимися при продолжении вывода. Иными словами, если $(\tilde{\sigma}, \ldots, u\psi v, uzv, \ldots, uw \ldots)$ и $(\tilde{\sigma}, \ldots, u\psi v', uz'v', \ldots, uw \ldots \ldots)$ — левые выводы в $LL(k)$-грамматике $\tilde{G}$, где u, w — терминальные цепочки, причем $|w| = k$, то $z = z'$, т. е. на выделенном шаге примененная продукция $(\psi \to z) \in \tilde{P}$ однозначно определена терминальным началом «u» строки $u\psi v$ и последующей терминальной цепочкой w длины k. Язык $L(\tilde{G})$, порожденный $LL(k)$-грамматикой $\tilde{G}$, называется **$LL(k)$-языком**.

[4] Под левым сокращением цепочек z и z' на цепочку u понимается одновременное деление слева каждой из этих цепочек на цепочку u.

Синтаксический анализ разверткой с помощью $LL(k)$-грамматик состоит в построении левого вывода анализируемой цепочки $x \in F(\mathfrak{A})$, каждый шаг которого удовлетворяет следующим условиям:

1) терминальное начало u строки $u\psi v$, полученной на данном шаге вывода, совпадает с некоторым началом цепочки x;

2) следующие k терминалов в x составляют цепочку, которая может появиться непосредственно в процессе вывода, если на данном шаге применить соответствующую продукцию $(\psi \to z) \in \tilde{P}$.

Если исследуемые k символов не совпадают ни с одной из таких цепочек, то $x \overline{\in} L(\tilde{G})$.

Заметим, что $LL(k)$-грамматики не вполне удобны для проведения анализа разверткой, поскольку на каждом шаге необходимо хранить терминальное начало «u» строки левого вывода, несмотря на то, что оно может быть сколь угодно большой длины. В этом отношении для анализа разверткой гораздо удобнее использовать строгие $LL(k)$-грамматики, предложенные в работах [151, 157].

Строгой $LL(k)$-грамматикой называется грамматика $\tilde{G}$, для которой на любом шаге левого вывода применение каждой продукции из $\tilde{P}$ однозначно определено очередными k терминалами, появляющимися при продолжении вывода. Иными словами, если $(\tilde{\sigma}, \ldots, u\psi v, uzv, \ldots, uw \ldots)$ и $(\tilde{\sigma}, \ldots, u_1\psi v_1, u_1z'v_1, \ldots, u_1w \ldots)$ — левые выводы в строгой $LL(k)$-грамматике $\tilde{G}$, где u_1, w — терминальные цепочки, причем $|w| = k$, то $z = z'$, т. е. на выделенном шаге примененная продукция $(\psi \to z) \in \tilde{P}$ однозначно определена терминальной цепочкой w длины k, идущей вслед за терминальным началом u строки вывода, полученной на данном шаге.

Пример 4. Рассмотрим грамматику, заданную схемой

$$\tilde{P}_1 = \begin{cases} p_1 : \tilde{\sigma} \to \sigma\#, & p_4 : \tau \to b\tau, \\ p_2 : \sigma \to a\sigma, & p_5 : \tau \to b. \\ p_3 : \sigma \to a\tau, & \end{cases}$$

Очевидно, эта грамматика порождает язык $L = \{a^n b^m \# \mid m, n = 1, 2, \ldots\}$. Покажем, что грамматика со схемой $\tilde{P}_1$ является строгой $LL(2)$-грамматикой. Действительно, продукция p_1 всегда применяется лишь в начале вывода, независимо от последующих терминалов, и поэтому удовлетворяет $LL(0)$-свойству, а значит, и $LL(1)$- и $LL(2)$-свойствам. Вообще говоря, из определения $LL(k)$-грамматики следует, что всякая продукция, обладающая $LL(k)$-свойством, удовлетворяет также свойству $LL(k+1)$. Продукция p_2 применяется всегда, когда очередными терминалами выводимой цепочки являются aa, продукция p_3 — когда очередными терминалами являются ab, так что p_2 и p_3 обладают свойством $LL(2)$. Аналогично продукция p_4 применяется всегда, когда

очередные терминалы — bb, продукция p_5 — при очередных терминалах $b\#$.

Таким образом, приведенная грамматика является $LL(2)$-грамматикой и, кроме того, строгой $LL(2)$-грамматикой, так как применение каждой продукции зависит от очередной пары терминалов и не зависит от истории вывода.

По определению каждая строгая $LL(k)$-грамматика является просто $LL(k)$-грамматикой. Однако обратное утверждение неверно: существуют нестрогие $LL(k)$-грамматики.

Пример 5. Рассмотрим грамматику, заданную схемой

$$\tilde{P}_2 = \begin{cases} p_1 : \tilde{\sigma} \to \sigma\#\#^3, & p_4 : \tau \to \psi b, & p_7 : \psi \to b, \\ p_2 : \sigma \to \tau cb, & p_5 : \tau \to a\psi cb, & p_8 : \psi \to c, \\ p_3 : \sigma \to d\tau bc, & p_6 : \psi \to e, & p_9 : \psi \to a\tau dd\psi \end{cases}$$

(e — пустое слово). Приведенная грамматика является $LL(3)$-грамматикой, так как продукция p_1 удовлетворяет $LL(0)$-свойству, $p_2 - p_5$, p_9 — $LL(1)$-свойству, а $p_6 - p_8$ — $LL(3)$-свойству. Действительно, предположим, что получена строка вывода с самым левым нетерминалом ψ. Применяя продукцию p_6, в дальнейшем можем получить цепочку cbb (если на предыдущих шагах применялись продукции p_3 и p_5). С другой стороны, эта же цепочка получается в результате применения продукции p_8 (если ранее применялись продукции p_3 и p_4). Следовательно, нет однозначности в выборе продукции при просмотре очередных трехсимвольных цепочек и поэтому приведенная грамматика не является строгой $LL(3)$-грамматикой. Однако первому (из рассмотренных выше) варианту появления цепочки bbc сопутствует терминальное начало wa, а второму — терминальное начало $w'd$, где w, w' — некоторые терминальные цепочки, так что данная грамматика является $LL(k)$-грамматикой. В то же время доказано [157], что для каждой $LL(k)$-грамматики существует эквивалентная строгая $LL(k)$-грамматика, так что класс $LL(k)$-языков совпадает с классом строгих $LL(k)$-языков.

Строгие $LL(k)$-грамматики допускают беступиковый анализ разверткой, при котором не учитывается влияние терминального начала некоторой строки вывода на его дальнейшее продолжение. Поэтому при строгих $LL(k)$-грамматиках может быть эффективно использован изложенный выше метод анализа делением на слово. Действительно, получив на некотором шаге анализа пару $(x_i, \psi v_i)$, по первым k символам цепочки x_i осуществим однозначный переход к паре (x_i, zv_i), где $(\psi \to z) \in \tilde{P}$ — продукция, применимая на данном шаге, $zv_i = u_{i+1}\tau v_{i+1}$. Выполнив затем сокращение цепочек x_i и zv_i на терминальное начало u_{i+1}, получим пару $(x_{i+1}, \tau v_{i+1})$ и перейдем к очередному шагу анализа. Если такой переход невозможен, то уже на данном шаге можно заключить, что анализируемая

цепочка не принадлежит языку, порожденному заданной строгой $LL(k)$-грамматикой.

Несколько иное определение грамматик, порождающих тот же класс языков, что и $LL(k)$-грамматики, предложено Д. Вудом [160].

Каждой продукции $(\psi \to z) \in \tilde{P}$ грамматики $\tilde{G}$ поставим в соответствие множество $M_{\psi, z}(k) = \{w\}$ всех терминальных цепочек w длины k таких, что в грамматике $\tilde{G}$ осуществляется левый вывод $(\tilde{\sigma}, \ldots, u\psi v, uzv, \ldots, uw, \ldots)$. Грамматика $\tilde{G}$ называется **левофакторной или $LF(k)$-грамматикой**, если для любых двух различных продукций $\psi \to z$ и $\psi \to z'$, принадлежащих схеме $\tilde{P}$, выполняется соотношение $M_{\psi, z}(k) \cap M_{\psi, z'}(k) = \varnothing$. Любая $LF(k)$-грамматика является строгой $LL(k)$-грамматикой и наоборот, так что классы языков, порожденных $LL(k)$- и $LF(k)$-грамматиками, совпадают.

Метод анализа делением на слово эффективен и при использовании класса $LL(f)$-грамматик, предложенных К. Чуликом [142]. Эти грамматики представляют собой обобщение (вообще говоря, не конструктивное) понятий $LL(k)$- и строгой $LL(k)$-грамматик.

Пусть $f: S \to D$ — функция, определенная на некотором подмножестве $S \subseteq F(\mathfrak{A})$ свободной подгруппы над алфавитом $\mathfrak{A}$ с произвольно выбранной областью значений D. Грамматика $\tilde{G}$ называется **$LL(f)$-грамматикой**, если для любых левых выводов $(\tilde{\sigma}, \ldots, u_1\psi v_1, u_1 z v_1, \ldots, u_1 w)$ и $(\tilde{\sigma}, \ldots, u_1\psi v_2, u_1 z' v_2, \ldots, u_1 w')$, где u_1, w, w' — терминальные цепочки, из $f(w) = f(w')$ следует $z = z'$. Функция f называется **различающей функцией** грамматики $\tilde{G}$.

Применяя в качестве различающей функцию $f(w) = k:w$ с областью определения $S = \{w\}$ всех $w \in F(\mathfrak{A})$ таких, что $|w| \geqslant k$, причем $k:w = u$, где u — терминальное начало цепочки w длины k, получаем $LL(k)$-грамматику. Аналогично можно определить **строгую $LL(f)$-грамматику** (обобщение строгой $LL(k)$-грамматики), потребовав, чтобы для любых левых выводов $(\tilde{\sigma}, \ldots, u_1\psi v_1, u_1 z v_1, \ldots, u_1 w)$ и $(\tilde{\sigma}, \ldots, u_2\psi v_2, u_2 z' v_2, \ldots, u_2 w')$, где u_1, u_2, w, w' — терминальные цепочки, из $f(w) = f(w')$ следовало $z = z'$. Функция f называется **строго различающей функцией** грамматики $\tilde{G}$. Для эффективного применения метода анализа делением на слово к кс-грамматике $\tilde{G}$ достаточно, чтобы для нее существовала строго различающая функция.

Наряду с рассмотренными методами левостороннего синтаксического анализа, при конструировании средств математического обеспечения для многопроцессорных вычислительных комплексов применяются методы параллельного синтаксического анализа, с помощью которых осуществляется одновременная обработка фрагментов входной программы [93, 123, 135, 153]. В работе [123]

выдвинута принципиально новая концепция двустороннего синтаксического анализа, состоящего в параллельной встречной обработке символов цепочки с противоположных ее концов двойственными методами развертки и свертки. Различаются два варианта стратегии двустороннего анализа: ПР-анализ (левая развертка — правая свертка) и РП-анализ (левая свертка — правая развертка) [83, 89]. В результате ПР- и РП-анализов порождаются соответственно левый и правый выводы анализируемой цепочки. Далее рассматривается стратегия ПР-анализа. При этом имеется в виду, что РП-анализ может быть организован двойственным образом.

Пусть $G = (\mathfrak{A}, V_н, \sigma, P)$ — приведенная кс-грамматика (см. § 5.3). Для цепочек $y_1, y_2, z_1, z_2 \in F(V)$ введем отношение непосредственной ПР-выводимости $(y_1, z_1) \underset{G}{|\!=\!|} (y_2, z_2)$, если выполняются следующие условия: 1) $y_1 = u\psi v$, $y_2 = uwv$, где $u \in F(\mathfrak{A})$; $w, v \in F(V)$, причем схема P содержит продукцию $p: \psi \to w$; 2) $z_1 = u'w'v'$, $z_2 = u'\tau v'$, где $u' \in F(\mathfrak{A})$; $w', v' \in F(V)$, причем схема P содержит продукцию $p': \tau \to w'$. Транзитивное замыкание непосредственной ПР-выводимости приводит к отношению $\overset{*}{\underset{G}{|\!=\!|}}$ ПР-выводимости в грамматике G.

Из приведенных определений следует, что цепочка x правильна, $x \in L(G)$, тогда и только тогда, когда $(\sigma, x) \overset{*}{\underset{G}{|\!=\!|}} (z, z')$, где $z = z'$ либо $z = u\psi v$, $z' = uwv$, и схема P содержит продукцию $p: \psi \to w$. Иными словами, если $\sigma = z_1 \overset{*}{\Rightarrow} z_i \overset{*}{\Rightarrow} z_{i+1} \overset{*}{\Rightarrow} z_n = x$ — левый вывод цепочки $x \in L(G)$, то можно указать такой номер i, что $(z_1, z_2, \ldots, z_i)$ — часть вывода, полученная разверткой слева направо, в то время как строки $z_n, z_{n-1}, \ldots, z_{i+1}$ порождаются сверткой справа налево, причем число шагов развертки и свертки отличается не более чем на единицу.

Пример 6. Рассмотрим кс-грамматику, заданную схемой

$$\begin{cases} p_1: \sigma \to a\sigma b, \\ p_2: \sigma \to ab. \end{cases}$$

Очевидно, эта грамматика порождает язык $L = \{a^n b^n \mid n = 1, 2, \ldots\}$ (см. § 5.2, пример 4). Выберем правильную цепочку $a^4b^4 \in L$, левый вывод которой имеет вид

$$w = (\sigma, a\sigma b, a^2\sigma b^2, a^3\sigma b^3, a^4 b^4).$$

По выводу w для цепочки a^4b^4 может быть построен следующий ПР-вывод:

$$(\sigma, a^4b^4) \underset{G}{|\!=\!|} (a\sigma b, a^3\sigma b^3) \underset{G}{|\!=\!|} (a^2\sigma b^2, a^2\sigma b^2).$$

ПР-вывод цепочки a^5b^5 имеет вид

$$(\sigma, a^5b^5) \models (a\sigma b, a^4\sigma b^4) \models (a^2\sigma b^2, a^3\sigma b^3),$$

причем из строки $z = a^2\sigma b^2$ в результате применения продукции $p_1: \sigma \to a\sigma b$ может быть получена строка $z' = a^3\sigma b^3$.

Анализаторы, ориентированные на стратегию двустороннего анализа, могут быть смоделированы с помощью магазинных ПР-анализаторов, впервые предложенных в работе [123]. На входной ленте этого автомата установлены указатели, фиксирующие фазу анализа в текущий момент. Внутренняя память состоит из двух магазинов: магазина развертки M_p и магазина свертки M_c. Начальное заполнение входной ленты имеет вид $\lceil a_1, a_2, \ldots, a_n \rceil$, где $x = a_1a_2 \ldots a_n$ — анализируемая цепочка, при этом в начальном состоянии магазин M_p содержит аксиому σ, а магазин M_c пуст. Управляющий блок организован в соответствии с грамматикой входного языка и позволяет на каждом шаге ПР-анализа применять пару продукций (p, q) соответственно по развертке и свертке с одновременным сдвигом указателей на входной ленте, отделяющих проанализированные части входной цепочки, и изменением содержимого магазинов M_p и M_c. При коллизии указателей (переходе одного из них через другой) осуществляется возврат по одному из направлений ПР-анализа. Например, возврат по развертке выражается в сдвиге на один шаг влево левого указателя и свертке в магазине M_p по последней из примененных в данном направлении продукций. Правильный переход в заключительное состояние совершается при слиянии указателей на входной ленте и совпадении содержимого магазинов M_p и M_c.

Проиллюстрируем функционирование магазинного ПР-анализатора в процессе анализа цепочек языка $L = \{a^nb^n \mid n = 1, 2, \ldots\}$, рассмотренного в примере 5. Анализ цепочки a^4b^4, приведенный в табл. 17, завершен успешно благодаря слиянию указателей и совпадению содержимого M_p и M_c, что подтверждает правильность данной цепочки $a^4b^4 \in L$. В табл. 18 показан ПР-анализ цепочки a^5b^5. На третьем шаге произошла коллизия указателей. Однако возврат

Таблица 17

Номер шага	Заполнение входной ленты	M_p	M_c	Использование пары продукций
0	$\lceil aaaabbbb \rceil$	σ	—	—
1	$a\lceil aa \rceil abbbb$	σb	σbbb	(p_1, p_2)
2	$aa\lceil \rceil aabbbb$	σbb	σbb	(p_1, p_1)

левого указателя и свертка в магазине M_p по продукции p_1 привели к слиянию указателей и совпадению содержимого магазинов. Следовательно, и в этом случае анализируемая цепочка a^5b^5 правильна, $a^5b^5 \in L$.

Таблица 18

Номер шага	Заполнение входной ленты	M_p	M_c	Использование пары продукций
0	$\lceil aaaaabbbbb \rceil$	σ	—	—
1	$a\lceil aaa \rceil abbbbb$	σb	$\sigma bbbb$	(p_1, p_2)
2	$aa\lceil a \rceil aabbbbb$	σbb	σbbb	(p_1, p_1)
3	$aa\rceil a \lceil aabbbbb$	σbbb	σbb	(p_1, p_1)
4	$aa\rceil\lceil aaabbbbb$	σbb	σbb	Свертка в M_p по p_1

Рассмотренная в настоящем параграфе техника управления левосторонним анализом сверткой и разверткой может быть использована также при организации беступикового двустороннего анализа. С целью изучения возможности эффективной реализации стратегии ПР-анализа разработана [83] подробная блок-схема ПР-анализатора параметрического типа. По этой блок-схеме составлена программная модель ПР-анализатора на языке АНАЛИТИК для ЭВМ МИР-2.

§ 6.4. Параметрические грамматики инверсно-рекурсивного и рекурсивного типа

Системы программирования относятся к основным и наиболее развитым средствам автоматизации программирования. Как уже упоминалось в § 6.1, широко применяются языки программирования самой различной ориентации, для которых разработаны эффективные системы программирования транслирующего и интерпретирующего типа. При этом процесс перевода с одного языка на другой включает кроме синтаксической и вторую фазу — семантическую, которая по дереву синтаксического анализа входной программы выполняет индуктивное построение ее семантического эквивалента на выходном языке (выходная программа), что обусловливает необходимость формализации не только синтаксиса, но и семантики языков программирования. Определенный вклад в этом направлении составляет теория интерпретированных языков и интерпретаторов. Интерпретированный язык сочетает в себе как синтаксический, так и семантический аспекты, что позволяет

подойти к проблеме параметризации процесса трансляции в целом, а не только его синтаксической фазы. Предложены **параметрические модели**, параметрами которых служат грамматики и другие объекты, отражающие наиболее общие и существенные черты классов входных и выходных языков соответствующих систем программирования. В результате подстановки фиксированных значений вместо указанных параметров получается конкретная система программирования для некоторой пары входного — выходного языков из выделенных классов. Такие конкретные системы программирования называются **параметризованными**, а сам процесс построения их — **процессом параметризации**.

К параметрическим моделям относятся **параметрические грамматики**, являющиеся эффективным средством управления процессом написания и анализа программ. В соответствии с двойственностью подходов к решению проблемы синтаксического анализа (см. § 6.3) различают параметрические грамматики инверсно-рекурсивного (аналог свертки) и рекурсивного (аналог развертки) типа [96].

Параметрическая грамматика инверсно-рекурсивного типа (ПГИРТ) $\overline{G} = (\overline{R}, A, \varphi)$ представляет собой объект, зависящий от следующих параметров: системы уравнений

$$\overline{R} = \{H_i(V) = :: \psi_i \mid i = 1, 2, \ldots, n\}$$

с выделенным уравнением $H_1(V) = :: \psi_1$, соответствующим аксиоме (см. § 5.3), где $H_i(V)$ — конечные языки в объединенном алфавите $V = \mathfrak{A} \cup V_н$; множества A бинарных отношений $\Delta \subseteq F(V) \times F(V)$; отображения $\varphi: \bigcup_{i=1}^{n} H_i(V) \to A$, ставящего в соответствие каждой цепочке $z \in H_i(V)$ некоторое бинарное отношение $\Delta \in A$, $\varphi(\psi_i, z) = \Delta$. Цепочка y непосредственно выводима из цепочки x в грамматике $\overline{G}$ ($x \sqsubset y(\overline{G})$) (или $x \sqsubset y$, если грамматика $\overline{G}$ подразумевается, $x, y \in F(V)$) при условии, что $x = u_1 z v_1$, $y = u_1 \psi_i v_1$, причем $z \in H_i(V)$ и $(u_1, v_1) \in \Delta = \varphi(\psi_i, z)$. Иными словами, подцепочку z, входящую в левую часть уравнения $H_i(V) = :: \psi_i$, можно свернуть в нетерминал ψ_i, если в цепочке x подцепочка z имеет контексты (u_1, v_1), которые входят в соответствующие бинарные отношения $\Delta = \varphi(\psi_i, z)$. Цепочка y выводима из цепочки x в грамматике $\overline{G}$, если существует вывод $z_0, z_1, \ldots, z_n$ такой, что $x = z_0$, $y = z_n$ и при каждом $i = 1, 2, \ldots, n$ $z_{i-1} \sqsubset z_i(\overline{G})$. Выводимость обозначается $x \overset{*}{\sqsubset} y(\overline{G})$ (или $x \overset{*}{\sqsubset} y$, если грамматика $\overline{G}$ подразумевается). Под языком $L(\overline{G})$, порождаемым грамматикой $\overline{G}$, понимается множество терминальных цепочек $x \in F(\mathfrak{A})$ таких, что $x \overset{*}{\sqsubset} \psi_1(\overline{G})$.

Пример 1. Рассмотрим параметрическую грамматику $\overline{G}_a = (\overline{R}, A_a, \varphi_a)$, где множество A_a состоит из единственного бинарного отношения $\Delta = \{e\} \times F(\mathfrak{A})$, так что каждой цепочке $z \in H_i(V)$, входящей в левую часть некоторого уравнения $H_i(V) = :: \psi_i$, отображение φ_a ставит в соответствие отношение $\Delta = \varphi(\psi_i, z)$, $i = 1, 2, \ldots, n$, $V = \mathfrak{A} \cup V_н$. Очевидно, язык $L(\overline{G}_a)$, порожденный данной параметрической грамматикой, **автоматный** (см. § 5.4). Действительно, свертка по цепочке $z \in H_i(V)$ возможна, когда продукция $\psi_i \to z$ леволинейна, причем контексты удовлетворяют выбранному отношению Δ. Подставив вместо соответствующего параметра грамматики $\overline{G}_a$ систему уравнений $\overline{R}$, состоящую из леволинейных продукций и порождающую произвольный автоматный язык L, получим $L(\overline{G}_a) = L$. Таким образом, справедлива теорема.

Теорема 6.4. *Параметрические грамматики инверсно-рекурсивного типа порождают произвольные автоматные языки.*

Пример 2. Рассмотрим параметрическую грамматику $\overline{G}_л = (\overline{R}, A_л, \varphi_л)$, где множество $A_л$ состоит из единственного бинарного отношения $\Delta = F(\mathfrak{A}) \times F(\mathfrak{A})$, так что $\varphi_л(\psi_i, z) = \Delta$ для любой цепочки $z \in H_i(V)$ ($V = \mathfrak{A} \cup V_н$, $i = 1, 2, \ldots, n$). Как и в примере 1, свертка цепочки $z \in H_i(V)$ возможна, когда продукция $\psi_i \to z$ линейна и имеющиеся на данном шаге контексты удовлетворяют выбранному отношению Δ. Таким образом, подставив вместо параметра $\overline{R}$ в грамматику $\overline{G}_л$ систему уравнений, порождающую произвольный линейный язык (см. § 5.4), получим $L(\overline{G}_л) = L$. Справедлива следующая теорема.

Теорема 6.5. *Параметрические грамматики инверсно-рекурсивного типа порождают произвольные линейные языки.*

Пример 3. Рассмотрим параметрическую грамматику $\overline{G}_{кс} = (\overline{R}, A_{кс}, \varphi_{кс})$, где множество $A_{кс}$ состоит из единственного бинарного отношения $\Delta = F(V) \times F(V)$, так что $\varphi_{кс}(\psi_i, z) = \Delta$ для любой цепочки $z \in H_i(V)$ ($V = \mathfrak{A} \cup V_н$, $i = 1, 2, \ldots, n$). Язык $L(\overline{G}_{кс})$ совпадает с кс-языком, порожденным системой уравнений (см. § 5.3). Приведенное построение называется **тривиальной параметризацией** системы уравнений $\overline{R}$. Таким образом, справедлива теорема.

Теорема 6.6. *Параметрические грамматики инверсно-рекурсивного типа порождают произвольные кс-языки.*

Пример 4. Рассмотрим нс-язык $L_6 = \{xbx \mid$ для любого непустого слова $x \in F(\mathfrak{A})\}$, где $\mathfrak{A} = \{a_1, a_2, \ldots, a_m\}$, с порождающей грамматикой G_6 (см. § 5.2, пример 6). Построим параметрическую грамматику $\overline{G}_6(\overline{R}_6, A_6, \varphi_6)$, порождающую нс-язык L_6. Параметры грамматики $\overline{G}_6$ определяются следующим образом:

$$\bar{R}_6 \left\{ \begin{array}{l} H_1(V) = \sigma\psi_i\alpha_i \vee \beta = :: \sigma \xrightarrow{\varphi_6} F(V_1) \times F(V_1) \\ H_2(V) = \rho_{ij} = :: \alpha_i \xrightarrow{\varphi_6} F(V_1) \times \psi_j F(V_1) \\ H_3(V) = \alpha_i = :: \psi_j \xrightarrow{\varphi_6} F(V_1)\rho_j \times F(V_1) \\ H_4(V) = \psi_j = :: \rho_{ij} \xrightarrow{\varphi_6} F(V_1) \times \alpha_i F(V_1) \\ H_5(V) = \tau_i = :: \beta \xrightarrow{\varphi_6} F(V_1) \times \psi_i F(V_1) \\ H_6(V) = \beta = :: \psi_i \xrightarrow{\varphi_6} F(V_1)\tau_i \times F(V_1) \\ H_7(V) = \alpha_i = :: \tau_i \xrightarrow{\varphi_6} F(V_1) \times \beta F(V_1) \\ H_8(V) = b = :: \beta \xrightarrow{\varphi_6} F(V_1) \times F(V_1) \\ H_9(V) = a_i = :: \alpha_i \xrightarrow{\varphi_6} F(V_1) \times F(V_1) \end{array} \right\} A_6.$$

Рассмотрим вывод цепочки $(a_1a_2ba_1a_2) \in L_6$ в грамматике $\bar{G}_6$. Сворачивая цепочки, входящие в левые части $H_8(V)$ и $H_9(V)$ приведенной системы уравнений, получаем цепочку $\alpha_1\alpha_2\beta\alpha_1\alpha_2$. Далее, последовательно сворачивая цепочки, входящие в H_7, H_6 и H_5, получаем $\alpha_1\beta\psi_2\alpha_1\alpha_2$. Вновь применяя в указанном порядке свертки, получаем $\beta\psi_1\psi_2\alpha_1\alpha_2$. Применяя последовательную свертку цепочек, входящих в H_4, H_3 и H_2, получаем $\beta\psi_1\alpha_1\psi_2\alpha_2$. Наконец, применяя свертки цепочек, входящих в H_1, получаем завершенную часть вывода

$$\beta\psi_1\alpha_1\psi_2\alpha_2 \vdash \sigma\psi_1\alpha_1\psi_2\alpha_2 \vdash \sigma\psi_2\alpha_2 \vdash \sigma.$$

Пусть $G = (\mathfrak{A}, V_н, \sigma, P)$ — произвольная нс-грамматика (см. § 5.2). Из каждой нс-продукции $(u_1\psi u_2 \to u_1 z u_2) \in P$ выделим кс-продукцию $\psi \to z$ и контексты (u_1, u_2), в которых она применима. Пусть

$$\{(u_{1j}, u_{2j}) \mid j = 1, 2, \ldots, r(\psi, z)\} \tag{6.11}$$

является совокупностью всех контекстов, в которых применима продукция $\psi \to z$. По выделенным кс-продукциям составим систему уравнений

$$\bar{R}_G = \{H_i(V) = :: \psi_i \mid i = 1, 2, \ldots, n\}.$$

Кроме того, по контекстам (6.11), соответствующим продукции $\psi \to z$, составим бинарное отношение

$$\Delta_{(\psi, z)} = \bigcup_{j=1}^{r(\psi, z)} F(V) u_{1j} \times u_{2j} F(V).$$

Объединив отношения $\Delta_{(\psi, z)}$ для всех z, входящих в левые части построенной системы уравнений $\bar{R}_G$, получим множество A_G. Наконец, определив отображение φ_G так, что $\varphi_{G(\psi_i, z)} = \Delta_{(\psi_i, z)}$, получим параметрическую грамматику $\bar{G} = (\bar{R}_G, A_G, \varphi_G)$ такую, что $L(\bar{G}) = L(G)$. Таким образом, справедлива теорема.

Теорема 6.7. *Параметрические грамматики инверсно-рекурсивного типа порождают нс-языки.*

Как отмечалось выше (см. § 6.3), с точки зрения практического построения систем программирования представляют интерес формализмы для описания языков, допускающие беспереборный анализ, при котором наличие хотя бы одного тупика означает, что анализируемая цепочка не принадлежит заданному языку. К таким формализмам относятся беспереборные грамматики $\bar{G} = (\bar{R}, A, \varphi)$. Доказано, что любой кс-язык может быть задан некоторой **беспереборной** грамматикой $\bar{G}$. Следовательно, эффективность синтаксического анализа цепочек данного языка зависит от сложности бинарных отношений, входящих в параметр A грамматики $\bar{G}$. Поэтому естественно классифицировать методы синтаксического анализа по сложности бинарных отношений из A. В работе [79] приводится формализация стратегии левосторонней свертки в терминах параметрических грамматик инверсно-рекурсивного типа.

Рассмотрим грамматику $\bar{G} = (\bar{R}, A, \varphi)$, где $\bar{R} = \{H_i(V) = :: \psi_i \mid i = 1, 2, \ldots, n\}$. Параметры A и φ определяются следующим образом. Каждой цепочке $z \in H_i(V)$ сопоставляется конечное множество $M_z \subset F(\mathfrak{A})$ цепочек в терминальном алфавите $\mathfrak{A}$. Отображение $\varphi : \bigcup_{i=1}^{n} H_i(V) \to A$ ставит в соответствие каждой цепочке $z \in H_i(V)$ бинарное отношение $\Delta = (L^1, L^2)$, где L^1 — множество допустимых цепочек $v \in F(V)$, которые используются для однозначного выделения в vzw самой левой подцепочки z, подлежащей свертке; $L^2 = M_z \cdot F(\mathfrak{A})$. Параметр A — множество всех бинарных отношений $\Delta = \varphi(\psi_i, z)$ для каждой цепочки $z \in H_i(V)$ $(i = 1, 2, \ldots, n)$. Грамматика $\bar{G}$ с определенными параметрами A и φ называется **параметрической грамматикой инверсно-рекурсивного типа с выводом слева направо или *LR*-грамматикой.** Отношение $(F(V)M_1 \times M_2F(V)) \subseteq F(V) \times F(V)$, где M_1, M_2 — конечные множества цепочек в алфавите $V = \mathfrak{A} \cup V_н$, называется **квазиуниверсальным.** В работе [79] доказано, что для ограниченно-контекстных грамматик Флойда и $LR(k)$-грамматик Кнута (см. § 6.3) существуют эквивалентные беспереборные грамматики $\bar{G}$ соответственно с квазиуниверсальными отношениями и выводом слева направо. Это позволяет автоматически получать эффективные алгоритмы синтаксического анализа для широкого класса кс-языков со сложностью, пропорциональной длине анализируемой цепочки.

Пусть $R = \{\psi_i :: = H_i(V) \mid i = 1, 2, \ldots, n\}$ — система уравнений, задающая некоторый кс-язык. Строка $\psi_k :: = H_k(V)$ **подобна** строке $\psi_r :: = H_r(V)$, если существует взаимно-однозначное соответствие $\psi : H_k(V) \leftrightarrow H_r(V)$, которое каждой цепочке $z = u_1\psi_{i_1}^k u_2\psi_{i_2}^k \ldots u_s\psi_{i_s}^k u_{s+1} \in H_k(V)$ сопоставляет цепочку $z' = u_1\psi_{i_1}^r \cdot u_2\psi_{i_2}^r \ldots u_s\psi_{i_s}^r u_{s+1} \in H_r(V)$, где $u_j \in F(\mathfrak{A})$ при любом $j = 1, 2, \ldots, s+1$, причем $\psi_{i_l}^k = \psi_{i_l}^r$ либо $\psi_{i_l}^k = \psi_k$ и $\psi_{i_l}^r = \psi_r$ при каждом $t = 1, 2, \ldots, s$. Нетерминал ψ_k **равносилен** нетерминалу ψ_r, если строки $\psi_k :: = H_k(V)$ и $\psi_r :: = H_r(V)$ подобны. Введенные отношения равносильны, и подобия являются отношениями эквивалентности на множествах нетерминалов и строк (соответственно) и определяют разбиение соответствующих множеств на классы эквивалентности.

Система уравнений R^* называется **приведенной формой** системы R, если R и R^* порождают один и тот же язык и выполняются следующие условия:

1) язык L_1^∞, порожденный по i-й компоненте системы R^* (см. § 5.3), бесконечен при каждом $i = 1, 2, \ldots, n$;

2) каждый класс подобия (равносильности) состоит из единственной строки (одного нетерминала).

Доказано, что любая система уравнений R, задающая некоторый кс-язык, может быть преобразована к эквивалентной приведенной форме [79].

Приведенная форма позволяет расширить класс кс-грамматик, для которых существуют эквивалентные бесперебросные грамматики $\bar{G}$ рассмотренных выше типов. Действительно, есть кс-грамматики, которые не являются ограниченно-контекстными и $LR(k)$-грамматиками, но приведенная форма которых уже принадлежит к перечисленным классам, откуда следует существование эквивалентных бесперебросных грамматик $\bar{G}$ соответственно с квазиуниверсальными отношениями и выводом слева направо.

Пример 5. Рассмотрим систему уравнений

$$R_1 = \begin{cases} \psi_0 :: = \# \psi_1 \#, \\ \psi_1 :: = b_1\psi_2 a_1 \vee b_2\psi_2 a_1 \vee \psi_4\psi_3 a_2 \vee \psi_4\psi_5 a_3, \\ \psi_2 :: = \psi_2 c \vee c, \\ \psi_3 :: = \psi_3 c \vee c, \\ \psi_4 :: = b_1 \vee b_2, \\ \psi_5 :: = \psi_5 c \vee c. \end{cases}$$

кс-Грамматика, соответствующая этой системе уравнений, не является $LR(k)$-грамматикой при любом k, так как при фиксированном k можно указать правильные цепочки, в которых однозначное определение свертки на некоторых шагах требует просмотра правого терминального контекста, превосходящего по длине выбранное k. Такая, например, цепочка $\# b_1 c^{k+1} a_1 \#$. Первое вхождение символа c является самой левой подцепочкой, подлежащей свертке. Однако определение нетерминала, который необходимо подставить, связано с просмотром последующих терминалов. В то же время приведенная форма данной системы имеет вид

$$R_1^* = \psi_0 :: = \# \psi_1 \#,$$

$$\psi_1 :: = b_1\psi_2 a_1 \vee b_2\psi_2 a_1 \vee b_1\psi_2 a_2 \vee b_2\psi_2 a_2 \vee b_1\psi_2 a_3 \vee b_2\psi_2 a_3,$$

$$\psi_2 :: = \psi_2 c \vee c$$

и, очевидно, является $LR(k)$-грамматикой. Ей эквивалентна бесперебросная грамматика $\bar{G}_1 = (\bar{R}_1^*, A_1, \varphi_1)$, где

$$\# \psi_1 \# = :: \psi_0,$$

$$H_1(V_1) = b_1\psi_2 a_1 \vee b_2\psi_2 a_1 \vee b_1\psi_2 a_2 \vee b_2\psi_2 a_2 \vee b_1\psi_2 a_3 \vee b_2\psi_2 a_3 = :: \psi_1,$$

$$H_2(V_1) = \psi_2 c \vee c = :: \psi_2,$$

$$A_1 = \begin{cases} \Delta_0 = (e, c), \\ \Delta_1 = (\#, \#), \\ \Delta_2 = (\bigcup\limits_{i=1,2} \# b_i, F(\mathfrak{A}_1)), \end{cases}$$

$$\varphi_1(\# \psi_1 \#) = \Delta_0,$$

$$\varphi(z) = \begin{cases} \Delta_1, & \text{если } z \in H_1(V_1), \\ \Delta_2, & \text{если } z \in H_2(V_1), \end{cases}$$

$$(V_1 = \mathfrak{A}_1 \cup V_{\text{н}}^1,\ \mathfrak{A}_1 = \{\#, a_1, a_2, a_3, b_1, b_2\},$$

$$V_{\text{н}}^1 = \{\psi_0, \psi_1, \psi_2\}).$$

Перейдем к параметрическим грамматикам рекурсивного типа (аналог развертки). Под **взвешенной цепочкой** понимается пара (z, L), где $z \in F(V)$, $L \subseteq F(\mathfrak{A})$; вторая компонента — язык L — называется **весом** цепочки z — первой компоненты пары (z, L). Взвешенная цепочка (z, L) называется **ассоциированной** с терминальной цепочкой $z \in F(\mathfrak{A})$, если $z \in L$. Примерами взвешенных цепочек служат пары $(a\psi, L)$ и (a^3b^2, L), где $L = \{a^n b^m \mid n, m = 1, 2, \ldots\}$, причем вторая пара представляет собой взвешенную цепочку, ассоциированную с цепочкой a^3b^2, в то же время взвешенная цепочка $(a^n L)$ не ассоциирована с терминальной цепочкой a^n, так как $a^n \overline{\in} L$.

Параметрической грамматикой рекурсивного типа (ПГРТ) называется грамматика $G = (R, A, \varphi)$ со следующими параметрами: $R = \{\psi_i :: = H_i(V) \mid i = 1, 2, \ldots, n\}$ — система уравнений с выделенной аксиомой ψ_1; A — множество бинарных отношений Δ, каждое

из которых состоит из совокупности взвешенных цепочек; $\varphi: \bigcup_{i=1}^{n} H_i(V) \to A$ — отображение, сопоставляющее каждой цепочке $z \in H_i(V)$ некоторое бинарное отношение $\Delta = \varphi(\psi_i, z)$ из множества A.

Пример 6. Грамматикой рекурсивного типа является грамматика $G_1 = (R_1, A_1, \varphi_1)$, параметры которой определяются следующим образом:

$$R_1 = \begin{cases} \sigma ::= a\psi, \\ \psi ::= a\psi \vee b\psi_1 \vee c\psi_2, \\ \psi_1 ::= b\psi_1 \vee b = H_1(V_1), \\ \psi_2 ::= c\psi_2 \vee c = H_2(V_1). \end{cases}$$

$$A_1 = \begin{cases} \Delta_\sigma = \{(\sigma, aF(\mathfrak{A}_1))\}, \\ \Delta_\psi = \{(a^n\psi, aF(a)F(\mathfrak{A}_1)) \mid n = 1, 2, \ldots\}, \\ \Delta_\psi^1 = \{(a^n\psi, aF(a)bF(\mathfrak{A}_1)) \mid n = 1, 2, \ldots\}, \\ \Delta_\psi^2 = \{(a^n\psi, cF(\mathfrak{A}_1)) \mid n = 1, 2, \ldots\}, \\ \Delta_{\psi_1} = \{(a^n b^m \psi_1, aF(a)bF(b)) \mid n, m, = 1, 2, \ldots\}, \\ \Delta_{\psi_2} = \{(a^n c^m \psi_2, cF(\mathfrak{A}_1)) \mid m, n = 1, 2, \ldots\}, \end{cases}$$

где $F(\mathfrak{A}_1)$ — свободная полугруппа над алфавитом $\mathfrak{A}_1 = \{a, b, c\}$; $F(a)$, $F(b)$ — свободные полугруппы над однобуквенными алфавитами $\{a\}$, $\{b\}$ соответственно. Отображение φ_1 определяется равенствами

$$\varphi_1(\sigma, a\psi) = \Delta_\sigma, \quad \varphi_1(\psi, a\psi) = \Delta_\psi, \quad \varphi_1(\psi, b, \psi_1) = \Delta_\psi^1, \quad \varphi_1(\psi, c\varphi_2) = \Delta_\psi^2,$$
$$\varphi_1(\varphi_1, z) = \Delta_{\psi_1} \text{ для каждой цепочки } z \in H_1(V_1),$$
$$\varphi_1(\varphi_2, z) = \Delta_{\psi_2} \text{ для каждой цепочки } z \in H_2(V_1),$$

где $V_1 = \{\mathfrak{A}_1 \cup \{\sigma, \psi, \psi_1, \psi_2\}\}$ — объединенный алфавит.

Взвешенная цепочка (y, L_2) непосредственно выводима из взвешенной цепочки (x, L_1) в грамматике $G = (R, A, \varphi)$, $(x, L_1) \sqsubseteq (y, L_2)(G)$ или $(x, L_1) \sqsubseteq (y, L_2)$, если грамматика G подразумевается, при условии, что $x = u_1\psi_i v_1$, $y = u_1 z v_1$, причем $u_1 \in F(\mathfrak{A})$, $z \in H_i(V)$ и $(x, \overline{L}) \in \Delta = \varphi(\psi_i, z)$, следовательно, язык $L_2 = L_1 \cap \overline{L}$ непуст. Так, для грамматики G_1 (см. пример 6) $(a^k\psi, F(\mathfrak{A}_1)) \sqsubseteq (a^k c\psi_2, cF(\mathfrak{A}_1))$. Действительно, к цепочке $a^k\psi$ с весом $L_1 = F(\mathfrak{A}_1)$ применяется продукция $\psi \to c\psi_2$, причем $\varphi_1(\psi, c\psi_2) = \Delta_\psi^2 = \{(a^n\psi, cF(\mathfrak{A}_1)) \mid n = 1, 2, \ldots\}$, так что в результате получается цепочка $a^k c\psi_2$ с весом $L_2 = F(\mathfrak{A}_1) \cap cF(\mathfrak{A}_1) = cF(\mathfrak{A}_1)$. В то же время взвешенная цепочка $(ac\varphi_2, L_2)$ не выводится непосредственно из взвешенной цепочки $(a\psi, aF(\mathfrak{A}_1))$, так как $cF(\mathfrak{A}_1) \cap aF(\mathfrak{A}_1) = \varnothing$.

Взвешенная цепочка (y, L_2) выводима из взвешенной цепочки (x, L_1) в грамматике $G = (R, A, \varphi)$, если существует вывод

$$(z_0, \overline{L}_0) \sqsubseteq (z_1, \overline{L}_1) \sqsubseteq \cdots \sqsubseteq (z_n, \overline{L}_n) \tag{6.12}$$

такой, что $x = z_0$, $L_1 = \overline{L}_0$ и $y = z_n$, $L_2 = \overline{L}_n$. Первые компоненты взвешенных цепочек, входящих в (6.12), представляют собой левосторонний вывод цепочки $y = z_n$ из цепочки $x = z_0$ в кс-грамматике, соответствующей системе уравнений R, а вторые — монотонно убывающую последовательность языков

$$L_1 = \overline{L}_0 \supset \overline{L}_1 \supset \cdots \supset \overline{L}_n = L_2. \tag{6.13}$$

Выводимость (y, L_2) из (x, L_1) обозначается $(x, L_1) \overset{*}{\sqsubseteq} (y, L_2)(G)$ или $(x, L_1) \overset{*}{\sqsubseteq} (y, L_2)$, если грамматика G подразумевается.

Под языком $L(G)$, порождаемым грамматикой $G = (R, A, \varphi)$, понимается множество таких и только таких терминальных цепочек $x \in F(\mathfrak{A})$, для каждой из которых существует ассоциированная цепочка (x, L), выводимая из взвешенной цепочки $(\psi_1, F(\mathfrak{A}))$. В общем случае такой язык $L(G)$ является собственным подмножеством языка $L(R)$, порожденного системой уравнений R, являющейся параметром данной грамматики $G = (R, A, \varphi)$, так как на замены ψ_i цепочками из $H_i(V)$ накладываются дополнительные ограничения, связанные с использованием параметров A и φ. В частности, грамматика G_1 (см. пример 6) порождает язык $L(G_1) = \{a^n b^m \mid n, m = 1, 2, \ldots\}$, который является собственным подмножеством языка $L(R_1) = \{a^n b^{m_1} \cup a^n c^{m_2} \mid n, m_1, m_2 = 1, 2, \ldots\}$, порожденного системой уравнений R_1. Действительно, в грамматике G выводимы взвешенные цепочки вида $(a^n b^m, aF(a)bF(b))$. Так, вывод

$$(\sigma, F(\mathfrak{A}_1)) \sqsubseteq (a\psi, aF(\mathfrak{A}_1)) \sqsubseteq (aa\psi, aF(a)F(\mathfrak{A}_1) \sqsubseteq (aab\varphi_1, aF(a)bF(\mathfrak{A}_1)) \sqsubseteq (aabb, aF(a)bF(b))$$

является выводом взвешенной цепочки $(a^2b^2, aF(a)bF(b))$ в грамматике G_1, откуда следует, что $a^2b^2 \in L(G_1)$. С другой стороны, $a^n c^m \overline{\in} L(G_1)$, поскольку при построении выводов соответствующих взвешенных цепочек замещение нетерминала ψ цепочкой $c\psi_2$ невозможно из-за пустоты пересечения весов, $aF(a)F(\mathfrak{A}_1) \cap cF(\mathfrak{A}_1) = \varnothing$. Таким образом, из определений непосредственной выводимости и выводимости в грамматике $G = (R, A, \varphi)$ следует, что вывод взвешенной цепочки, ассоциированной с цепочкой $x \in L(G)$, управляется параметрами R, A, φ так, что на каждом шаге в качестве второй компоненты строится некоторый вес (язык, содержащий цепочку x), тогда как первой компонентой является цепочка, терминальное начало которой совпадает с началом цепочки x.

Заметим, что понятие веса является обобщением понятия множества $M_{\psi, z}(k)$, которое используется при определении $LF(k)$-грамматик (см. § 6.3), так как оба эти понятия содержат дополнительную информацию о возможности замещения нетерминала ψ цепочкой z на каждом шаге вывода.

Пусть $R = \{\psi_i :: = H_i(V) \mid i = 1, 2, \ldots, n\}$ — некоторая система уравнений, порождающая язык $L(R)$. В частности, система R может быть представлена соответствующей БНФ. Рассмотрим грамматику $G = (R, A, \varphi)$, параметр A которой состоит из единственного бинарного отношения $\Delta = \{(x, F(\mathfrak{A}) \mid$ для всякого $x \in F(V)\}$. Язык $L(G)$, порожденный этой грамматикой, очевидно, совпадает с языком $L(R)$, т. е. $L(G) = L(R)$. Переход от задания языка посредством системы R к его заданию с помощью параметрической грамматики G является тривиальной параметризацией системы R (или соответствующей БНФ). Таким образом, справедливо следующее утверждение.

Теорема 6.8. *Параметрическая грамматика $G = (R, A, \varphi)$, полученная с помощью тривиальной параметризации системы уравнений R, задает тот же язык, что и система уравнений R.*

Из этой теоремы следует, что параметрические грамматики рекурсивного типа являются естественным обобщением кс-грамматик, а значит, и БНФ.

Грамматика $G = (R, A, \varphi)$ называется **беспереборной,** если для различных цепочек $z, z' \in H_i(V)$, $\varphi(\psi_i, z) \bigcap \varphi(\psi_i, z') = \varnothing$ при каждом $i = 1, 2, \ldots, n$, т. е. бинарные отношения, соответствующие цепочкам z и z', при отображении φ не пересекаются. Это означает, что на каждом шаге вывода в беспереборной грамматике G замена нетерминала ψ_i цепочкой $z \in H_i(V)$ однозначно определяется информацией о возможном продолжении вывода, содержащейся в весе соответствующего бинарного отношения. Доказано, что любой кс-язык порождается некоторой беспереборной грамматикой G [96]. В то же время обратное утверждение неверно, так как не всякий язык, порожденный беспереборной грамматикой G, контекстно-свободный. Таким образом, эффективность синтаксического анализа языков, заданных грамматиками из рассмотренного класса, как и при задании грамматиками инверсно-рекурсивного типа, зависит от сложности бинарных отношений, входящих в множество A. Поэтому при классификации методов синтаксического анализа разверткой с помощью грамматик $G = (R, A, \varphi)$ следует учитывать сложность параметра A. В работе [126] формулируются достаточные условия, позволяющие выделить классы грамматик $G = (R, A, \varphi)$, ориентированных на беспереборный синтаксический анализ разверткой. Эти классы являются обобщениями классов $LL(k)$-строгих $LF(k)$- и $LL(f)$-грамматик, при этом осуществляется параметризация грамматик из указанных классов, т. е. переход к эквивалентной грамматике G.

Строго беспереборной называется грамматика $G = (R, A, \varphi)$, параметры которой удовлетворяют следующим условиям: 1) если $(v, L_1) \in \varphi(\psi_i, z)$ и $(v, L_2) \in \varphi(\psi_i, z')$ для различных цепочек $z, z' \in H_i(V)$, то $L_1 \bigcap L_2 = \varnothing$; 2) для любой цепочки $z \in H_i(V)$ из $(v, L_1) \in \varphi(\psi_i, z)$ и $(v, L_2) \in \varphi(\psi_i, z')$ следует, что $L_1 = L_2$. Первое условие требует, чтобы на каждом шаге вывода замена ψ_i цепочкой z однозначно определялась информацией о возможности продолжения этого вывода, второе означает, что всякое преобразование первой компоненты взвешенной цепочки влечет за собой однозначное преобразование второй компоненты — веса этой цепочки. Так, грамматика G_1 (см. пример 6) не является беспереборной, а значит, и строго беспереборной, поскольку пересечение $\varphi_1(\psi_1, b\psi_1) \bigcap \varphi_1(\psi_1, b) = \Delta\psi_1$ непусто.

Пример 7. Строго беспереборной является грамматика $G_2 = (R_2, A_2, \varphi_2)$, которая порождает $LL(k)$-язык, $L = \{a^n b^m \# \mid n, m = 1, 2, \ldots\}$ (см. § 6.3, пример 4) и параметры которой определяются следующим образом:

$$R_2 = \begin{cases} \psi_0 :: = \psi_1 \#, \\ \psi_1 :: = a\psi_2, \\ \psi_2 :: = a\psi_2 \vee b\psi_3, \\ \psi_3 :: = b\psi_3 \vee b, \end{cases}$$

$$A_2 = \begin{cases} \Delta_{\psi_0} = \{(\psi_0, F(\mathfrak{A}_2))\}. \\ \Delta_{\psi_1} = \{(\psi_1 \#, aF(\mathfrak{A}_2))\}, \\ \Delta^1_{\psi_2} = \{(a^n\psi_2 \#, a^n aF(\mathfrak{A}_2)) \mid \text{для любого } n \geqslant 1\}, \\ \Delta^2_{\psi_2} = \{(a^n\psi_2 \#, a^n bF(\mathfrak{A}_2)) \mid \text{для любого } n \geqslant 1\}, \\ \Delta^1_{\psi_3} = \{(a^n b^m \psi_3 \#, a^n b^m bF(\mathfrak{A}_2) \mid \text{для любого } n, m \geqslant 1\}, \\ \Delta^2_{\psi_3} = \{(a^n b^m \psi_3 \#, a^n b^m \# \mid \text{для всех } n, m \geqslant 1\}, \end{cases}$$

где $\mathfrak{A}_2 = \{a, b, \#\}$ — терминальный алфавит. Отображение φ_2 определяется равенствами

$$\varphi_2(\psi_0, \psi_1, \#) = \Delta_{\psi_0},\ \varphi_2(\psi_1, a\psi_2) = \Delta_{\psi_1},\ \varphi_2(\psi_2, a\psi_2) = \Delta^1_{\psi_2},$$
$$\varphi_2(\psi_2, b\psi_3) = \Delta^2_{\psi_2},\ \varphi_2(\psi_3, b\psi_3) = \Delta^1_{\psi_3},\ \varphi_2(\psi_3, b) = \Delta^2_{\psi_3}.$$

Действительно, для взвешенных цепочек $(a^n\psi_2 \#, L_1) \in \Delta^1_{\psi_2} = \varphi_2(\psi_2, a\psi_2)$ и $(a^n\psi_2 \#, L_1') \in \Delta^2_{\psi_2} = \varphi_2(\psi_2, b\psi_3)$, $L_1 \bigcap L_1' = \varnothing$, где $L_1 = a^n aF(\mathfrak{A}_2)$, $L_1' = a^n bF(\mathfrak{A}_2)$. Аналогично для взвешенных цепочек $(a^n b^m \psi_3 \#, L_2) \in \Delta^1_{\psi_3} = \varphi_2(\psi_3, b\psi_3)$ и $(a^n b^m \#, L_2') \in \Delta^2_{\psi_3} = \varphi_2(\psi_3, b)$, где $L_2 \bigcap L_2' = \varnothing$, $L_2 = a^n b^m bF(\mathfrak{A}_2)$, $L_2' = \{a^n b^m \#\}$. Кроме того, каждое бинарное отношение, входящее в A_2, функционально по второй компоненте (т. е. не содержит двух пар с одинаковой первой и разными вторыми компонентами). Таким образом, грамматика G_2 удовлетворяет условиям строгой беспереборности.

Вывод $\psi_1, F(\mathfrak{A}) \sqsubseteq (z_1, L_1) \sqsubseteq (z_2, L_2) \sqsubseteq \cdots \sqsubseteq (z_k, L_k)$ $(k \geqslant 1)$ взвешенной цепочки (z_k, L_k) в грамматике G называется управляемым некоторой терминальной цепочкой $x \in F(\mathfrak{A})$, если каждая цепочка $z_i = u_i \psi_j v_i$ имеет терминальное начало u_i, совпадающее с началом цепочки x, причем $x \in L_i$ при любом $i = 1, 2, \ldots, k$. Примером управляющего вывода в грамматике G служит вывод взвешенной цепочки (x, L), ассоциированной с цепочкой $x \in L(G)$, управляемый цепочкой x; в грамматике G_2 (см. пример 7) — вывод $(\psi_0, F(\mathfrak{A}_2)) \sqsubseteq (\psi_1 \#, F(\mathfrak{A}_2)) \sqsubseteq (a\psi_2\#, aF(\mathfrak{A}_2)) \sqsubseteq (aa\psi_2\#, aaF(\mathfrak{A}_2)) \sqsubseteq \sqsubseteq (aab\psi_3\#, aabF(\mathfrak{A}_2))$, управляемый цепочкой $x_1 = aaba\# \in L(G_2)$. Действительно, x' принадлежит весу каждой взвешенной цепочки данного вывода и терминальные начала первых компонент этих цепочек совпадают с началом цепочки x_1.

Теорема 6.9. *Для каждой цепочки $x \in F(\mathfrak{A})$ в строго беспереборной грамматике G существует единственный максимальный по длине вывод, управляемый данной цепочкой.*

Действительно, пусть $(\psi_1, F(\mathfrak{A})) \sqsubseteq (z_1, L_1) \sqsubseteq \cdots \sqsubseteq (z_k, L_k)$ — некоторый максимальный по длине вывод в грамматике G, управляемый цепочкой $x \in F(\mathfrak{A})$. В силу строгой беспереборности на каждом шаге данного вывода преобразование первой компоненты взвешенной цепочки влечет за собой однозначное преобразование ее второй компоненты. С другой стороны, если на произвольно выбранном r-м шаге данного вывода $(r \geqslant 1)$, на котором осуществляется переход от (z_{r-1}, L_{r-1}) к (z_r, L_r), где $z_{r-1} = u_{r-1}\psi_i v_{r-1}$, $z_r = u_{r-1} z v_{r-1}$, $z \in H_i(V)$, заменить ψ_i цепочкой $z' \in H_i(V)$, отличной от z, то согласно условию строгой беспереборности (6.11) получим взвешенную цепочку $(u_{r-1} z' v_{r-1}, L_r')$ такую, что $L_r \cap L_r' = \varnothing$, т. е. $x \overline{\in} L_r'$, а это противоречит свойству вывода, согласно которому он управляем цепочкой x. Отсюда следует, в частности, что всякая строгая беспереборная грамматика G однозначна, т. е. для любой цепочки $x \in L(G)$ существует единственный вывод в G взвешенной цепочки (x, L), ассоциированной с x. Кроме того, всякая строго беспереборная грамматика G допускает беспереборный синтаксический анализ разверткой. Для этого достаточно рассмотреть максимальный по длине вывод в G, управляемый произвольно выбранной цепочкой $x \in F(\mathfrak{A})$. Тогда $x \in L(G)$ в том и только в том случае, когда данный вывод завершается взвешенной цепочкой (x, L), ассоциированной с x.

Дополнительные ограничения, налагаемые на параметры строго беспереборных грамматик, позволяют, в частности, решить проблему параметризации известных классов кс-грамматик, ориентированных на беспереборный синтаксический анализ разверткой (см. § 6.3). Рассмотрим множество P всех беспереборных грамматик $G = (R, A, \varphi)$, параметры которых удовлетворяют соотношению

$$\varphi(\psi_i, z) = \{(u\psi_i v,\ uL(u) \mid \text{для всех } u \in L_i\}, \qquad (6.14)$$

где $L_i \subseteq F(\mathfrak{A})$ и $L(u)$ — некоторый язык (вообще говоря, зависящий от u). Пусть G — произвольно выбранная $LL(f)$-грамматика. Нетерминал $\psi_i \in V_н$ имеет префикс u, если в G существует левый вывод $\sigma \overset{*}{\to} x$, где $x = u\psi_i$. В качестве языка L_i, входящего в соотношение (6.14), выберем множество всех префиксов нетерминалов ψ_i. Тогда, конкретизируя язык $L(u)$ как некоторое множество прообразов различающей функции f $LL(f)$-грамматики $\tilde{G}$, можно доказать, что существует строго беспереборная грамматика $G \in P$, эквивалентная грамматике $\tilde{G}$. Накладывая дальнейшие ограничения на параметры строго беспереборных грамматик, можно решить проблему параметризации строгих $LL(f)$-, $LF(k)$- и $LL(k)$-грамматик [126].

Теория параметрических грамматик рекурсивного и инверсно-рекурсивного типа может быть положена в основу параметрических моделей, ориентированных на беступиковый двусторонний ПР-анализ (см. § 6.3).

Пусть $G = (P, A, \varphi)$ — произвольная ПГРТ, где $P : \{p : \psi \to w\}$ — схема кс-грамматик, соответствующая системе уравнений R; $A = \{\Delta_p \mid p \in P\}$ — совокупность бинарных отношений, функциональных по второй компоненте; $\varphi : P \to A$ — отображение, сопоставляющее каждой продукции $p \in P$ бинарное отношение $\Delta_p \in A$, $\varphi(p) = \Delta_p$. С рассмотренным выше понятием управляемого вывода ассоциирована концепция α-выводимости [89, 90]: в ПГРТ G из цепочки $z_1 = u\varphi v$ непосредственно α-выводима цепочка $z_2 = uwv$ с управлением терминальной цепочкой $x = ux'$ $(z_1 \overset{x}{\underset{G}{\sqsubseteq}} z_2)$, если есть продукция $p : \psi \to \omega \in P$, причем в отношении $\Delta_p = \varphi(p)$ найдется пара $(z_1 L) \in \Delta_p$ такая, что $x' \in L$. Транзитивное замыкание отношения $\overset{x}{\underset{G}{\sqsubseteq}}$ приводит к отношению α-выводимости $z \overset{x}{\underset{G}{\vdash}} z'$, так что существует последовательность $z_0, z_1, \ldots z_h \in F(V)$, где $z = z_0$, $z' = z_h$, причем при любом $i = 0, 1, \ldots, k-1$, $z_i \overset{x}{\underset{G}{\sqsubseteq}} z_{i+1}$. Язык, порожденный ПГРТ G, представляет собой множество терминальных цепочек $L(G) = \{x \mid \sigma \overset{x}{\underset{G}{\vdash}} x\}$, α-выводимых из аксиомы σ.

Пусть $z = u\psi v$ — произвольная строка вывода в кс-грамматике P и $z' = uwv$ получена из z в результате применения продукции $p : \psi \to w \in P$. Строго беспереборная ПГРТ $G^* = (P, A^*, \varphi^*)$ называется совершенной, если для любого бинарного отношения $\Delta_p \in A^*$ выполняется условие $(z, L) \in \Delta_p$ тогда и только тогда, когда $L = \{x' \mid \sigma^* \overset{*}{\Rightarrow} z \overset{p}{\Rightarrow} z' \overset{*}{\Rightarrow} ux'\}$.

К важным проблемам теории кс-языков относится проблема Гинзбурга [19], состоящая в построении грамматических и автоматных моделей, характеризующих класс однозначных кс-языков. Эта проблема решается теоремой, устанавливающей критерии однозначности кс-языков в терминах совершенных ПГРТ [125].

Теорема 6.10. *кс-Язык L однозначный тогда и только тогда, когда существует совершенная ПГРТ $G^* = (P, \mathfrak{A}^*, \varphi^*)$ такая, что $L = L(G^*)$.*

В терминах α-выводимости естественно формулируется процесс управляемого двустороннего вывода в ПГРТ G (ПР (α)-вывод). Из пары (z_1, z_1') непосредственно ПР (α)-выводима пара (z_2, z_2') при управлении терминальной цепочкой x $(z_1, z_1') \overset{x}{\underset{G}{\models\!=\!\models}} (z_2, z_2')$, если $z_1 \overset{x}{\underset{G}{\models\!=\!\models}} z_2$ и $z_2' \overset{x}{\underset{G}{\models\!=\!\models}} z_1'$. Транзитивное замыкание отношения $\overset{x}{\underset{G}{\models\!=\!\models}}$ приводит к отношению ПР (α)-выводимости $\overset{x}{\underset{G}{\vdash}}$ [89, 90]. Цепочка x принадлежит $L(G)$ тогда и только тогда, когда существует ПР (α)-вывод $W = (\sigma, x) \overset{x}{\underset{G}{\models\!=\!\models}} (z, z')$ такой, что $z = z'$ либо $z \overset{x}{\underset{G}{\models\!=\!\models}} z'$. Вывод W, удовлетворяющий указанным условиям, правильный. Тупиковым (или тупиком) будем считать ПР (α)-вывод W в грамматике G, не являющийся продолжением правильного ПР (α)-вывода, причем в G не существует правильного ПР (α)-вывода, до которого может быть продолжен вывод W. Грамматика G называется ПР (α)-беступиковой (ориентированной на беступиковый ПР-анализ), если наличие для цепочки $x \in F(\mathfrak{A})$ по крайней мере одного тупика означает, что $x \overline{\in} L(G)$ [125].

Теорема 6.11. 1) *Произвольная совершенная ПГРТ G^* ориентирована на беступиковый однозначный ПР (α)-анализ и*

2) *классы $LL(k)$-, $LLS(k)$-, $LF(k)$-, $LL(R)$-грамматик допускают параметризацию, ориентированную на беступиковый* ПР (α)-*анализ* [89, 90].

Пример 8. Рассмотрим зеркальный язык $L = \{x, x^{-1} \mid x \in F(\mathfrak{A})\}$, $\mathfrak{A} = \{a, b\}\}$, порождаемый ПГРТ $G = (P, A, \varphi)$, где

$$P = \begin{cases} p_1 : \sigma \to a\sigma a, \\ p_2 : \sigma \to aa, \\ p_3 : \sigma \to b\sigma b, \\ p_4 : \sigma \to bb, \end{cases}$$

$$A = \begin{cases} \Delta_1 = \{(z\sigma z^{-1}, aF(\mathfrak{A})) \mid \text{для любого } z \in F(\mathfrak{A})\}, \\ \Delta_2 = \{(z\sigma z^{-1}, aaF(\mathfrak{A})) \mid \text{для любого } z \in F(\mathfrak{A})\}, \\ \Delta_3 = \{(z\sigma z^{-1}, bF(\mathfrak{A})) \mid \text{для любого } z \in F(\mathfrak{A})\}, \\ \Delta_4 = \{(z\sigma z^{-1}, bbF(\mathfrak{A})) \mid \text{для любого } z \in F(\mathfrak{A})\}, \\ \varphi(p_i) = \Delta_i \quad (i = 1, 2, 3, 4). \end{cases}$$

Проиллюстрируем процесс ПР (α)-вывода по ПГРТ G на примере цепочки $x = aabbaa \in L(G)$, $(\sigma, x) \overset{x}{\underset{G}{\models\!=\!\models}} (a\sigma a, aa\sigma, aa)$. Из первой компоненты заключительной пары непосредственно α-выводима ее вторая компонента, откуда следует правильность анализируемой цепочки.

Техника управления левосторонним выводом, положенная в основу ПГРТ, может быть использована при организации беступикового двустороннего ПР-анализа. Двойственным образом в процессе управления ПР-анализом применим аппарат ПГИРТ. В работах [89, 125] предложены двусторонние параметрические грамматики, в которых управление процессами развертки и свертки разделено. По каждому из направлений можно сочетать различные способы управления разверточного типа. В частности, посредством двусторонних грамматик могут быть промоделированы стратегии левостороннего анализа.

Особый интерес представляет изучение параметрических моделей, по которым могут быть сконструированы анализаторы, функционирующие с временной сложностью, пропорциональной длине анализируемых цепочек. К таким моделям относятся грамматики ПГРТ G, в которых множество A бинарных отношений ассоциировано с совокупностью регулярных языков, что позволяет существенно упростить процесс анализа с помощью предварительной конечно-автоматной обработки входных цепочек [2].

Концепция предварительной конечно-автоматной обработки допускает обобщение на случай двустороннего ПР-анализа. Для такого обобщения каждому регулярному языку L_p, ассоциированному с бинарным отношением Δ_p, сопоставим конечный автомат $\overrightarrow{A}_p = (S_p \Sigma_p \cup \{\times\}, \delta_p, L_p^0, F_p)$ такой, что $L_p = L(\overrightarrow{A}_p)$, где S_p — множество состояний, $\Sigma_p \cup \{\times\}$ — входной алфавит, δ_p — функция переходов, L_p^0 — начальное состояние, F_p — множество заключительных состояний автомата $\overrightarrow{A}_p$. Автоматы $\overrightarrow{A}_p$ $(p \in P)$ назовем левосторонними. Всякому левостороннему автомату $\overrightarrow{A}_p$ соответствует правосторонний автомат $\overleftarrow{A}_p = (\mathfrak{B}(S_p), \Sigma_p \cup \{\times\}, \tilde{\delta}_p, \{F_p\})$ без заключительных состояний, где $\mathfrak{B}(S_p)$ — булиан над множеством s_p, элементы которого (подмножества в S_p) являются состояниями автомата $\overleftarrow{A}_p$; $\Sigma_p \cup \{\times\}$ — его входной алфавит; $\tilde{\delta}_p(S, a) = \{s \mid \delta_p \cdot (s, a) \in S, S \subseteq S_p, a \in \Sigma_p \cup \{\times\}\}$ — функция переходов; F_p — начальное ссстояние автомата $\overleftarrow{A}_p$. Правосторонний автомат $\overleftarrow{A}_p$ может быть построен по левостороннему автомату $\overrightarrow{A}_p$ по аналогии с алгоритмом перехода от праволинейной грамматики к леволинейной, предложенным в § 5.4.

Для предварительной конечно-автоматной обработки используется техника следов [100]. Автоматы $\vec{B} = \times_{p \in P} \vec{A}_p$ и $\overleftarrow{B} = \times_{p \in P} \overleftarrow{A}_p$ движутся по входной ленте навстречу друг другу (рис. 35) ($\times$ — декартово произведение автоматов). Каждая ячейка входной ленты разделена на три секции. Средняя секция содержит соответствующий символ анализируемой цепочки. В левую (правую) секцию помещаются конфигурации состояний автомата $\vec{B}(\overleftarrow{B})$,

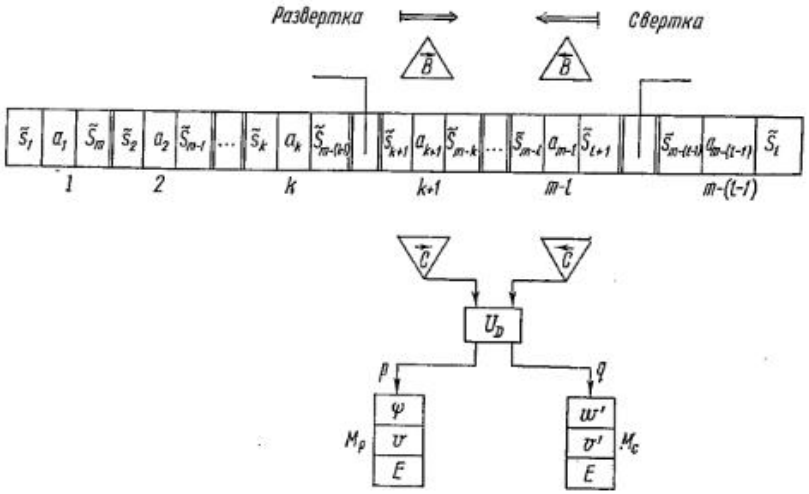


Рис. 35.

вырабатываемые в процессе предварительной конечно-автоматной обработки. После достижения противоположных концов анализируемой цепочки автоматы $\vec{B}$ и $\overleftarrow{B}$ выключаются и начинает функционировать магазинный ПР-анализатор A_G. При этом включаются ассоциированные с автоматами $\vec{B}$ и $\overleftarrow{B}$ контролирующие устройства $\vec{C}$ и $\overleftarrow{C}$, которые по результатам конечно-автоматной обработки проверяют терминальные контексты, допускающие применение продукций p и q в процессе соответственно развертки и свертки. Затем ПР-анализатор A_G функционирует по схеме, описанной в § 6.3.

Параметрические модели с регулярным управлением, ориентированные на беступиковый однозначный ПР-анализ, называются r-моделями.

Справедливо следующее утверждение [124, 125].

Теорема 6.12. *Для произвольной r-модели D может быть сконструирован магазинный ПР-анализатор, A_D с предварительной конечно-автоматной обработкой такой, что $L(A_D) = L(D)$. При этом временная сложность алгоритма ПР-анализа, реализованного в A_D, пропорциональна длине анализируемой цепочки.*

Из этой теоремы следует, в частности, что магазинные ПР-анализаторы с предварительной конечно-автоматной обработкой можно применять при конструировании средств математического обеспечения для многопроцессорных вычислительных комплексов.

Параметрические модели грамматик, ориентированные на ПР-анализ, могут быть применены и для организации управления неоднозначным ПР-анализом, необходимость в котором подтверждается практикой использования современных языков программирования. В работах [89, 90] рассматривается понятие скобочной ПГРТ, отражающей сущность неоднозначности в терминах пар согласованных скобок — точек ветвления и слияния. Внутренняя память ПР-анализатора A_G, ассоциированного с некоторой скобочной ПГРТ G, содержит кроме магазинов M_p и M_c также по одному бс-магазину (см. § 6.1) для каждого из направлений ПР-анализа. бс-Магазин функционирует как бобслей при поиске согласованных пар скобок, достигнутых по обоим направлениям ПР-анализа, и как магазин — когда соответствующая пара скобок достигнута по одному из направлений. Автомат A_G называется бс-анализатором, соответствующим скобочной ПГРТ G. Справедливо следующее утверждение [89, 90].

Теорема 6.13. *Произвольная скобочная ПГРТ G допускает беступиковый ПР-анализ на соответствующем бс-анализаторе.*

Метод неоднозначного ПР-анализа может быть проиллюстрирован на примере существенно неоднозначного языка

$$L = \{a^k b^k c^m d^n e^n\} \cup \{a^k b^m c^m d^n e^s\} \cup \{a^k b^m c^n d^n e^s\},$$

где $k, m, n, s = 1, 2, \ldots$, порожденного скобочной ПГРТ [89].

Естественным обобщением двусторонней стратегии является концепция многослойного анализа, состоящая в разметке входной программы связанными парами указателей, ограничивающими независимо друг от друга анализируемые фрагменты программы, и их совместной ПР- или РП-обработке. В работах [124, 125] предложены параметрические модели языков, ориентированные на проведение эффективного многослойного ПР-анализа. Заметим, что методы параллельного двустороннего анализа допускают эффективную реализацию в однородных структурах, элементной базой которых являются автоматы над внутренней памятью (см. § 6.1).

Стратегия двустороннего анализа и ее обобщения могут быть использованы при конструировании анализаторов для гибких и достаточно сложных систем математического обеспечения, ориентированных на многопроцессорные вычислительные комплексы,

а также при решении задач отладки программ, их контроля и локализации ошибок в системах программирования коллективного пользования, работающих в режиме диалога с разделением времени.

§ 6.5. Метаязык СМ-грамматик и проблемы трансляции

Решение проблемы синтаксического анализа и контроля языков программирования — одной из главных проблем в современной теории и практике программирования (см. § 6.1—6.3) — тесно связано с разработкой эффективных форм представлений грамматик анализируемых языков. Достаточно полно отражают специфику алгоритмов анализа и контроля метаязыки распознавающего типа. К таким метаязыкам относится, в частности, метаязык СМ-грамматик — форма представления n-магазинных автоматов, ориентированная на описание синтаксиса и позволяющая учитывать семантические эквиваленты элементов синтаксического разложения анализируемых программ [15, 129]. Отличительной особенностью СМ-грамматик является сочетание свойств грамматик распознающего и порождающего типа, а также выполнение функций нетерминальных символов именами, обозначающими комплексы продукций.

Выбор продукции из комплекса связан с выполнением дополнительной операции над памятью, состоящей из n магазинов ($n \geqslant 0$). Строго говоря, под СМ-грамматикой подразумевается тройка $\mathfrak{G} = (\tilde{\mathfrak{G}}, R, \tilde{r})$, где $\tilde{\mathfrak{G}} = (\mathfrak{A}, V, \Gamma)$ — система алфавитов грамматики $\mathfrak{G}$, $\mathfrak{A}$ — терминальный алфавит, причем пустой символ e принадлежит $\mathfrak{A}$, V — вспомогательный алфавит, в котором выделен специальный символ $\wedge \in V$, Γ — магазинный алфавит такой, что $\Gamma \supset V$, R — множество имен, соответствующих комплексам продукций определенного вида, причем $R \cap V = \varnothing$ и $\Gamma \subset (R \cup V)$, $\hat{r} = \{r_0, r_\varnothing\}$ — выделенные имена комплексов продукций СМ-грамматики, r_0 — аксиома грамматики $\mathfrak{G}$ (имя начального комплекса продукций); $r_\varnothing$ — имя заключительного комплекса (комплекса с пустым множеством продукций).

Продукции, входящие в комплексы грамматики $\mathfrak{G}$, имеют следующий вид:

$$a \xrightarrow{w_m(\alpha_1, \ldots, \alpha_n)} r, \tag{6.15}$$

где $a \in \mathfrak{A}$ — символ терминального алфавита, $w_m(\alpha_1, \ldots, \alpha_n)$ — n-арное отношение, определяющее вид операции над памятью ($m = 0, 1, 2$), α_i — цепочка в магазинном алфавите Γ при любом i, $1 \leqslant i \leqslant n$, $r \in R$ — имя комплекса продукций преемников, к которым осуществляется переход после применения данной продукции при условии, что выполняется отношение w_m.

При $m = 0$ никакие операции над магазинами не производятся. При $m = 1$ ($m = 2$) в магазин с номером i при всех $i = 1, \ldots, n$ последовательно слева направо записывается (стирается) цепочка $\alpha_i \neq \wedge$. Операция записи выполняется безусловно, а стирания — при условии, что цепочка $\alpha_i \neq \wedge$ в продукции грамматики и цепочка, содержащаяся в верхних ячейках i-го магазина, совпадают. Если эти цепочки не совпадают хотя бы при одном i, то соответствующая продукция неприменима и стирание из магазинов не производится. Если $\alpha_i = \wedge$, то содержание i-го магазина не изменяется.

Метаязык СМ-грамматик удобен как для порождения, так и для распознавания. Порождение некоторой цепочки языка, заданного СМ-грамматикой $\mathfrak{G}$, начинается с применения какой-нибудь продукции, которая принадлежит начальному комплексу, помеченному аксиомой r_0. Примененная продукция определяет первый символ порождаемой цепочки, необходимую операцию над магазинами, а также последующие применимые продукции-преемники, из которых выбирается очередная продукция; порождение цепочки завершается применением продукции, имеющей в первой части заключительное состояние $r_\varnothing$. Распознавание принадлежности исходной цепочки заданному языку сводится к проверке вхождения первого символа данной цепочки в левую часть одной из продукций начального комплекса r_0; каждый следующий символ цепочки должен встречаться в левой части одной из применимых продукций-преемников, а последний определен продукцией с $r_\varnothing$ в правой части. Как при порождении, так и при распознавании применение каждой продукции сопровождается определенными предписанными продукцией действиями над магазинами. Кроме того, в начале и в конце процесса содержимое всех магазинов должно быть пустым.

Язык $L(\mathfrak{G})$ порождается (распознается) СМ-грамматикой $\mathfrak{G}$, если в $\mathfrak{G}$ порождаются (распознаются) те и только те цепочки, которые принадлежат заданному языку.

Пример 1. Любой конечный язык $L_1 = \{x_1, x_2, \ldots, x_k\}$, где $x_i = a_{i_1}, a_{i_2}, \ldots, a_{i_{n_i}}$ — непустая цепочка в алфавите $\mathfrak{A} = \{a_1, a_2, \ldots, a_m\}$ при любом $1 \leqslant i \leqslant k$, порождается СМ-грамматикой $\mathfrak{G}_1$ с совокупностью комплексов

$$r_0 = \{a_{11} \xrightarrow{w_0} r_{12},\ a_{21} \xrightarrow{w_0} r_{22},\ \ldots,\ a_{k_1} \xrightarrow{w_0} r_{k_2}\},$$
$$r_{12} = \{a_{12} \xrightarrow{w_0} r_{13}\},\ r_{22} = \{a_{22} \xrightarrow{w_0} r_{23}\},\ \ldots,$$
$$r_{k_2} = \{a_{k_2} \xrightarrow{w_0} r_{k_3}\},\ \ldots,$$
$$r_{1n_1} = \{a_{1n_1} \xrightarrow{w_0} r_\varnothing\},\ r_{2n_2} = \{a_{2n_2} \xrightarrow{w_0} r_\varnothing\},\ \ldots,$$
$$r_{kn_k} = \{a_{kn_k} \xrightarrow{w_0} r_\varnothing\}.$$

(В данном примере при интерпретации продукций СМ-грамматики магазинная память не используется, в таком случае пометки w_0 могут опускаться).

Пример 2. Универсальный язык L_2, состоящий из всех непустых цепочек в алфавите $\mathfrak{A} = \{a_1, a_2, \ldots, a_m\}$, порождается СМ-грамматикой $\mathfrak{G}_2$ с комплексом продукций

$$r_0 = \{a_i \Rightarrow r_0,\ a_i \Rightarrow r_\varnothing \mid i = 1, 2, \ldots, m\}.$$

Пример 3. Язык $L_3 = \{a^n b a^n \mid n = 0, 1, 2, \ldots\}$ порождается СМ-грамматикой $\mathfrak{G}_3$ с терминальным алфавитом $\mathfrak{A}_3 = \{a, b\}$, алфавитом магазина $\Gamma_3 = \{a'\}$ и совокупностью комплексов

$$r_0 = \{a \overset{w_1(a')}{\Rightarrow} r_0,\ a \overset{w_1(a')}{\Rightarrow} r_1\},$$
$$r_1 = \{b \Rightarrow r_2\},$$
$$r_2 = \{a \overset{w_2(a')}{\Rightarrow} r_2,\ a \overset{w_2(a')}{\Rightarrow} r_\varnothing\}.$$

Пример 4. Язык $L_4 = \{a^n b^n \mid n = 1, 2, \ldots\}$ порождается СМ-грамматикой $\mathfrak{G}_4$ с терминальным алфавитом $\mathfrak{A}_4 = \{a, b\}$, алфавитом магазина $\Gamma_4 \{a'\}$ и совокупностью комплексов

$$r_0 = \{a \overset{w_1(a')}{\Rightarrow} r_0,\ a \overset{w_1(a')}{\Rightarrow} r_1\},$$
$$r_1 = \{b \overset{w_2(a')}{\Rightarrow} r_2\},$$
$$r_2 = \{b \overset{w_2(a')}{\Rightarrow} r_2,\ b \overset{w_2(a')}{\Rightarrow} r_\varnothing\}.$$

Пример 5. Язык $L_5 = \{xx^{-1}\}$ для пустой цепочки $x \in F(\mathfrak{A}_5)$ порождается СМ-грамматикой $\mathfrak{G}_5$ с терминальным алфавитом $\mathfrak{A}_5 = \{a_1, \ldots, a_m\}$, алфавитом магазина $\Gamma_5 = \{a_1', \ldots, a_m'\}$ и совокупностью комплексов

$$r_0 = \{a_i \overset{w_1(a_i')}{\Rightarrow} r_0,\ a_i \overset{w_1(a_i')}{\Rightarrow} r_1\},$$
$$r_1 = \{a_i \overset{w_2(a_i')}{\Rightarrow} r_1,\ a_i \overset{w_2(a_i')}{\Rightarrow} r_\varnothing\}.$$

В приведенных примерах СМ-грамматики $\mathfrak{G}_1 - \mathfrak{G}_5$ порождают контекстно-свободные языки (см. § 5.2, примеры 1—5, а также § 5.3).

Пример 6. $L_6 = \{x \times x\}$ для любой непустой цепочки $x \in F(\mathfrak{A}_4)$ порождается СМ-грамматикой $\mathfrak{G}_6$ с терминальным алфавитом $\mathfrak{A}_6 = \{a, b, \times\}$ над памятью, состоящей из двух магазинов с алфавитом $\Gamma_6 = \{a', b'\}$. Грамматика $\mathfrak{G}$ имеет следующую совокупность комплексов:

$$r_0 = \{p_{01} : a \xrightarrow{w_1(a',\ \wedge)} r_0,\ p_{02} : b \xrightarrow{w_1(b',\ \wedge)} r_0,\ p_{03} : \times \Rightarrow r_1\},$$

$$r_1 = \{p_{11} : e \xrightarrow{w_2(a',\ \wedge)} r_2,\ p_{12} : e \xrightarrow{w_2(b',\ \wedge)} r_3\},$$
$$r_2 = \{p_{21} : e \xrightarrow{w_1(\wedge,\ a')} r_1,\ p_{22} : e \xrightarrow{w_1(\wedge,\ a')} r_4\},$$
$$r_3 = \{p_{31} : e \xrightarrow{w_1(\wedge,\ b')} r_1,\ p_{32} : e \xrightarrow{w_1(\wedge,\ b')} r_4\},$$
$$r_4 = \{p_{41} : a \xrightarrow{w_2(\wedge,\ a')} r_4,\ p_{42} : b \xrightarrow{w_2(\wedge,\ b')} r_4,$$
$$p_{43} : a \xrightarrow{w_2(\wedge,\ a')} r_\varnothing,\ p_{44} : b \xrightarrow{w_2(\wedge,\ b')} r_\varnothing\},$$

где e — пустое слово, $\wedge$ — вспомогательный символ для «защиты» содержимого магазина. Рассмотрим грамматику $\mathfrak{G}_6$, порождающую язык для цепочки $aab \times aab \in L_6$. В начале соответствующего вывода применяем продукции $p_{01}, p_{01}, p_{02}, p_{03} \in r_0$, которые порождают цепочку $aab \times$ и записывают в первый магазин последовательно a', a', b'. Затем с помощью $p_{12} \in r_1$ считываем b' из первого магазина и переходим к продукции $p_{31} \in r_3$, записывая во второй магазин b' с возвратом к r_1. Далее, применяя последовательность продукций $p_{11}, p_{21}, p_{11}, p_{22}$, полностью переписываем содержимое первого магазина во второй. Наконец, используя $p_{41}, p_{42}, p_{44} \in r_4$, завершаем вывод исходной цепочки.

Пример 7. Язык $L_7 = \{a^n b^n c^n \mid n = 1, 2, \ldots\}$ порождается СМ-грамматикой $\mathfrak{G}_7$ с терминальным алфавитом $\mathfrak{A}_7 = \{a, b, c\}$ над памятью, состоящей из двух магазинов с алфавитом $\Gamma_7 = \{a'\}$. Грамматика $\mathfrak{G}_7$ имеет следующую совокупность комплексов:

$$r_0 = \{p_{01} : a \xrightarrow{w_1(a',\ a')} r_0,\ p_{02} : a \xrightarrow{w_1(a',\ a')} r_1\},$$
$$r_1 = \{p_{11} : b \xrightarrow{w_2(a',\ \wedge)} r_1,\ p_{12} : b \xrightarrow{w_2(a',\ \wedge)} r_2\},$$
$$r_2 = \{p_{21} : c \xrightarrow{w_2(\wedge,\ a')} r_2,\ p_{22} : c \xrightarrow{w_2(\wedge,\ a')} r_\varnothing\}.$$

Цепочка $aabbcc \in L_7$ имеет в грамматике $\mathfrak{G}_7$ следующий вывод. Вначале применяются продукции $p_{01}, p_{02} \in r_0$, которые порождают подцепочку aa и одновременно записывают в оба магазина цепочки $a'a'$, затем — продукции $p_{11}, p_{12} \in r_1$, порождающие подцепочку $aabb$ и стирающие содержимое первого магазина. Наконец, продукции $p_{21}, p_{22} \in r_2$ завершают выход исходной цепочки. Для практических приложений используются более компактные приемы описания языков в терминах СМ-грамматик [30].

В примерах 6 и 7 СМ-грамматики порождают нс-языки (см. § 6.2, примеры 1, 2, а также § 5.2), не являющиеся кс-языками.

Класс языков, порождаемых СМ-грамматиками, совпадает с классом рекурсивно-перечислимых множеств. Это следует из возможности представления в терминах СМ-грамматик средств известных алгоритмических систем, например машин Тьюринга. Установим связь между однемагазинными СМ-грамматиками

и кс-грамматиками (см. § 5.3), аналогом которых являются БНФ, используемые для описания синтаксиса языков программирования [132].

Пусть $G = (\mathfrak{A}, V_н, \sigma, P)$ — некоторая кс-грамматика и $L(G)$ — порождаемый ею язык. Как известно (см. § 5.4), используя дополнительные нетерминалы, можно исключить вхождения в схему P продукций вида $\xi \to u\tau$ и $\xi \to u$, где $u \in F(\mathfrak{A})$ — терминальная цепочка длины $|u| > 1$, $\xi, \tau \in V_н$. Так, продукция $\xi \to a_1 a_2 \tau$ заменима парой продукции $\xi \to a_1\tau_1$, $\tau_1 \to a_2\tau$, где τ_1 — дополнительный нетерминал $a_1, a_2 \in \mathfrak{A}$. Продукции, входящие в схему P, разобьем на две группы: 1) P_1 — множество всех правосторонних продукций грамматики G (см. § 5.4); 2) $P_2 = P \setminus P_1$ — остальные продукции грамматики G.

Рассмотрим одномагазинную СМ-грамматику $\mathfrak{G} = (\mathfrak{A}, V, R, \sigma)$, где $\mathfrak{A}$ — терминальный алфавит, совпадающий с терминальным алфавитом грамматики G, $V = V_н \cup \mathfrak{A}'$ — магазинный алфавит грамматики $\mathfrak{G}$, $\mathfrak{A}' = \{a' \mid$ для любого $a \in \mathfrak{A}\}$, R — совокупность комплексов продукций грамматики G, помеченных символами, входящими в V, σ — метка начального комплекса продукций. Каждой правосторонней продукции $(\xi \to a\tau) \in P_1$ грамматики G сопоставим СМ-продукцию $\xi : a \Rightarrow \tau$, заключительной продукции вида $(\xi \to a) \in P$ — СМ-продукции $\xi : a \Rightarrow M$ и $\xi : a \Rightarrow r_\varnothing$, где M — метка обращения к магазину, т. е. считывания символа из вершины магазина и перехода к комплексу продукций, помеченному данным символом. Каждой кс-продукции $\xi \to z \in P_2$, где $z = u_1\xi_1 u_2 \xi_2 \ldots u_k \xi_k u_{k+1}$, $u_i u_{k+1} \in F(\mathfrak{A})$, $\xi_i \in V_н$ $(i = 1, 2, \ldots, k)$, поставим в соответствие СМ-продукцию $\xi : e \overset{w_1(z')}{\Rightarrow} M$, где $z' = u_1'\xi_1 u_2'\xi_2 \ldots u_k'\xi_k u_{k+1}'$, так что если $u = a_{e_1}a_{e_2}\ldots a_{e_s}$, то $u' = a_{e_1}'a_{e_2}'\ldots a_{e_s}'$. Кроме того, к продукциям СМ-грамматики $\mathfrak{G}$ отнесем продукцию вида $a' : \{a \Rightarrow M, a \Rightarrow r_\varnothing \mid$ для любого $a \in \mathfrak{A}\}$. Начальный комплекс образуют все продукции, помеченные меткой σ. Заключительным состоянием в $\mathfrak{G}$ является $r_\varnothing$. Переход в это состояние при опустошении магазина означает принадлежность цепочки языку $L(\mathfrak{G})$. Докажем, что СМ-грамматика $\mathfrak{G}$ эквивалентна кс-грамматике G, т. е. грамматики G и $\mathfrak{G}$ порождают один и тот же язык $L(G) = L(\mathfrak{G})$.

Рассмотрим произвольную правильную цепочку

$$u = a_1 a_2 \ldots a_n \in L(G) \tag{6.16}$$

и покажем, что $u \in L(\mathfrak{G})$. Согласно (6.16) в грамматике G найдется левый вывод $z_1 z_2 \ldots, z_r$, где $z_1 = \sigma$, $z_r = u$, причем

$$p_1, \ldots, p_{r-1} \in P \tag{6.17}$$

является последовательностью продукций, порождающих данный вывод. По последовательности (6.17) можно построить последовательность СМ-продукций, которая является выводом этой же цепочки и в грамматике $\mathfrak{G}$.

Действительно, продукции $p_i : (\xi \to a\tau) \in P_1$ (или $p_i : (\xi \to a) \in P_1$) поставим в соответствие СМ-продукцию $\xi : a \Rightarrow \tau$ (или $\xi : \{a \overset{w(z')}{\Rightarrow} M, a \Rightarrow r_\varnothing\}$, а продукциям $p_i : (\xi \to z) \in P_2$ — СМ-продукции $\xi : e \overset{w(z')}{\Rightarrow} M$. Для чтения из магазина, в вершине которого находится символ $a' \in V$, введем вспомогательные продукции $a' : \{a \Rightarrow M, a \Rightarrow r_\varnothing\}$. Заметим, что переход в $r_\varnothing$ допустим лишь на заключительном шаге СМ-вывода. Построенный таким образом СМ-вывод порождает цепочку (6.16) в грамматике $\mathfrak{G}$, так что справедливо включение $L(G) \subseteq L(\mathfrak{G})$. Для установления обратного включения $L(\mathfrak{G}) \subseteq L(G)$ достаточно в СМ-выводе произвольной цепочки $u \in L(\mathfrak{G})$ заменить СМ-продукции соответствующими продукциями грамматики G, отбросив вспомогательные продукции вида $a' : a \Rightarrow M$. В результате получим левый вывод цепочки u в грамматике G. Таким образом, справедливо следующее утверждение.

Лемма 6.1. *Для каждого контекстно-свободного языка существует порождающая его одномагазинная СМ-грамматика* $L(\mathfrak{G}) = L$.

Проиллюстрируем приведенную процедуру построения СМ-грамматики $\mathfrak{G}$ на следующем примере.

Пример 8. Пусть схема кс-грамматики имеет вид

$$P = \left\{ \begin{array}{l} \sigma \to a\tau \\ \tau \to \sigma b \\ \sigma \to e \end{array} \right\}.$$

Тогда

$$P_1 = \left\{ \begin{array}{l} \sigma \to a\tau \\ \sigma \to e \end{array} \right\}, \quad P_2 = \{\tau \to \sigma b\}.$$

Соответствующие продукции СМ-грамматики $\mathfrak{G}$ имеют вид

$$\sigma : \left\{ \begin{array}{l} a \Rightarrow \tau \\ e \Rightarrow M, \ e \Rightarrow r_\varnothing \end{array} \right\}, \quad \tau : \{e \overset{w_1(\sigma b')}{\Longrightarrow} M\}.$$

Кроме того, грамматика $\mathfrak{G}$ имеет вспомогательные продукции $b' : \{b \Rightarrow M, b \Rightarrow r_\varnothing\}$. Выберем цепочку $aabb \in L(\mathfrak{G})$, причем

$$\sigma,\ a\tau,\ a\sigma b,\ aa\tau b,\ aa\sigma bb,\ aabb \tag{6.18}$$

является ее левым выводом в грамматике G. Тогда

$$\left. \begin{array}{l} \sigma : a \Rightarrow \tau, \ \tau : e \overset{w_1(\sigma b')}{\Rightarrow} M \\ \sigma : a \Rightarrow \tau, \ \tau : e \overset{w_1(\sigma b')}{\Rightarrow} M \\ \sigma : e \Rightarrow M, \ b' : b \Rightarrow M \\ b' : b \Rightarrow r_\varnothing \end{array} \right\} \tag{6.19}$$

есть вывод цепочки $aabb$ в СМ-грамматике $\mathfrak{G}$, причем, если в выводе (6.19) заменить СМ-продукции соответствующими кс-продукциями, а вспомогательные СМ-продукции отбросить, получим первоначальный левый вывод (6.18) в грамматике G.

Рассмотрим произвольную n-магазинную СМ-грамматику $\mathfrak{G}$, порождающую некоторый язык $L(\mathfrak{G})$. Для грамматики $\mathfrak{G}$ можно построить n-магазинный автомат A (см. § 6.2) (вообще говоря, недетерминированный) воспринимающий язык $L(\mathfrak{G})$. Состояниями такого автомата являются метки продукций грамматики $\mathfrak{G}$, причем в каждом состоянии автомат A может одновременно выполнять совокупность элементарных действий, предписанных соответствующей СМ-продукцией. Начальным состоянием автомата A является аксиома грамматики $\mathfrak{G}$, отмечающая начальный комплекс продукций, а заключительным состоянием — соответствующий символ грамматики $\mathfrak{G}$. Отсюда следует, что и для одномагазинной СМ-грамматики $\mathfrak{G}$ существует одномагазинный автомат A, воспринимающий язык $L(\mathfrak{G})$. Однако, как известно, одномагазинные автоматы могут воспринимать лишь кс-языки. Поэтому одномагазинные СМ-грамматики также порождают только кс-языки.

Таким образом, согласно доказанной выше лемме справедлива следующая теорема [132].

Теорема 6.14. *Язык L порождается одномагазинной СМ-грамматикой тогда и только тогда, когда L — кс-язык, т. е. одномагазинные СМ-грамматики равносильны (в смысле порождения языков) кс-грамматикам.*

Заметим, что одномагазинная СМ-грамматика, построенная описанным выше способом, сохраняет основные свойства соответствующей кс-грамматики. В частности, однозначным кс-грамматикам соответствуют детерминированные одномагазинные СМ-грамматики, представляющие класс детерминированных магазинных автоматов (см. § 6.2). Кроме того, для кс-грамматики, представленной нормальной формой Грейбах [147], правые части продукций в которой начинаются терминальными символами, можно построить эквивалентную одномагазинную СМ-грамматику без пустого слова в левых частях СМ-продукций.

Пример 9. Пусть схема кс-грамматики G' имеет вид

$$P = \begin{cases} \sigma \to a\sigma b, \\ \sigma \to a\psi, \\ \psi \to b. \end{cases}$$

Соответствующие продукции СМ-грамматики $\mathfrak{G}'$ имеют вид

$$\sigma : \{a \overset{w_1(b')}{\Rightarrow} \sigma,\ a \to \psi\},\ \psi : \{b \to M,\ b \to r_{\varnothing}\}.$$

Кроме того, грамматика $\mathfrak{G}$ имеет вспомогательные продукции $b' : \{b \to M,\ b \to r_{\varnothing}\}$. Как и в примере 8, для построенных грамматик

$$L(G') = L(\mathfrak{G}') = \{a^n b^n \mid n = 1, 2, \ldots\}.$$

Из приведенных выше рассуждений следует, что переход от задания языков кс-грамматиками (в частности, БНФ) к эквивалентным СМ-грамматикам не сложен. Моделирование левого вывода в терминах СМ-грамматик позволяет использовать их для двустороннего ПР-анализа [125].

Пусть D — параметрическая модель, ориентированная на беступиковый однозначный ПР-анализ, A_D — соответствующий магазинный ПР-анализатор такой, что $L(A_D) = L(D)$. Для кс-грамматики P, которая является параметром модели D, построим эквивалентную СМ-грамматику $\mathfrak{G}$. Функционирование ПР-анализатора A_D может быть организовано в соответствии с СМ-грамматикой $\mathfrak{G}$ и аппаратом управления, заложенным в модели D. При этом СМ-продукции по развертке выполняются в режиме записи, а при свертке — в режиме чтения. Такая организация функционирования ПР-анализатора A_D называется **представлением в форме двусторонней СМ-грамматики $\mathfrak{G}_D$.**

Теорема 6.15. *Произвольный ПР-анализатор A_D представим в форме двусторонней СМ-грамматики $\mathfrak{G}_D$.*

Пусть D — некоторая r-модель, A_D — эквивалентный ПР-анализатор с предварительной конечно-автоматной обработкой (см. § 6.4).

Следствие. ПР-анализатор A_D представим в форме двусторонней СМ-грамматики $\overrightarrow{\mathfrak{G}_D}$.

Метаязык СМ-грамматик может быть использован при решении задачи анализа и синтеза в проблеме трансляции с одного языка программирования на другой, а также в параметрических системах программирования. В работе [129] предлагается уточнение понятия продукций СМ-грамматик, удобное для описания процесса перевода цепочки (программы) входного языка на некоторый промежуточный язык, цепочки которого представляют собой последовательности операторов обращения к семантическим подпрограммам, причем фактическими параметрами последних являются символы входной цепочки.

При трансляции с класса входных языков в левых частях СМ-продукций допускаются совокупности синтаксически однотипных терминальных символов или их сочетаний, принадлежащих к одному или нескольким входным языкам и требующих одинаковой (с точностью до фактических параметров) семантической обработки. С помощью инвариантного относительно класса входных языков ядра параметрической системы, описанного в терминах СМ-продукций, устанавливается соответствие между символами

входной программы и их семантическими эквивалентами в выходных языках.

Пусть T — параметрическая система транслирующего типа. При описании системы T могут быть использованы СМ-продукции вида

$$X \overset{w_m(v)}{\Rightarrow} \delta \times \rho \times n \times P. \qquad (6.20)$$

Здесь $X = (x)$ — совокупность символов входного алфавита, δ, ρ принадлежат выходному алфавиту и означают имена соответствующих семантических подпрограмм, n — натуральное число (номер одной из продукций), $0 \leqslant n \leqslant N$ (N — общее число продукций, задающих систему T), P — некоторый кортеж номеров продукций, задающих систему T. Внутренняя память состоит из регистра ψ, хранящего в каждый данный момент номер обозреваемой продукции, и трех магазинов A, B, C. Магазин A используется для трансляции парных символов, однозначно соответствующих друг другу (например, открывающая и закрывающая скобки), магазин B — при обработке символов, семантика которых определяется правым контекстом, магазин C — для обработки символов, позволяющих прогнозировать возможные пути синтаксического анализа. В начале работы транслятора все магазины пусты, а регистр ψ содержит единицу.

Синтаксический анализ и построение семантического эквивалента некоторой входной цепочки осуществляются при ее однократном (бесперебором) посимвольном просмотре слева направо. Обработка каждого входного символа связана с последовательным пересмотром продукций от продукции с номером $'\psi$ до нахождения применимой продукции вида (6.20) или обнаружения во входной цепочке синтаксической ошибки. Признаком синтаксической ошибки является попытка чтения из пустых магазинов, а также наличие хотя бы одного непустого магазина после завершения работы. Информация о ней выдается с помощью специальной продукции

$$\text{СОШ}\ \alpha, \qquad (6.21)$$

где α — имя подпрограммы, определяющей тип допущенной ошибки. Продукция вида (6.20) неприменима тогда и только тогда, когда:

1) просматриваемый символ входной цепочки не совпадает ни с одним символом, входящим в совокупность $x \in X$ из левой части продукции (6.20), соответствующей выбранному входному языку;

2) просматриваемый символ совпадает по крайней мере с одним указанным символом, но $m = 2$, а символ $v \in V$ не совпадает с символом, хранящимся в вершине магазина A, и просмотр неприменимой продукции приводит к увеличению содержимого регистра ψ на единицу, что соответствует выбору очередной продукции, применимость которой подлежит проверке.

Выполнение каждой продукции вида (6.20) состоит в определении семантической реакции (в зависимости от символов δ и ρ) и установлении номера продукции преемника. Определение семантической реакции заключается в следующем.

1. $m = 0$. Если символ δ непуст, а $\rho = e$, то δ дописывается к выходной цепочке. Если $\delta = e$ и $\rho \neq e$, то символ ρ дописывается к выходной цепочке, причем его адрес запоминается в вершине магазина B для возможного последующего переопределения в соответствии с правым контекстом. Просматриваемый символ в этом случае условно определен. Наконец, если $\delta \neq e$ и $\rho \neq e$, то символ ρ засылается в выходную цепочку по адресу, считываемому из вершины магазина B. Эта ситуация соответствует переопределению ранее условно определенного символа. При $\delta = \rho = e$ не возникает никаких реакций, связанных с формированием выходной цепочки (что соответствует обработке символов, не несущих семантической нагрузки, например комментариев в АЛГОЛе).

2. $m = 1$ ($m = 2$). Кроме действий, подлежащих выполнению при $m = 0$, в вершину магазина A засылается (из вершины A считывается) символ v, причем при $m = 2$ в случае несовпадения символа в вершине магазина A с символом $v \in V$, указанным в рассматриваемой продукции, последняя по определению неприменима.

Установление номера преемника производится следующим образом.

При $P \neq \varnothing$ все элементы преемника засылаются в магазин C. Если при этом $n = 0$, то продукция-преемник определяется путем увеличения содержимого ψ на единицу. В противном случае n является номером продукции-преемника и засылается в регистр ψ. Если $P = \varnothing$ и $n \neq 0$, продукция-преемник определяется путем засылки n в регистр ψ. При $n = 0$ и пустых магазинах A, B, C анализ завершен успешно; если хотя бы один из этих магазинов непуст, то это свидетельствует о наличии синтаксической ошибки. Для обращения к магазинам используются продукции вида

$$\text{СТЕК}\ \beta, \qquad (6.22)$$

где β — имя магазина. Выполнение такой продукции при $\beta = C$ состоит в выборе символа из вершины магазина C и засылке его в регистр ψ для определения номера продукции-преемника по данному символу анализируемой цепочки. Если магазин C оказывается пустым, алгоритм прекращает работу, выдавая информацию о наличии синтаксической ошибки. При выполнении продукции СТЕК β при $\beta = B$ из вершины магазина B считывается очередной символ.

Заметим, что кроме определения семантических эквивалентов интерпретация продукций транслятора T обеспечивает полный синтаксический контроль: 1) обнаруживает все те символы входной цепочки, которые в соответствии с синтаксисом языка не

согласуются с предшествующей подцепочкой (с их левым контекстом); 2) указывает на незавершенность цепочки (если в ней нет ошибок первого вида).

Пример 10. Рассмотрим фрагмент описания грамматики простого арифметического выражения языка АЛГОЛ-60:

⟨арифметическое выражение⟩ :: = ⟨простое арифметическое выражение⟩ #

⟨простое арифметическое выражение⟩ :: = ⟨терм⟩|⟨операция типа сложения⟩ ⟨терм⟩|⟨простое арифметическое выражение⟩ ⟨операция типа сложения⟩ ⟨терм⟩

⟨операция типа сложения⟩ :: = +|−

⟨терм⟩ :: = ⟨множитель⟩|⟨терм⟩ ⟨операция типа умножения⟩ ⟨множитель⟩

⟨множитель⟩ :: = ⟨первичное выражение⟩

⟨операция типа умножения⟩ :: = ×|/

⟨первичное выражение⟩ :: = ⟨число⟩|(⟨арифметическое выражение⟩)|⟨переменная⟩

⟨переменная⟩ :: = ⟨идентификатор⟩|⟨идентификатор⟩[⟨список индексов⟩]

⟨список индексов⟩ :: = ⟨простое арифметическое выражение⟩| ⟨список индексов⟩, ⟨простое арифметическое выражение⟩

⟨идентификатор⟩ :: = ⟨буква⟩|⟨идентификатор⟩⟨буква⟩| ⟨идентификатор⟩⟨цифра⟩

⟨число⟩ :: = ⟨цифра⟩|⟨число⟩⟨цифра⟩

Обозначим кортежи, состоящие из всех букв и цифр, соответственно через Б и Ц; кортеж + — через $\oplus$ и кортеж ×, / — через $\otimes$. В данном случае СМ-грамматика языка описывается следующими 48 продукциями, причем $\mathfrak{A} = \{Б, Ц, \oplus, \otimes, [,], (,), \#\}$:

1) $\oplus \to P_{13} \times e \times 2 \times \varnothing$,
2) $Б \to e \times P_7 \times 6 \times 12$,
3) $Ц \to P_9 \times e \times 10 \times 12$,
4) $(\xrightarrow{w_1(v_1)} P_3 \times e \times 16 \times 25$,
5) СОШ,
6) $Б \to P_8 \times e \times 6 \times \varnothing$,
7) $Ц \to P_8 \times e \times 6 \times \varnothing$,
8) $[\xrightarrow{w_1(v_1)} P_5 \times P_{11} \times 32 \times 45$,
9) СТЕК В × С,
10) $Ц \to P_{10} \times e \times 10 \times \varnothing$,
11) СТЕК С,
12) $\otimes \to P_2 \times e \times 2 \times \varnothing$,
13) $\oplus \to P_1 \times e \times 2 \times \varnothing$,
14) $\# \to P_{14} \times e \times 0 \times \varnothing$,
15) СОШ,
16) $\oplus \to P_{13} \times e \times 17 \times \varnothing$,
17) $Б \to e \times P_7 \times 12 \times 29$,
18) $Ц \to P_9 \times e \times 10 \times 29$,
19) $(\xrightarrow{w_1(v_2)} P_3 \times e \times 16 \times 25$,
20) СОШ,
21) $Б \to P_8 \times e \times 21 \times \varnothing$,
22) $Ц \to P_8 \times e \times 21 \times \varnothing$,
23) $[\xrightarrow{w_1(v_2)} P_5 \times P_{11} \times 32 \times 45$,
24) СТЕК В × С,
25) $) \xrightarrow{w_2(v_1)} P_4 \times e \times 12 \times \varnothing$,
26) $) \xrightarrow{w_2(v_2)} P_4 \times e \times 29 \times \varnothing$,

27) $) \xrightarrow{w_2(v_3)} P_4 \times e \times 41 \times \varnothing$,
28) СТЕК С,
29) $\oplus \to P_1 \times e \times 17 \times \varnothing$,
30) $\otimes \to P_2 \times e \times 17 \times \varnothing$,
31) СТЕК С,
32) $\oplus \to P_{13} \times e \times 33 \times \varnothing$,
33) $Б \to e \times P_7 \times 37 \times 41$,
34) $Ц \to P_9 \times e \times 10 \times 41$,
35) $(\xrightarrow{w_1(v_3)} P_3 \times e \times 16 \times 25$,
36) СОШ,
37) $Б \to P_8 \times e \times 37 \times \varnothing$,
38) $Ц \to P_8 \times e \times 37 \times \varnothing$,
39) $[\xrightarrow{w_1(v_3)} P_5 \times P_{11} \times 32 \times 45$,
40) СТЕК В × С,
41) $\oplus \to P_1 \times e \times 33 \times \varnothing$,
42) $\otimes \to P_2 \times e \times 33 \times \varnothing$,
43) $, \to P_{12} \times e \times 32 \times \varnothing$,
44) СТЕК С,
45) $] \xrightarrow{w_2(v_1)} P_6 \times e \times 12 \times \varnothing$,
46) $] \xrightarrow{w_2(v_2)} P_6 \times e \times 29 \times \varnothing$,
47) $] \xrightarrow{w_2(v_3)} P_6 \times e \times 41 \times \varnothing$,
48) СТЕК С.

Для семантического определения указанной грамматики используются 14 подпрограмм. Каждая из них выполняет некоторую функцию трансляции текущего символа исходной последовательности: P_1 и P_2 — трансляция арифметической операции типа сложения и умножения; P_3 и P_4, P_5 и P_6 — обработка соответственно открывающей и закрывающей, круглой и квадратной скобок; P_7 и P_8 — первой буквы и последующих знаков идентификатора; P_9 и P_{10} — первой и последующих цифр числа; P_{11} — первой буквы в записи переменной с индексом; P_{12} — запятой; P_{13} и P_{14} — соответственно знаков одноместной операции и конца выражения.

Определим для примера грамматическую правильность (неправильность) и семантику цепочек A, $A1$, заданных в рассматриваемом языке, по приведенной выше грамматике. В результате анализа входной цепочки $A : xyz \times (F2[5] - x) + 1 \#$ получаем следующую цепочку имен семантических подпрограмм, над символами которой проставлены соответствующие символы входной цепочки (под именами семантических подпрограмм проставлены номера примененных продукций грамматики):

x	y	z	×	(	F	2	[	5	]	−	x	)	+	1	#
$\underline{P_7}$	P_8	P_8	P_2	P_3	P_7 P_{11}	P_8	P_5	P_9	P_6	P_1	$\underline{P_7}$	P_4	P_1	P_9	P_{14}
2	6	6	9	4	17	22	23	34	11	29	17	24	13	3	11
			12						44			31			14
									46			31			
												25			
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16

Имена семантических подпрограмм, которые могут быть подвергнуты переопределению, подчеркиваются. В данном случае переопределению было подвергнуто второе из трех подчеркнутых имен, что отмечено стрелкой с указанием на ней переопределяющего имени P_{11}. При выполнении этой цепочки подпрограмм получаем последовательность команд некоторой гипотетической одноадресной машины:

$$\begin{array}{l}\rightarrow F2\,[5]\\ -x\\ \times xyz\\ +1,\end{array}$$

где $F2[5]$ соответствует адресу пятого элемента в массиве $F2$, адрес которого (и адреса переменных x и xyz) задается специальной таблицей, полученной при трансляции описания программы. Для цепочки $A1:((a+1)\,\#$ аналогично получаем

(	(	a	+	1	)	#
P_3	P_3	P_7	P_1	P_9	P_4	P_{14}
4	19	17	24	18	11	28
			29		29	31
					25	

Последней выполняемой продукцией является продукция 31, приводящая к чтению из пустого стека C, что означает наличие ошибки во входной цепочке.

Как показал опыт построения систем программирования, предлагаемый метод анализа языков позволяет существенно сократить сроки изготовления таких систем.

§ 6.6. СМ-формализмы и их применение к системам программирования

Естественным обобщением СМ-грамматик являются **СМ-системы** [14], представляющие собой универсальную схему, которая позволяет с общих позиций подойти к рассмотрению таких сложных проблем, как конструирование языков программирования и трансляторов для формального перевода с одного языка на другой, создание метатрансляторов и генераторов транслирующих систем, разработка методики доказательств утверждений о конкретных программах, языках программирования и трансляторах, развитие теории синтаксиса, формализация семантики языков и метаязыков [48].

Характерная особенность СМ-систем заключается в том, что запоминающие устройства того или другого типа вводятся непосредственно в грамматику языка. Как и в СМ-грамматиках, в СМ-системах сочетаются порождающие и распознающие свойства грамматик.

Обычно для каждого формального языка тем или иным способом формулируются его синтаксис и семантика, так что проблема трансляции заключается в построении отображения, которое сопоставляло бы каждой цепочке формального языка его семантический эквивалент в машинном языке. Естественно, что в этом случае отдельное определение семантики формального и машинного языков носит вспомогательный характер, основную роль играет само отображение, при задании которого учитываются семантические особенности рассматриваемой пары языков. Таким образом, можно сформулировать относительное семантическое определение, предполагающее установление связи между цепочками двух языков и перенесение семантики цепочки одного языка на цепочку другого. При этом можно рассматривать не только пары языков формальный — машинный, но и формальный — формальный, машинный — машинный и более общие комбинации, включающие несколько языков, как и в параметрических системах программирования.

Формально связь между цепочками пары языков может быть задана в виде бинарного отношения. Семейство всех таких отношений слишком велико, и обычно рассматриваются только его некоторые подклассы. Если учитывать общий метод задания формальных и машинных языков в виде грамматик, то весьма естественным является рассмотрение подкласса конечно-порожденных отношений. Общий способ задания отношений такого типа заключается в следующем (см. § 5.1). В каждом языке выделяется конечное множество цепочек, между которыми устанавливаются связи, т. е. образуются пары цепочек. Кроме того, задается конечное множество преобразований (правил вывода), которые позволяют из данного конечного множества пар цепочек выводить все другие слова языков. Выводимые слова порождаются заданными преобразованиями в виде пар, причем каждая такая пара принадлежит определяемому отношению. Далее следует описать подкласс отношений, замкнутый относительно фиксированного набора преобразований, или, наоборот, если задан некоторый подкласс отношений, то нужно определить, имеет ли указанный класс конечную систему образующих (связанные пары цепочек) относительно конечного набора преобразований операций (см. § 3.1—3.3).

Пусть $\mathfrak{A}$, $\mathfrak{M}$ — два алфавита, где $\mathfrak{M}$ может совпадать с $\mathfrak{A}$; $L_{\mathfrak{A}}$ и $L_{\mathfrak{M}}$ — соответственно языки в алфавитах $\mathfrak{A}$ и $\mathfrak{M}$; T — конечно-порожденное отношение на языках $L_{\mathfrak{A}}$ и $L_{\mathfrak{M}}$, $T \subseteq F(\mathfrak{A}) \times F(\mathfrak{M})$, так что $T\{(x, y) \mid x \in L_{\mathfrak{A}},\ y \in L_{\mathfrak{M}}\}$, причем $\mathrm{pr}_1 T = L_{\mathfrak{A}}$, $\mathrm{pr}_2 T = L_{\mathfrak{M}}$; θ — отношение эквивалентности на T, т. е. для некоторых пар (x, y) и (x', y') из T справедливо $(x, y) \equiv (x', y')$.

Отношение θ определяет отношения эквивалентности $\theta_{\mathfrak{A}}$, $\theta_{\mathfrak{M}}$ на языках $L_{\mathfrak{A}}$, $L_{\mathfrak{M}}$: $x \underset{\theta_{\mathfrak{A}}}{\equiv} x'$ тогда и только тогда, когда существуют такие $y, y' \in L_{\mathfrak{M}}$, что $(x, y) \underset{\theta}{\equiv} (x'y')$. В этом случае также $y \underset{\theta_{\mathfrak{M}}}{\equiv} y'$.

Отношение T^{θ} называется **θ-трансляцией**, если T — взаимно-однозначное отношение с точностью до выбора представителя из класса эквивалентности по θ, т. е. если по одному члену пары (x, y), например по x, можно найти $y' \underset{\theta_{\mathfrak{M}}}{\equiv} y$ и, наоборот, по y можно найти $x' \underset{\theta_{\mathfrak{M}}}{\equiv} x$. Отношение T^{θ} называется **θ-интерпретацией**, если T — однозначное отношение с точностью до выбора представителя из класса эквивалентности по $\theta_{\mathfrak{M}}$.

Представляется целесообразным расширить понятие интерпретации, сведя к интерпретации любое преобразование исходной цепочки, при котором используется, возможно, лишь частичная информация о данной цепочке (примером может служить построение блок-схемы по программе, написанной в АЛГОЛе). Распространенное определение понятия интерпретации, объединяющее перевод программы в машинный код с одновременным выполнением операций, также включается в данное определение. Понятие трансляции сохраняется только за такими преобразованиями программы из одного языка в другой, которые позволяют, в случае необходимости, восстановить исходную программу. Чтобы сохранить возможность эквивалентных преобразований транслируемых программ, вводится дополнительное отношение θ. Конечно, можно определить отношение θ так, чтобы θ-трансляция совпадала на множестве $F(\mathfrak{M})$ с θ-интерпретацией, однако θ-интерпретация непредполагает порождение соответствующего элемента в множестве $F(\mathfrak{A})$.

Пусть отношение T задано. Тогда **проблема принадлежности** формулируется следующим образом: для любой пары цепочек $(x, y) \in F(\mathfrak{A}) \times F(\mathfrak{M})$ необходимо выяснить, принадлежит ли пара (x, y) отношению T. **Проблема существования** для заданного T состоит в выяснении, существует ли для произвольно выбранного $x \in F(\mathfrak{A})$ цепочка $y \in F(\mathfrak{M})$ такая, что $(x, y) \in T$. Известно несколько способов задания отношения T. Наиболее общие из них — представление отношений в терминах порождающих грамматик (см. гл. 5), определение автомата, индуцирующего данное отношение (см. § 6.1, 6.2), а также системное задание, объединяющее некоторые черты порождающих грамматик и распознающих автоматов (см. § 6.5). Для любого из перечисленных способов возникает **проблема перечислимости**, состоящая в установлении принадлежности пар цепочек (x, y) отношению T. Формализм СМ-систем

является обобщением метаязыка СМ-грамматик и относится к системному способу задания T.

Рассмотренное бинарное отношение T является частным случаем более общего отношения

$$T_{(p, q)} \subseteq F(\mathfrak{A}_1) \times F(\mathfrak{A}_2) \times \cdots \times F(\mathfrak{A}_p) \times F(\mathfrak{M}_1) \times \cdots \times F(\mathfrak{M}_q)$$

при $p = q = 1$. Введение отношения $T_{(p, q)}$ позволяет описать задачу переработки языковой информации с учетом реальных машинных условий и возможности одновременного выполнения нескольких операций, а также решить несколько задач за один просмотр цепочки в реальном масштабе времени.

Пусть $\mathfrak{A}_1, \mathfrak{A}_2, \ldots, \mathfrak{A}_p$ — входные, а $\mathfrak{M}_1, \mathfrak{M}_2, \ldots, \mathfrak{M}_q$ — выходные алфавиты и $\mathfrak{A} = \bigcup_{i=1}^{p} \mathfrak{A}_i$, $\mathfrak{M} = \bigcup_{j=1}^{q} \mathfrak{M}_j$. Рассмотрим декартовы произведения

$$L_{\mathfrak{A}} = L_{\mathfrak{A}_1} \times L_{\mathfrak{A}_2} \times \cdots \times L_{\mathfrak{A}_p},$$
$$L_{\mathfrak{M}} = L_{\mathfrak{M}_1} \times L_{\mathfrak{M}_2} \times \cdots \times L_{\mathfrak{M}_q},$$

где $S_1 = \{L_{\mathfrak{A}_i} \subseteq F(\mathfrak{A}_i) \mid i = 1, 2, \ldots, p\}$ и $S_2 = \{L_{\mathfrak{M}_j} \subseteq F(\mathfrak{M}_j) \mid j = 1, 2, \ldots, q\}$ — семейства соответственно входных и выходных языков. Так, каждая цепочка языка $L_{\mathfrak{A}}$ ($L_{\mathfrak{M}}$) представляет собой упорядоченную последовательность из p (q) цепочек, принадлежащих языкам семейства S_1 (S_2). В общем случае СМ-система состоит из p входных лент, на которых записана цепочка языка $L_{\mathfrak{A}}$, q выходных лент, на которые в процессе работы записывается цепочка языка $L_{\mathfrak{M}}$, и некоторого множества отмеченных продукций.

Введение нескольких входных лент вызвано тем, что часто входную информацию удобно задавать разными способами, причем на каждом шаге она должна вводиться параллельно, а не последовательно, например программа и исходные данные или входная информация и управляющие воздействия. Этим объясняется и введение нескольких выходных лент, на которых записывается перевод входной цепочки.

Каждая продукция СМ-системы включает в себя метку данной продукции, операцию считывания с входных и записи на выходные ленты, операцию с внутренним запоминающим устройством СМ-системы, множество меток продукций, одна из которых применяется после выполнения данной. Внутреннее запоминающее устройство СМ-системы может использовать магазинную память, счетчики, листовую память, различные комбинации запоминающих устройств. Более того, в некоторых случаях удобно использовать память, заданную в виде магнитных барабанов, дисков и лент. Однако для задания конкретной СМ-системы вид памяти и способ оперирования ею должен быть заранее зафиксирован.

Формально **СМ-системой** называется множество отмеченных продукций

$$\{r_\mu < a_{1\mu}, \ldots, a_{p\mu} \# b_{1\mu}, \ldots, b_{q\mu}; \mathfrak{M}_\mu; r_{\mu_1}, \ldots, r_{\mu_s}\rangle \mid \mu \in I\}, \quad (6.23)$$

где r_μ — метка продукции, $a_{i\mu} \in \mathfrak{A}_i$, $b_{j\mu} \in \mathfrak{M}_j$, $\mathfrak{M}_\mu \subseteq \mathfrak{M}$ ($\mathfrak{M}$ — совокупность всех допустимых операций над внутренней памятью СМ-системы, I — конечное множество индексов меток ($i = 1, 2, \ldots, p$; $j = 1, 2, \ldots, q$).

По предположению каждый алфавит $\mathfrak{A}_i$, $\mathfrak{M}_j$ содержит символ пустого слова e, т. е. любой символ a_i, b_j, входящий в продукцию, может совпадать с e. Это означает, что в продукции с данного внешнего входа (выхода) ничего не читается (не записывается). Кроме того, вместо буквы $a_{i\mu}$ может стоять специальный символ $\sim$, означающий, что с i-й ленты считывается любой очередной находящийся на ней символ. Количество продукций СМ-системы определяется числом элементов множества I.

СМ-система функционирует следующим образом. На первом шаге к цепочке $x \in L_{\mathfrak{A}}$, записанной на входных лентах, применяется первая продукция с меткой r_1. Продукция r_μ вида (6.23) применима на данном шаге, если очередными символами, считываемыми с входных лент, являются $a_{i\mu}$ и над внутренней памятью СМ-системы выполняются условия применимости операций, входящих в $\mathfrak{M}_\mu$. В этом случае все символы $a_{i\mu}$ ($i = 1, \ldots, p$) стираются с соответствующих входных лент; символы $b_{j\mu}$ ($j = 1, \ldots, q$) записываются в соответствующие выходные ленты; над внутренней памятью выполняются операции из $\mathfrak{M}_\mu$. На следующем шаге СМ-система переходит к применению одной из продукций r_{μ_t} ($t = 1, \ldots, s$). Если ни одна из продукций r_{μ_t} не применима, система останавливается. Среди меток СМ-системы есть метка заключительного состояния $r_\varnothing$, означающая конец работы. СМ-система переводит входную цепочку $x \in L_{\mathfrak{A}}$ в выходную $y \in L_{\mathfrak{M}}$, если существует последовательность продукции

$$r_{\mu_1}, r_{\mu_2}, \ldots, r_{\mu_n}, \quad (6.24)$$

где $r_{\mu_1} = r_1$, $r_{\mu_n} = r_\varnothing$, и на k-м шаге применима продукция r_{μ_k} ($k = 1, 2, \ldots, n-1$), преемником которой является продукция $r_{\mu(k+1)}$. В результате применения продукций последовательности (6.24) цепочка x преобразуется в цепочку y. До начала работы и после перехода к заключительному состоянию память СМ-системы пуста.

Если выходных лент нет, СМ-система вырождена. В этом случае распознается принадлежность входной цепочки x языку $L_{\mathfrak{A}}$. Аналогично, если нет входных лент, перечисляются (порождаются) цепочки языка $L_{\mathfrak{M}}$.

Транслирующие СМ-грамматики (ТСМ-грамматики) являются частным случаем СМ-систем, память которых состоит из n магазинов. В качестве операций над памятью используются операции w_m ($m = 0, 1, 2$), описанные ранее (см. § 6.5). Рассмотрим следующий пример ТСМ-грамматики.

Пусть $L_{\mathfrak{A}} = \{a^n b^n c^n \mid n \geqslant 1\}$, $L_{\mathfrak{M}} = \{a^n b c^n \mid n \geqslant 1\}$ и задана трансляция $T \subseteq L_{\mathfrak{A}} \times L_{\mathfrak{M}}$ такая, что $T = \{(a^n b^n c^n, a^n b c^n) \mid n \geqslant 1\}$. Эта трансляция осуществляется ТСМ-грамматикой с двумя внутренними магазинами одной входной и одной выходной лентами:

$$\begin{aligned}
&r_1 < a \# a;\ w_1(1, 1);\ r_2, r_3 >,\\
&r_2 < a \# a;\ w_1(2, 2);\ r_2, r_3, r_4 >,\\
&r_3 < b \# e;\ w_2(2, \wedge);\ r_3, r_4 >,\\
&r_4 < b \# b;\ w_2(1, \wedge);\ r_5, r_6 >,\\
&r_5 < c \# c;\ w_2(\wedge, 2);\ r_5, r_6 >,\\
&r_6 < c \# c;\ w_2(\wedge, 1);\ r_\varnothing >.
\end{aligned}$$

Формализм, основанный на понятии СМ-системы, может быть использован при описании операционных систем [13], являющихся важной частью средств математического обеспечения современных ЭВМ (см. § 6.1).

Рассмотрим класс СМ-систем, использующих память типа БС (см. § 6.1), с помощью которой информация быстро и равномерно передается на устройства, перерабатывающие ее для минимизации времени их простаивания. Конечный БС состоит из конечного числа ячеек, расположенных в виде ленты. Запись символа в БС производится всегда с одного конца ленты, а считывание — с другого. Записываемый символ занимает самую левую ячейку. Если символ считывается из БС, то он стирается с правого конца и одновременно все символы, записанные в БС, сдвигаются на одну ячейку вправо. Если БС пуст, то выходная ячейка заполняется символом $\wedge$, который исчезает только после записи какого-либо символа.

Обозначим операции в БС следующим образом: запись — $BS_1(x)$, чтение — $BS_2(x)$ (в скобках стоит символ, считываемый из БС или записываемый в БС). Если все ячейки БС заняты, то операция BS_1 неприменима. Если символ, стоящий в правом конце БС, не совпадает с символом x, операция $BS_2(x)$ неприменима. Если СМ-система содержит n БС, то указанные операции имеют вид $BS_i(x_1, \ldots, x_n)$ ($i = 1, 2$). Замена x_k ($k = 1, 2, \ldots, n$) символом $\wedge$ означает, что с данным БС в этой продукции никакие операции не производятся. Символами входного алфавита данного класса СМ-систем являются номера входных устройств и некоторые

вспомогательные символы, вырабатываемые СМ-системой в процессе работы. Выходной алфавит составляют имена команд, которые передаются на исполнение.

Задачи, решаемые операционными системами, могут быть сведены к переводу последовательности директив пользователя в приказы операционной системы. Такая трансляция может быть осуществлена построенной для этой цели СМ-системой. Рассмотрим описание в СМ-формализме простейшей операционной системы ОС-1. Пусть ОС-1 рассчитана на сопровождение n задач в вычислительной системе, имеющей n магазинных лент (по числу задач), m устройств ввода и k устройств вывода. Допустим, что задачи поступают в систему в произвольные моменты времени и обслуживаются вычислителем в порядке их поступления. Пусть на работу вычислителя ОС-1 отводит определенные, вообще говоря произвольные кванты времени. Если задача не решена за отведенный квант времени, то ОС-1 записывает задачу на свободную магнитную ленту и продолжает ее решение только после удовлетворения остальных заявок, принятых машиной для решения к моменту прерывания. Общее число таких заявок не превышает n. ОС-1 запоминает порядок поступления заявок и распределяет внутреннюю память (магнитные ленты) для хранения неоконченных задач. ОС-1 работает по следующей схеме. В начале каждого кванта времени анализируется запрос устройств ввода (в порядке их номеров) на прием задачи. Если запросов нет ни на одной из них, анализируется БС. Если запросов нет и в БС, то ОС-1 выключается на единицу времени. При поступлении входного сигнала a определяется, не превышает ли уже имеющаяся очередь заявок общее число задач n, которые могут одновременно находиться в памяти машины. Если это число равно n, то входной сигнал не допускается в машину. В противном случае номер входного устройства, на котором находится поступившая задача, записывается в БС, т. е. ставится в очередь. После этого из БС в рабочую ячейку считывается номер устройства, содержимое которого должно обрабатываться процессором. Если этим устройством оказывается магнитная лента, то ее содержимое переписывается в процессор, а номер освободившейся ленты заносится в БС. Если этим устройством оказывается входное с номером i, то его содержимое вводится в процессор, а устройство ввода освобождается для приема новой задачи.

После ввода задачи поступает команда σ — «счет», и в течение τ единиц времени эта команда исполняется, т. е. ОС-1 «ждет» время τ. В процессе работы каждая задача выдает в рабочую ячейку данные «задача окончена» — b или «задача не окончена» — $\bar{b}$. Если задача окончена, то из БС считывается в рабочую ячейку номер свободного выходного устройства и выходная информация оконченной задачи поступает на это устройство. Одновременно номер данного устройства заносится в БС. Если задача не окончена, то из БС считывается номер свободной магнитной ленты и содержимое процессора (неоконченная задача) записывается на нее. После этого номер заносится в БС и ОС-1 переходит к повторению цикла обслуживания.

Описанная ОС-1 может быть представлена как СМ-система, содержащая m входных лент, на которые поступает информация о наличии в данном входном устройстве задачи (символы a); одну выходную ленту, на которую ОС-1 выдает приказы входным и выходным устройствам и процессору; три внутренние БС: одна для запоминания последовательности решаемых задач, две другие соответственно для запоминания свободных магнитных лент и выходных устройств. Входной алфавит $\mathfrak{A}$ содержит единственный символ a, соответствующий наличию задачи на данном входном устройстве. Выходной алфавит $\mathfrak{M}$ состоит из следующих символов — кодов соответствующих приказов операционной системы устройствам машины: $\alpha\rho$ — пересылка содержимого магнитной ленты, номер которой содержится в рабочей ячейке; $\beta\rho$ — пересылка в процессор содержимого входного устройства, номер которого задан в рабочей ячейке; γ — подготовка процессора для выполнения новой задачи; $\delta\rho$ — запись содержимого процессора на магнитную ленту, номер которой указан в рабочей ячейке; $\xi\rho$ — включение выходного устройства, номер которого содержится в рабочей ячейке; σ — передача управления задаче.

Символами внутреннего алфавита, используемыми в операциях с БС, для входных устройств считаются целые числа от 1 до m; для магнитных лент — от $m+1$ до $m+n$; для выходных устройств — от $m+n+1$ до $m+n+k$. Символами внутреннего алфавита для операций с рабочей ячейкой — целые числа от 1 до $m+n+k$ и два символа b и $\bar{b}$ — «конец» или «не конец» задачи. Кроме того, в записи операций над памятью в продукциях СМ-системы используются символы $BS_2(\sim)$ и $BS_1(\rho)$: первый соответствует считыванию символа из БС и записи его в рабочую ячейку, второй — записи в БС содержимого рабочей ячейки. Символы τS, стоящие в продукции после $\#$, означают задержку в выполнении перехода к следующей продукции на τ единиц времени. Если τ равно единице времени, символ $1S$ опускается. $R(b)$ — считывание b в рабочую ячейку.

Формальное описание ОС-1 на языке СМ-систем имеет вид

$$r_0\langle a, e, \ldots, e\,\#;\ BS_1(1, \wedge, \wedge); r_1|$$
$$\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots$$
$$e, e, \ldots, e, a\,\#;\ BS_1(m, \wedge, \wedge); r_1|$$
$$e, e, \ldots, e\,\#;\ BS_2(\vee, \wedge, \wedge); r_0|$$

$e, e, \ldots, e \# \rho \# \rho;\ BS_2(\sim, \wedge, \wedge); r_2\rangle$
$r_1\langle e, \ldots, e \# \rho;\ BS_2(\sim, \wedge, \wedge); r_2\rangle$
$r_2\langle e, \ldots, e \# \alpha\rho;\ BS_1(\wedge, '\rho, \wedge); r_3|$
$e, \ldots, e \# \beta\rho;\ BS_1(\wedge, \wedge, \wedge); r_3\rangle$
$r_3\langle e, \ldots, e \# \tau S\sigma;\ BS_1(\wedge, \wedge, \wedge); r_4\rangle$
$r_4\langle e, \ldots, e \# \rho;\ R(b),\ BS_2(\wedge, \wedge, \sim); r_5|$
$e, \ldots, e \# \delta\rho;\ R(\bar{b}),\ BS_2(\wedge, \sim, \wedge); r_6\rangle$
$r_5\langle e, \ldots, e \# \xi\rho;\ BS_1(\wedge, \wedge, '\rho); r_0\rangle$
$r_6\langle e, \ldots, e \# \gamma;\ BS_1('\rho, \wedge, \wedge); r_0\rangle$

§ 6.7. Многоосновные алгебры и языки программирования

Конструирование параметрических систем программирования связано с решением многих сложных проблем. К ним относится прежде всего разработка формализмов, ориентированных на описание синтаксиса и семантики языков программирования (см. § 6.3—6.6). При решении этой проблемы оказывается весьма эффективным использование аппарата многоосновных алгебр [71].

Рассмотрим класс порождающих алфавитных алгебр $\langle \mathfrak{M}, \Omega\rangle$ с выделенной основной компонентой — множеством $A_{\psi_0} \in \mathfrak{M}$, элементами которых являются цепочки в некотором терминальном алфавите $\mathfrak{A}$ (см. § 3.6). Многоосновная алгебра $\langle \mathfrak{M}, \Omega\rangle$ порождает язык $L \subseteq F(\mathfrak{A})$, если $L = A_{\psi_0}$. Для каждой кс-грамматики $G = (\mathfrak{A}, V_н, \sigma, P)$ может быть построена многоосновная алгебра $\mathfrak{M}(G) = \{A_\psi \mid \psi \in V_н\}$ с выделенной компонентой A_σ, $L(G) = A_\sigma$. Смысл такой конструкции можно проиллюстрировать на примере кс-грамматики $G' = (\mathfrak{A}', V'_н, \sigma, P')$ со схемой

$$P' = \begin{cases} p_1: \sigma \to a\sigma, \\ p_2: \sigma \to a\delta, \\ p_3: \delta \to b\delta, \\ p_4: \delta \to b, \end{cases}$$

где $\mathfrak{A}' = \{a, b\}$; $V'_н = \{\sigma, \delta\}$. Очевидно, кс-грамматика G' порождает язык $L(G') = \{a^n b^m \mid n, m = 1, 2, \ldots\}$. Рассмотрим систему множеств $\mathfrak{M}(G) = \{A_\psi \mid \psi \in V_н\}$, где A_ψ — множество всех терминальных цепочек, выводимых из нетерминала ψ в грамматике G. Для кс-грамматики G' система основных множеств имеет вид $\mathfrak{M}(G') = \{A'_\sigma, A'_\delta\}$, где $A'_\sigma = \{a^n b^m \mid n, m = 1, 2, \ldots\}$, $A'_\delta = \{b^m \mid m = 1, 2, \ldots\}$. Каждой кс-продукции $p: \psi_i \to u_{i1}\psi_1^i u_{i2}\psi_2^i \ldots u_{ir_i}\psi_{r_i}^i \cdot u_{i(r_i+1)} \in P$ поставим в соответствие конкатенарную операцию $w_p: A_{\psi_1^i} \times A_{\psi_2^i} \times \cdots \times A_{\psi_{r_i}^i} \to A_{\psi_i}$, определенную на множествах

$A_{\psi_1^i}, A_{\psi_2^i}, \ldots, A_{\psi_{r_i}^i}$ и такую, что при подстановке вместо соответствующих аргументов произвольных цепочек $x_s = A_{\psi_s^i}\ (1 \leqslant s \leqslant r_i)$ значением операции w_p является цепочка $x = u_{i1}x_1u_{i2}x_2 \ldots u_{ir_i}x_{r_i}u_{i(r_i+1)}$, принадлежащая множеству A_{ψ_i}. Конкатенарные операции, ассоциированные со схемой P' кс-грамматики G', составляют сигнатуру Ω', в которую входят унарные операции

$$w_{p_1}: A'_\sigma \to A'_\sigma,$$
$$w_{p_2}: A'_\delta \to A'_\sigma,$$
$$w_{p_3}: A'_\delta \to A'_\sigma,$$

где $w_{p_1}(\sigma) = a\sigma$, $w_{p_2}(\delta) = a\delta$, $w_{p_3}(\delta) = b\delta$, а также нульарная операция $w_{p_4} = b$ над множеством A'_δ. Построенная таким образом многоосновная алгебра $\langle \mathfrak{M}(G); \Omega\rangle$ с выделенной основной компонентой A_σ, соответствующей аксиоме грамматики G, называется **контекстно-свободной многоосновной алгеброй (кс-алгеброй)**, ассоциированной с грамматикой G. Так, с кс-грамматикой G' ассоциирована кс-алгебра $\langle \mathfrak{M}(G'); \Omega\rangle$ с выделенной основной компонентой A'_σ.

Рассмотренная конструкция подтверждает справедливость следующего утверждения.

Теорема 6.16. *Язык L является кс-языком тогда и только тогда, когда существует порождающая его контекстно-свободная алгебра.*

Заметим, что при описании языков с помощью порождающих многоосновных алгебр сохраняется возможность проведения синтаксического анализа данных языков изложенными выше методами.

Проблема синтаксического анализа допускает алгебраическую интерпретацию, связанную с введением синтаксической алгебры $\mathfrak{S} = \langle S_\psi; \Omega\rangle$, которая представляет собой свободную многоосновную алгебру (см. § 3.6), состоящую из элементов, соответствующих синтаксическому разложению языковых объектов. Синтаксическая структура языка L определяется порождающей алгеброй $\mathfrak{A} = \langle A_\psi; \Omega\rangle$, синтаксической алгеброй $\mathfrak{S} = \langle S_\psi; \Omega\rangle$ и синтаксическим гомоморфизмом $f_{sin}: \mathfrak{S} \to \mathfrak{A}$. При этом элементу $a \in A_\psi$, если $f_{sin}(s) = a$, соответствует элемент $s \in S_\psi$ — синтаксическое разложение элемента a. В математической лингвистике широко распространен способ представления синтаксической структуры языковых объектов в виде деревьев грамматического разбора (см. гл. 5). Синтаксическая структура языка L однозначна, если гомоморфизм f_{sin} индуцирует взаимно-однозначное отображение множества S_{ψ_0} на A_{ψ_0}, где S_{ψ_0} и A_{ψ_0} — выделенные компоненты алгебр соответственно $\mathfrak{S}$ и $\mathfrak{A}$, так что каждая правильная цепочка языка имеет единственное синтаксическое разложение. Если гомоморфизм f_{sin} является изоморфизмом, синтаксические структуры языковых объектов строго однозначны.

Наряду с синтаксической естественно рассматривать семантическую структуру языка, которая определяется алгеброй $\mathfrak{B} = \langle B_\psi; \Omega \rangle$ и гомоморфизмом $g_{sem}: \mathfrak{S} \to \mathfrak{B}$. При этом $\mathfrak{B}$ называется **семантической алгеброй,** а гомоморфизм g_{sem} — **семантическим гомоморфизмом,** который сопоставляет синтаксической структуре языковых объектов (элементов алгебры $\mathfrak{A}$) их семантическую структуру. Семантика элемента $a \in A_\psi$ с синтаксической структурой $s \in S_\psi$ ($f_{sin}(s) = a$) определяется элементом $b \in B_\psi$ таким, что $b = g_{sem}(s)$. Подобно синтаксису, языковые объекты в зависимости от свойств гомоморфизма g_{sem} могут иметь однозначную, строго однозначную и неоднозначную семантическую структуру. При строгой однозначности семантической структуры существует отображение $h_{gen}: \mathfrak{A} \to \mathfrak{B}$ такое, что гомоморфизм g_{sem} представим в виде композиции отображений f_{sin} и h_{gen}, т. е. $g_{sem} = f_{sin} \cdot h_{gen}$. Отображение g_{sem}, если $g_{sem}(s) = g'_{sem}(s')$, индуцирует на алгебре $\mathfrak{S}$ отношение конгруэнтности ρ, $s\rho s'$, которое в случае строгой однозначности переносится также на алгебру $\mathfrak{A}$. Отношение ρ называется **отношением семантической эквивалентности языковых объектов** и может быть использовано при эквивалентных преобразованиях программ по выбранным критериям.

В процессе трансляции с языка L на язык L' необходимо установить взаимосвязь семантических структур соответствующих языковых объектов. Рассмотрим случай, когда порождающие алгебры языков L и L' имеют одну и ту же сигнатуру операций Ω и общую семантическую алгебру $\mathfrak{B}$. Для языков L и L' справедливо отношение перевода θ, $a\theta a'$, где $a \in L$, $a' \in L'$, если существуют синтаксические разложения s и s' элементов соответственно a и a' с одной и той же семантикой, $g_{sem}(s) = g'_{sem}(s')$. Алгебраическая характеристика процесса трансляции для языков L и L', которым соответствуют порождающие многоосновные алгебры с несовпадающими сигнатурами операций, приведена в работе [71].

Разновидностью многоосновных алгебр являются мультиалгебры, аппарат которых также может быть эффективно использован для решения проблемы формализации синтаксиса и семантики языков программирования. Полученные в данном направлении результаты подробно изложены в монографии [36] (см. § 6.7).

Семантика языков программирования может трактоваться, в силу его разрешимости, как семантика программ, составленных на данном языке. Это означает, что семантическое погружение языка в алгоритмические алгебры сопряжено с поиском базиса и системы образующих, ориентированных на представление его программ. Базисные операторы интерпретируются как машинные средства, в терминах которых представлены в регулярной форме основные языковые конструкции, входящие в систему образующих алгоритмических алгебр. Семантика языка, погруженного в алгоритмические алгебры, определяется как замыкание соответствующей системы образующих посредством суперпозиции входящих в нее операторных функциональных и логических структур. Подход, примененный при построении семантического погружения адресного языка в алгоритмические алгебры (см. § 4.5, а также [59]), нашел дальнейшее развитие при описании семантики языков программирования типа АЛГОЛ-60 с достаточно простой структурой данных.

ЛИТЕРАТУРА

1. Александров П. С. Введение в общую теорию множеств и функций. М., Гостехиздат, 1948. 411 с.
2. Анисимов А. В. Формальные грамматики, учитывающие внешние терминальные контексты. — Кибернетика, 1974, № 3. с. 81—88.
3. Ахоу Ульман. Теория языков. — Кибернет. сб. Новая сер., 1969, вып. 6, с. 145—184.
4. Бабенко Л. П., Ющенко Е. Л. О классификации языков программирования. — Системное и теорет. программирование. Тезисы докл. III Всесоюз. симпозиума. Т. 2. Кишинев, 1974, с. 145—152.
5. Бардзинь Я. М. Сложность распознавания симметрии на машинах Тьюринга. — Проблемы кибернетики, 1965, вып. 15, с. 245—249.
6. Биркгоф Г. Теория структур. М., Изд-во иностр. лит., 1952. 407 с.
7. Боднарчук В. Г. Системы уравнений в алгебре событий. — Журн. вычисл. математики и мат. физики, 1963, 3, № 6, с. 1077—1088.
8. Боднарчук В. Г., Цейтлин Г. Е. Об алгебрах периодически определенных преобразований бесконечного регистра. — Кибернетика, 1969, № 1, с. 18—28.
9. Бурбаки Н. Теория множеств. М., «Мир», 1965. 450 с.
10. Валуцэ И. И. Отображения. Алгебраические аспекты теории. Кишинев, «Штиинца», 1976. 139 с.
11. Васильев В А. Язык АЛГОЛ-68. Основные понятия. М., «Наука», 1972. 128 с.
12. Вельбицкий И. В. Технология производства программ на базе *R*-метаязыков. — Системное и теорет. программирование. Тезисы докл. III Всесоюз. симпозиума. Т. 1. Кишинев, 1974, с. 260—271.
13. Вельбицкий И. В., Мейтус В. Ю., Ющенко Е. Л. *M*-формализмы и их применения к операционным системам. — Труды Всесоюз. симпозиума по теории языков и методам построения систем программирования. К., 1972, с. 22—30.
14. Вельбицкий И. В., Мейтус В. Ю., Ющенко Е. Л. Теория *M*-систем и ее приложения. — Проблемы кибернетики, 1973, вып. 27, с. 251—266.
15. Вельбицкий И. В., Ющенко Е. Л. Метаязык, ориентированный на синтаксический анализ и контроль. — Кибернетика, 1970, № 2, с. 50—54.
16. Виленкин С. Я., Трахтенгерц Э. А. Математическое обеспечение управляющих вычислительных машин., М., «Энергия», 1972. 392 с.
17. Виноградов И. М. Основы теории чисел. М., «Наука», 1965. 172 с.
18. Вычислительные машины с развитыми системами интерпретации. К., «Наук. думка», 1970. 260 с. Авт.: В. М. Глушков, А. А. Барабанов, Л. А. Калиниченко, С. Д. Михновский, З. Л. Рабинович.
19. Гинзбург С. Математическая теория контекстно-свободных языков. М., «Мир», 1970. 328 с.
20. Гладкий А. В. Лекции по математической лингвистике. Для студентов НГУ. Новосибирск, Изд-во Новосиб. ун-та, 1966. 189 с.

21. Гладкий А. В. Формальные грамматики и языки. М., «Наука», 1973. 368 с.
22. Гладкий А. В., Мельчук И. А. Элементы математической лингвистики. М., «Наука», 1969. 192 с.
23. Глушков В. М. Абстрактная теория автоматов. — Успехи мат. наук, 1961, 16, вып. 5, с. 3—62.
24. Глушков В. М. Синтез цифровых автоматов. М., Физматгиз, 1962. 476 с.
25. Глушков В. М. Введение в кибернетику. К., Изд-во АН УССР, 1964. 324 с.
26. Глушков В. М. Теория автоматов и формальные преобразования микропрограмм. — Кибернетика, 1965, № 5, с. 1—10.
27. Глушков В. М. К вопросу о минимизации программ и схем алгоритмов. — Кибернетика, 1966, № 5, с. 1—4.
28. Глушков В. М. О простых алгоритмах анализа и синтеза магазинных автоматов. — Кибернетика, 1968, № 5, с. 1—10.
29. Глушков В. М., Боднарчук В. Г., Гринченко Т. А., Дородницына А. А., Клименко В. П., Летичевский А. А., Погребинский С. Б., Стогний А. А., Фишман Ю. С. АНАЛИТИК (алгоритмический язык для описания вычислительных процессов с использованием аналитических преобразований). — Кибернетика, 1971, № 3, с. 102—134.
30. Глушков В. М., Вельбицкий И. В., Стогний А. А. Об одном подходе к построению системного математического обеспечения современных вычислительных машин. — Кибернетика. 1972, № 3, с. 25—36.
31. Глушков В. М., Капитонова Ю. В., Летичевский А. А. О методике проектирования вычислительных машин в системе ПРОЕКТ. — Кибернетика, 1971, № 2, с. 1—17.
32. Глушков В. М., Капитонова Ю. В., Летичевский А. А. Автоматизация проектирования вычислительных машин. К., «Наук. думка», 1975. 231 с.
33. Глушков В. М., Капитонова Ю. В., Летичевский А. А. Теория структур данных и параллельные синхронные вычисления. — Кибернетика, 1976, № 6, с. 2—15.
34. Глушков В. М., Летичевский А. А. Теория автоматов и программирование. — В кн.: Первая Всесоюз. конф. по программированию. Пленарные докл. К., 1968, с. 3—19.
35. Глушков В. М., Летичевский А. А. Теория дискретных преобразователей. — В кн.: Избранные вопросы алгебры и логики. Новосибирск, 1973, с. 5—40.
36. Глушков В. М., Цейтлин Г. Е., Ющенко Е. Л. Алгебра. Языки. Программирование. К., «Наук. думка», 1974. 328 с.
37. Глушков В. М., Цейтлин Г. Е., Ющенко Е. Л. Теория автоматов и некоторые вопросы синтеза структур языковых процессоров. — Кибернетика, 1975, № 5, с. 1—21.
38. Гнеденко Б. В., Королюк В. С., Ющенко Е. Л. Элементы программирования. М., «Наука», 1964. 348 с.
39. Гонца М. Г., Марчук М. Н. К проблеме синтаксического анализа языков с помощью рекурсивно-полных грамматик. — Прикл. математика и программирование, 1973, вып. 10, с. 61—67.
40. Горинович Л. Н., Трахтенгерц Э. А. Особенности математического обеспечения многопроцессорной вычислительной системы с перенастраиваемой структурой. — Системное и теорет. программирование. Тезисы докл. III Всесоюз. симпозиума. Т. 2. Кишинев, 1974, с. 153—163.
41. Дал У., Дейкстра Э., Хоор К. Структурное программирование. М., «Мир», 1975. 248 с.

42. Дал У., Мюрхауг Б., Нюгорд К. Симула-67. Универсальный язык программирования. М., «Мир», 1969. 99 с.
43. Евреинов Э. В. Теоретические основы построения универсальных вычислительных сред. — Вычисл. системы, 1965, вып. 16, с. 3—72.
44. Евреинов Э. В., Косарев Ю. Г. Однородные универсальные вычислительные системы высокой производительности. Новосибирск, «Наука», 1966. 308 с.
45. Ершов А. П. Об операторных схемах Янова. — Проблемы кибернетики, 1967, вып. 20, с. 181—201.
46. Ершов А. П. Теория программирования и вычислительные системы. М., «Знание», 1972. 64 с.
47. Ершов А. П. Современное состояние теории схем программ. — Проблемы кибернетики, 1974, вып. 27, с. 87—111.
48. Ершов А. П., Ляпунов А. А. О формализации понятия программы. — Кибернетика, 1967, № 5, с. 40—57.
49. Журавлев Ю. И. Теоретико-множественные методы алгебры логики. — Проблемы кибернетики, 1962, вып. 8, с. 5—44.
50. Иванов П. М. Аксиоматизация микропрограммных алгебр. — Теорет. кибернетика, 1970, вып. 5, с. 9—48.
51. Ингерман П. Синтаксически ориентированный транслятор. М., «Мир», 1969. 176 с.
52. Камынин С. С., Любимский Э. З. Алгоритмический машинно-ориентированный язык АЛМО. — Алгоритмы и алгоритмич. языки, 1967, вып. 1, с. 1—64.
53. Капитонова Ю. В. Дискретные системы и задачи их реализации. — Кибернетика, 1975, № 4, с. 7—11; № 5, с. 21—27.
54. Карри Х. Основания математической логики. М., «Мир», 1969. 576 с.
55. Кекелия В. И., Кирсанов Г. М., Цейтлин Г. Е. Реализация средств алгоритмических алгебр в однородных структурах. — Кибернетика, 1974, № 5, с. 29—36.
56. Кекелия В. И., Цейтлин Г. Е. К реализации многорегистровых периодически определенных преобразований в одной абстрактной модели вычислительной среды. — Вычисл. системы, 1971, вып. 47, с. 87—102.
57. Кирсанов Г. М., Цейтлин Г. Е. К вопросу автоматизации формальных преобразований в системе алгоритмических алгебр. — В кн.: Вопросы обучения языкам программирования. К., 1974, с. 3—18.
58. Кирсанов Г. М., Цейтлин Г. Е. Некоторые вопросы полноты аксиоматической системы в алгоритмических алгебрах. — Кибернетика, 1977, № 4, с. 35—37.
59. Кирсанов Г. М., Цейтлин Г. Е., Ющенко Е. Л. О применении микропрограммных алгебр к некоторым вопросам автоматизации программирования. — В кн.: Математическое обеспечение ЭЦВМ. К., 1971, с. 99—117.
60. Клини С. К. Введение в математику. М., Изд-во иностр. лит., 1957. 526 с.
61. Кон П. Универсальная алгебра. М., «Мир», 1968. 341 с.
62. Косарев Ю. Г., Миренков Н. Н. Математическое обеспечение однородных вычислительных систем. — Вычислительные системы, 1974, вып. 58, с. 61—79.
63. Котов В. Е. Теория параллельного программирования. Прикладные аспекты. — Кибернетика, 1974, № 1, с. 1—16; № 2, с. 1—18.
64. Коэн П. Дж. Теория множеств в континуум-гипотеза. М., «Мир», 1969. 345 с.
65. Криницкий Н. А. Равносильные преобразования алгоритмов и программирование. М., «Сов. радио», 1970. 304 с.
66. Курош А. Г. Лекции по общей алгебре. М., Физматгиз, 1962. 396 с.

67. Курош А. Г. Теория групп. М., «Наука», 1967. 648 с.
68. Лавров С. С. Введение в программирование. М., «Наука», 1973. 351 с.
69. Лазарев В. Г. Принципы реализации и особенности синтеза автоматов в базисе однородных сред. — Труды Междунар. симпозиума ИФАК. Дискретные системы. Т. 1. Рига, 1974, с. 26—36.
70. Летичевский А. А. Представление контекстно-свободных языков в автоматах с памятью типа push—down. — Кибернетика, 1965, № 2, с. 80—85.
71. Летичевский А. А. Синтаксис и семантика формальных языков. — Кибернетика, 1968, № 4, с. 1—10.
72. Летичевский А. А. Функциональная эквивалентность дискретных преобразователей. — Кибернетика, 1969, № 2, с. 15—17; 1970, № 2, с. 14—29; 1972, № 1, с. 1—5.
73. Логика. Автоматы. Алгоритмы. М., Физматгиз, 1963. 556 с. Авт.: М. А. Айзерман, Л. А. Гусев, Л. И. Розоноэр, И. М. Смирнова, А. А. Таль.
74. Ляпин Е. С. Полугруппы. М., Физматгиз, 1960. 592 с.
75. Ляпунов А. А. К алгебраической трактовке программирования. — Проблемы кибернетики, 1962, вып. 8, с. 235—243.
76. Мальцев А. И. Алгоритмы и рекурсивные функции. М., «Наука» 1965. 391 с.
77. Мальцев А. И. Алгебраические системы. М., «Наука», 1970. 391 с.
78. Мальцев А. И. Итеративные алгебры и многообразия Поста. — В кн.: Избранные труды А. И. Мальцева. Т. 2. М., 1976, с. 316—330.
79. Манакова Е. И. К проблеме синтаксического анализа языков слева направо. — В кн.: Математическое обеспечение ЭЦВМ. К., 1971, с. 43—54.
80. Марков А. А. Теория алгоритмов. М., Изд-во АН СССР, 1954. 374 с. (Труды Мат. ин-та АН СССР, 42).
81. Микроэлектроника и однородные структуры для построения логических и вычислительных устройств. М., «Наука», 1967. 227 с. Авт.: И. В. Прангишвили, Н. В. Абрамова, Е. В. Бабичева, В. В. Игнатущенко.
82. Миренков Н. Н. Структурное параллельное программирование. — Программирование, 1975, № 3, с. 3—14.
83. Мищенко В. В., Цейтлин Г. Е., Шаповалова Н. Н. К вопросам автоматного взаимодействия при реализации некоторых системных процессов. — В кн.: Вопросы обучения языкам программирования. К., 1974, с. 18—40.
84. Нариньяни А. С. Теория параллельного программирования. Формальные модели. — Кибернетика, 1974, № 3, с. 1—16; № 5, с. 1—14.
85. Наур П. Алгоритмический язык АЛГОЛ-60. Пересмотренное сообщение. М., «Мир», 1965. 76 с.
86. Новиков П. С. Элементы математической логики. М., Физматгиз, 1959. 400 с.
87. Операционная система IBM/360. Супервизор и управление данными. М., «Сов. радио», 1973. 312 с.
88. Параметрические транслирующие системы. Ч. 1—2. Кишинев, «Штиинца», 1974. Авт.: М. Г. Гонца, М. Н. Маричук, Г. А. Бостан, Г. А. Магарцу.
89. Перевозчикова О. Л., Цейтлин Г. Е., Шевченко В. В. О моделях грамматик, ориентированных на двусторонний синтаксический анализ языков. — Кибернетика, 1976, № 3, с. 1—11.
90. Перевозчикова О. Л., Цейтлин Г. Е., Шевченко В. В., Ющенко Е. Л. Стратегия двустороннего синтаксического анализа языков и бс-автоматы. — Системное и теорет. программирование. Тезисы докл. III Всесоюз. симпозиума. Т. 1. Кишинев, 1974, с. 249—250.

91. Подловченко Р. И. *R*-схемы и отношения эквивалентности между ними. — Проблемы кибернетики, 1973, вып. 27, с. 213—237.
92. Поспелов Д. А. Введение в теорию вычислительных систем. М., «Сов. радио», 1972. 280 с.
93. Пронина В. А., Трахтенгерц Э. А. Параллельный синтаксический анализатор для многопроцессорной вычислительной системы с перестраиваемой структурой. — Системное и теорет. программирование. Тезисы докл. III Всесоюз. симпозиума. Т. 2. Кишинев, 1974, с. 208—214.
94. Редько В. Н. Об определяющей совокупности соотношений алгебры регулярных событий. — Укр. мат. журн., 1964, 16, № 1, с. 120—126.
95. Редько В. Н. Некоторые вопросы теории языков. — Кибернетика, 1965, № 4, с. 12—22.
96. Редько В. Н. Параметрические грамматики и проблема параметризации. — Автоматизация программирования, 1969, вып. 1, с. 27—36.
97. Редько В. Н. Интерпретированные языки и интерпретаторы. — Кибернетика, 1969, № 5, с. 81—90.
98. Редько В. Н., Ющенко Е. Л. Алгоритмические языки и транслирующие системы. — Кибернетика, 1967, № 5, с. 87—91.
99. Саломаа А. Аксиоматизация алгебры событий, реализуемых логическими сетями. — Проблемы кибернетики, 1966, вып. 17, с. 237—246.
100. Трахтенброт Б. А., Бардзинь Я. М. Конечные автоматы. Поведение и синтез. М., «Наука», 1970. 400 с.
101. Трахтенгерц Э. А., Шумей А. С. Синтаксический анализ языков, порождаемых однозначными грамматиками предшествования. — Журн. вычисл. математики и мат. физ., 1971, 11, № 4, с. 1005—1013.
102. Универсальный язык программирования PL/1. М., «Мир», 1968. 352 с.
103. Управляющая машина широкого назначения «Дніпро» и программирующая программа к ней. К., «Наук. думка», 1964. 280 с. Авт.: Е. Л. Ющенко, Б. Н. Малиновский, Г. А. Полищук, Э. К. Ядренко, А. И. Никитин.
104. Фельдман Дж., Грайс Д. Системы построения трансляторов. — Алгоритмы и алгоритмич. языки, 1971, вып. 5, с. 105—184.
105. Фишер П., Мейер А., Розенберг А. Счетчиковые машины и счетчиковые языки. — В кн.: Проблемы математической логики. М., 1970, с. 380—401.
106. Флорес А. Программное обеспечение. М., «Мир», 1971. 519 с.
107. Флорес А. Организация вычислительных машин. М., «Мир», 1972. 428 с.
108. Хаусдорф Ф. Теория множеств. М.—Л., ОНТИ, 1937. 304 с.
109. Хомский Н. О некоторых формальных свойствах грамматик. — Кибернет. сб., 1962, вып. 5, с. 279—312.
110. Хомский Н. Формальные свойства грамматик. — Кибернет. сб. Новая сер., 1966, вып. 2, с. 121—230.
111. Цейтлин Г. Е. Вопросы полноты в множестве *n*-отношений. — Теория автоматов, 1967, вып. 3, с. 16—29.
112. Цейтлин Г. Е. Элементы теории булевых функций. К., «Техніка», 1967. 75 с.
113. Цейтлин Г. Е. Вопросы функциональной полноты для одной модификации алгебры логики. — Кибернетика, 1969, № 4, с. 39—45.
114. Цейтлин Г. Е. Структура модифицированной алгебры Поста и ее свойства. — Кибернетика, 1970, № 4, с. 39—55.
115. Цейтлин Г. Е. О бесконечно порожденных подалгебрах модифицированной алгебры Поста. — Кибернетика, 1971, № 2, с. 43—56.
116. Цейтлин Г. Е. Канонические представления логических условий в системе алгоритмических алгебр. — В кн.: Языки программирования и методы их реализации. К., 1973, с, 22—34.
117. Цейтлин Г. Е. Формальные преобразования в алгоритмической алгебре условий. Препринт Ин-та кибернет., 73-25. К., 1973. 28 с.
118. Цейтлин Г. Е. Некоторые структурные проблемы в алгебрах контекстно-свободных языков. — Проблемы кибернетики, 1974, вып. 28, с. 269—275.
119. Цейтлин Г. Е. О критериях бесконечной порожденности в универсальных алгебрах. — Кибернетика, 1974, № 3, с. 46—52.
120. Цейтлин Г. Е. Однородные структуры и модифицированные алгебры Поста. — Труды Междунар. симпозиума ИФАК. Дискретные системы. Т. 3. Рига, с. 238—247.
121. Цейтлин Г. Е. Проблема тождественных преобразований схем структурированных программ с замкнутыми логическими условиями. Ч. 1. — Кибернетика, 1978, № 3, с. 50—57.
122. Цейтлин Г. Е. Система алгоритмических алгебр и некоторые схемы управления в однородных структурах. — В кн.: Вопросы теории и построения вычислительных систем (Вычислительные системы). Новосиб., 1977, с. 29—40.
123. Цейтлин Г. Е., Ющенко Е. Л. О представлении языков в бс-автоматах. — Кибернетика, 1974, № 6, с. 40—52.
124. Цейтлин Г. Е., Ющенко Е. Л. Некоторые вопросы теории параметрических моделей языков и параллельный синтаксический анализ. — Труды Всесоюз. симпозиума по методам реализации новых алгоритмич. языков. Ч. 2. Новосибирск, 1975, с. 61—73.
125. Цейтлин Г. Е., Ющенко Е. Л. О параметрических моделях языков, ориентированных на однозначный ПР-анализ. — Кибернетика, 1977, № 1, с. 21—35.
126. Шевченко В. В. Об одном подходе к проблеме синтаксического анализа. — Кибернетика, 1974, № 4, с. 30—39.
127. Шрейдер Ю. А. Равенство, сходство, порядок. М., «Наука», 1971. 256 с.
128. Ющенко Е. Л. Адресное программирование. К., Гостехиздат УССР, 1963. 288 с.
129. Ющенко Е. Л., Лаврищева Е. М. Метод анализа программ на базе СМ-грамматик. — Кибернетика, 1972, № 2, с. 41—45.
130. Ющенко Е. Л., Перевозчикова О. Л. Развитие языков программирования и диалоговых систем в СССР. — Кибернетика, 1976, № 6, с. 16—33.
131. Ющенко Е. Л., Цейтлин Г. Е. Об алгебре многорегистровых операторов. — Кибернетика, 1971, № 2, с. 66—71.
132. Ющенко Е. Л., Цейтлин Г. Е. О СМ-грамматиках контекстно-свободных языков. — В кн.: Системное программирование. Ч. 2. Новосибирск, 1973, с. 155—160.
133. Яблонский С. В. Функциональные построения в *k*-значной логике. — Труды Мат. ин-та АН СССР. 1958, 51, с. 5—142.
134. Яблонский С. В., Гаврилов Г. П., Кудрявцев Б. В. Функции алгебры логики и классы Поста. М., «Наука», 1966. 119 с.
135. Янгер Д. Х. Распознавание и анализ контекстно-свободных языков за время n^3. — В кн.: Проблемы математической логики. М., 1970. с. 344—363.
136. Янов Ю. И. О логических схемах алгоритмов. — Проблемы кибернетики, 1958, вып. 1, с. 75—127.
137. Янов Ю. И. Об инвариантных операциях над событиями. — Проблемы кибернетики, 1964, вып. 12, с. 253—258.

138. Янов Ю. И. О некоторых подалгебрах событий, не имеющих конечных полных систем тождеств. — Проблемы кибернетики, 1966, вып. 17, с. 255—259.
139. Янов Ю. И., Мучник А. А. О существовании k-значных замкнутых классов, не имеющих конечного базиса. — Докл. АН СССР, 1959, **127**, № 1, с. 44—46.
140. Brooker R., Morris D. An assembly program for a phrase structure language. — Comput. J., 1960, **3**, № 3, p. 168—174.
141. Brooker R., Morris D., McCallum J., Roh J. The compiler compiler. — Annu. Rev. Automat. Program., 1963, № 3, p. 229.
142. Čulic Karel. II Contribution to deterministic top-down analysis of context-free languages. — Kybernetika, 1968, 4, № 5, p. 422—431.
143. Floyd R. W. Syntactic analysis and operator precendence. — J. Assoc. Comput. Mach., 1963, **10**, № 3, p. 316—333.
144. Floyd R. W. Bounded context syntactic analysis. — Communs ACM, 1964, 7, № 2, p. 62—67.
145. Glushkov V. M., Ignatiev M. B., Mjasnikov V. A., Torgashev V. A. Recursive machines and compnting technology. — In: IFIP Congress-74. Stockholm, 1974, p. 65—70.
146. Glushkov V. M., Tseitlin G. E., Jushchenko E. L. The theory of language processors and homogeneous structures. — In: II International Symposium on Programming. Paris, 1976, p. 201—218.
147. Greibach S. A. A new normal form theorem for context-free phrase structure grammars. — J. Assoc. Comput. Mach., 1965, **12**, № 1, p. 42—52.
148. Hennie F. S. Iterative arrays of logical circuits. New York — London, John Willey, Sons, Ins., 1961. 242 p.
149. Irons E. T. A syntax directed compiler for ALGOL-60. — Communs ACM, 1961, 4, № 1, p. 51—55.
150. Knuth D. E. On the translation of languages from left to right. — Inform. and Contr., 1965, 8, № 6, p. 607—639.
151. Kurki-Suonio R. Notes on top-down languages. — BIT, 1969, 9, № 4, p. 225—238.
152. Lewis P. M., Stearns R. E. Syntax-directed transduction. — J. Assoc. Comput. Mach., 1968, **15**, № 3, p. 464—488.
153. Małuszyński J. Contribution to the bottom-up recognition. — Bull. Acad. pol. sci. Ser. sci. math., astron. et phys., **20**, № 9, p. 785—788.
154. Minsky M. H. Recursive unsolvability of Post's problem of «Tag» and other topics in the theory of Turing machines. — Ann. Math., 1961, 74, № 2, p. 437—455.
155. Post E. L. The two-valued iterative systems of mathematical logic.— Ann. Math. Stud., 1941, № 5, p. 2—337.
156. Rosenberg I. G. Completeness, closed classes and relations in multiple-valued logics. — Montréal, Centre de rech. math. Univ. de Montréal, 1974. 26 p.
157. Rosenkrantz D. J., Stearns R. E. Properties of deterministic top-down grammars. — Inform. and Contr., 1970, 17, № 3, p. 226—256.
158. Tseytlin G. E. The theory of the modified Post algebras and multidimentional automata structnres. — Lect. Notes Comput. Sci., 1975, № 32, p. 418—424.
159. Wirth N., Weber H. EULER: a generalization of ALGOL and its formal definition. P. 1, 2. — Communs ACM, 1966, 9, № 1, p. 13—25; № 2, p. 89—99.
160. Wood D. The theory of left-factored languages. P. 1, 2. — Comput. J., 1969, **12**, № 4, p. 349—356; 1970, **13**, № 1, p. 55—62.

Теоремы

Леммы

[1] В указателе приведены номера теорем и лемм и соответствующие им страницы.