М. Ф. Бондаренко
Н. В. Белоус
А. Г. Руткас

компьютерная ДИСКРЕТНАЯ МАТЕМАТИКА

Рекомендовано
Министерством образования и науки Украины
как учебник для студентов
технических специальностей
высших учебных заведений по направлению
«Компьютерные науки»

«Компания СМИТ»
Харьков
2004

УДК 519.1
Б–81

Рекомендовано Министерством образования и науки Украины как учебник для студентов технических специальностей высших учебных заведений по направлению «Компьютерные науки» (Письмо №14/18.2-1530 от 15.07.2002 г.)

Рецензенты: Варбанец П. Д. — д.физ.-мат.н., проф., зав. кафедрой «Компьютерная алгебра и дискретная математика», Одесский национальный университет;
Сироджа И. Б. — д.техн.н., проф., Национальный аэрокосмический университет им. Н. Е. Жуковского «ХАИ», г. Харьков

Бондаренко М. Ф., Белоус Н. В., Руткас А. Г.

Б–81 Дискретная математика.— Харьков: «Компания СМИТ», 2004.— 480 с.

ISBN 966–8530–10–1

В учебнике изложены основные разделы дискретной математики — теория множеств, теория отношений, математическая логика, алгебраические структуры, автоматы, алгоритмы, формальные языки и грамматики, теория графов и комбинаторика.

Теоретический материал проиллюстрирован примерами из разных областей знаний. Приведено большое количество упражнений и задач для приобретения практических навыков.

Учебник предназначен для студентов различных специальностей, изучающих дискретную математику, аспирантов и специалистов, использующих соответствующие математические и компьютерные методы.

ISBN 966–8530–10–1

© Бондаренко М. Ф., Белоус Н. В., Руткас А. Г.
© «Компания СМИТ», 2004

ВВЕДЕНИЕ

Основные способы представления информации являются дискретными: это слова и конструкции языков и грамматик — естественных и формализованных; табличные массивы реальных данных в технических системах и научно-естественных наблюдениях; данные хозяйственной, социальной, демографической, исторической статистики и т.п.

Для количественного анализа и вычислительных преобразований непрерывных процессов приходится их «дискретизировать». Понятно, что математические методы обработки, анализа и преобразования дискретной информации необходимы во всех областях научной, хозяйственной и социальной сфер. Обычно эти методы излагаются в курсах дискретной математики; иногда употребляется термин «конечная математика», или даже «конкретная математика».

Часто для анализа реальных систем с непрерывными конструктивными элементами строятся модели конечной или дискретной математики. Например, классическая транспортная или информационная сеть трактуется как граф с заданными пропускными способностями или весами ветвей, а геометрическая форма ветви между двумя пунктами-узлами сети не играет роли. Более того, «непрерывное» строение реальной ветви также не работает в сетевой модели: важно, что между двумя узлами a, b сети либо нет ветви, либо есть ветвь с заданным ограничением $c(a, b)$ объема переноса вещества или информации. В модели достаточно задать числа $c(a, b)$ для каждой пары узлов a, b. Если ветви нет, то $c(a, b) = 0$. Такая числовая модель отображения сети идеально приспособлена для записи, хранения и преобразований в компьютере.

Учебник рассчитан, в первую очередь, на студентов, а также читателей, желающих изучать математические методы для использования их в естественных науках и компьютерных технологиях. В нем отражены все основные разделы нормативного курса «дискретная математика» по специальностям компьютерных, физико-математических и инженерно-технических направлений. От читателя требуется знание математики в объеме средней школы, и вся дальнейшая «математическая техника» приобретается по мере работы

с книгой и предлагаемыми в ней упражнениями. Авторы надеются, что некоторые разделы книги могут быть полезными также преподавателям и работающим специалистам, особенно при использовании рекомендуемой дополнительной литературы.

Структура книги, перечень излагаемых разделов и их наполнение легко усматриваются из содержания. Для лучшего усвоения материала в тексте приводится значительное количество иллюстрированных примеров, а в конце разделов — вопросы и задания. В главах по теории графов и комбинаторике часть информации сообщается в форме справок и задач. Список литературы содержит источники двух типов: общие учебники или монографии по дискретной математике и книги по отдельным разделам — логике, языкам, теории графов, комбинаторике и др. Для первичного изучения весь список литературы является избыточным, однако может оказаться полезным при углубленном изучении отдельных разделов.

Эпитет «компьютерная» в названии употреблен по двум причинам. Во-первых, постановки и математические модели излагаемых задач — теоретических и прикладных — предполагают использование компьютера: для символьных преобразований, для численной реализации вычислительных алгоритмов и т.д. Как правило, реальные образцы этих задач обычно имеют такую размерность и трудоемкость, что осуществить их анализ и эффективное численное решение без использования компьютера невозможно. Во-вторых, отбор и изложение разделов дискретной математики в книге выполнены с учетом требований фундаментального образования по компьютерным наукам, информационным технологиям, современным инженерным и социально-экономическим направлениям с высоким уровнем автоматизации и компьютеризации.

СПИСОК ОБОЗНАЧЕНИЙ

Множества

N — множество натуральных чисел.
Z — множество целых чисел.
$N_ч$ — множество четных чисел.
Q_+ — множество положительных рациональных чисел.
R_+ — множество положительных чисел.
Q — множество рациональных чисел.
R — множество действительных чисел.
$|M|$ — мощность множества M.
$A \subseteq B$ — нестрогое включение.
$A \subset B$ — строгое включение.
$x \in A$ — элемент x принадлежит множеству A.
$x \notin A$ — элемент x не принадлежит множеству A.
U — универсальное множество.
$\varnothing$ — пустое множество.
2^X — множество-степень (множество всех подмножеств множества X) или булеан.
$A \cup B$ — объединение множеств A и B.
$A \cap B$ — пересечение множеств A и B.
$\overline{A}$ — дополнение множества A.
$A \backslash B$ — разность множеств A и B.
cl — класс множеств, называемых алгеброй.
c — континуум.

Отношения

$A \times B$ — декартово произведение множеств A и B.
X^n — декартова степень множества X.
R^{-1} — обратное отношение к отношению R.
$S \circ R$ — композиция отношений R и S.
R^n — степень отношения R.
$R(x)$ — сечение отношения R по элементу x.
$R(Z)$ — сечение отношения R по подмножеству Z.
Y/R — фактор-множество множества Y по отношению R.
D_R — область определения отношения R.
$\Re_R$ — область значений отношения R.
F — функциональное отношение.
$f: X \to Y$ — отображение.
$\sim$ — отношение эквивалентности.
$\leq$ — отношение частичного порядка.
$<$ — отношение строгого порядка.

Реляционная алгебра

σ — операция ограничения отношения.
π — операция проекции отношения.
$|\triangleright\triangleleft|$ — операция естественного соединения отношений.
$\div$ — операция деления отношений.

Алгебраические структуры

e — единичный элемент.
x' — обратный элемент к элементу x.

1 — единичный элемент относительно умножения.
x^{-1} — обратный элемент к элементу x относительно умножения.
0 — единичный элемент относительно сложения.
$-x$ — обратный элемент к элементу x относительно сложения.
$\oplus_n$ — сложение по модулю n.
$\otimes_n$ — умножение по модулю n.
$\wedge$ — операция определения нижней грани для двух элементов в решетке.
$\vee$ — операция определения верхней грани для двух элементов в решетке.
Z_+ — множество целых неотрицательных чисел.
$\varphi: A \to C$ — гомоформизм из структуры $(A, \otimes)$ в структуру $(C, \oplus)$.

Булевы функции и преобразования

$(x_1, x_2, ..., x_n)$ — двоичное слово, булевый набор или интерпретация булевой функции f.
$\wedge$ — конъюнкция.
$\vee$ — дизъюнкция.
$\bar{}$ — отрицание.
$\sim$ — эквиваленция.
$\to$ — импликация.
$(B, \wedge, \vee, \bar{})$ — алгебраическая структура — булева алгебра.
$B(\wedge, \vee, \bar{}, \to, \sim)$ — алгебра логики.
$f^*(x_1, x_2, ..., x_n)$ — функция, двойственная функции $f(x_1, x_2, ..., x_n)$.
$\oplus$ — сумма по модулю 2, исключающее «или».
$(B, \wedge, \oplus, 0, 1)$ — алгебра Жегалкина.
T_0 — класс функций, сохраняющих ноль.
T_1 — класс функций, сохраняющих единицу.
S — класс самодвойственных функций.
L — класс линейных функций.
M — класс монотонных функций.

Математическая логика

$(\{Л, И\}, \wedge, \vee, \bar{}, \to, \sim, Л, И)$ — логика высказываний.
$\dfrac{A_1, ..., A_n}{B}$ — B следствие из посылок $A_1, ..., A_n$.
D — предметная область предиката.
$(\forall x)$ — квантор всеобщности.
$(\exists x)$ — квантор существования.
$\neg x$ — циклическое отрицание.
N_x — отрицание Лукашевича.
$I^\sigma(x)$ — обобщенное отрицание.
$J^\sigma(x)$ — характеристическая функция.
$\min(x_i, x_j)$ — обобщение конъюнкции.
$x_i x_j (\mathrm{mod}\ k)$ — обобщение конъюнкции.
$\max(x_i, x_j)$ — обобщение дизъюнкции.

Теория графов

$G = (X, Y, f)$ — абстрактный граф.
X — множество вершин графа.
Y — множество ребер графа.
$\delta(x_i) = \deg x_i$ — степень вершины x_i.

$d(G)$ — диаметр графа.
$r(G)$ — радиус графа.
ρ — бинарное отношение.
$\delta^+(x)$ — число дуг, начинающихся в вершине x.
$\delta^-(x)$ — число дуг, заканчивающихся в вершине x.
A — матрица смежности.
$\underline{B}$ — матрица инциденций.
G — двойственный граф.
$\gamma(G)$ — хроматическое число или индекс.
T — дерево.
g_{nm} — число неизоморфных n-вершинных графов с m ребрами.
τ_n — число помеченных деревьев с n вершинами.
t_n — число свободных деревьев с n вершинами.
T_n — число корневых деревьев.
$K(G)$ — матрица Кирхгофа.
S — матрица сечений.
Q — матрица циклов.
K_n — полный граф с n вершинами.
K_{nm} — двудольный граф.
$\rho(G)$ — кликовое число графа G.
$\alpha(G)$ — число независимости графа G.

Языки и грамматики

ε — пустая строка.
A^* — множество всех строк алфавита A, включая пустую строку.
A^+ — множество всех строк алфавита A, не включая пустую строку.
$|\alpha|$ — длина строки.
a^n — n символов a подряд в строке символов.
$G(N, T, P, S)$ — грамматика.
T — множество терминальных символов.
N — множество нетерминальных символов.
S — начальный символ грамматики.
P — множество продукций грамматики.
$L(G)$ — язык, определяемый грамматикой G.
$\varphi \Rightarrow \psi$ — ψ непосредственно выводима из φ.
$\varphi \Rightarrow^* \psi$ — ψ выводима из φ за конечное число шагов.
D — дерево вывода.
$|$ — сокращенная запись продукции.

Алгоритмы

Базовые функции:
$O_n(x_1, \ldots, x_n)$ — функции тождественно равные нулю.
$V_{n,i}(x_1, \ldots, x_n)$ — функции выбора.
$\lambda(x)$ — функции следования.
Операторы построения рекурсивных функций:
S — оператор подстановки (суперпозиции).
R — оператор рекурсии.
μ — оператор минимизации.
$T(n)$ — временная сложность алгоритма.
$\Theta(g(x))$ — время работы алгоритма с порядком роста $g(x)$.

Автоматы

λ — концевой маркер на входной ленте автомата.

ε — пустая ячейка на входной ленте автомата.

$M = (S, I, O, f, g, s_0, F)$ — конечный автомат.

$M = (S, I, O, Z, f, g, s_0, z_0, F)$ — автомат с магазинной памятью.

$T = (S, I, f, s_0)$ — машина Тьюринга.

S — конечное множество состояний автомата.

I — конечное множество допустимых входных символов автомата.

O — конечное множество допустимых выходных символов автомата.

f — функция переходов автомата.

g — функция выходов автомата.

s_0 — начальное состояние управляющего устройства автомата.

F — множество заключительных состояний автомата.

Z — конечное множество допустимых символов памяти.

z_0 — начальный символ памяти.

Комбинаторика

A_n^k — количество k-размещений n-множества.

P_n — число перестановок из n элементов.

$\overline{A_n^k}$ — количество k-размещений (с повторениями) n-множества.

C_n^k — количество k-сочетаний из n элементов.

$\overline{C_n^k}$ — количество k-сочетаний с повторениями из элементов n типов.

D_n — количество смещений.

$D_{n,\,r}$ — количество перестановок из n элементов, из которых $n - r$ изменяют свое положение, а ровно r элементов не изменяют.

S_n^k — число Стирлинга второго рода — число способов размещения n разных предметов по k урнам.

B_n — числа Белла (число способов разбиения n-множества на непустые непересекающиеся подмножества).

$p_k(n)$ — количество всех разбиений числа n на k частей (натуральных слагаемых).

$p(n)$ — количество всех разбиений числа n со всевозможными значениями числа частей k.

раздел 1

Множества

1.1. Множество. Способы задания множеств

Элементы множества, способы задания множеств, конечные и бесконечные множества, упорядоченные множества, парадоксы

В повседневной жизни и практической деятельности часто приходится говорить о некоторых совокупностях различных объектов: предметов, понятий, чисел, символов и т.п. Например, совокупность деталей механизма, аксиом геометрии, чисел натурального ряда, букв алфавита. На основе интуитивных представлений о подобных совокупностях сформировалось математическое понятие *множества*. Большой вклад в теорию множеств внес Георг Кантор. Впоследствии, благодаря его исследованиям, теория множеств стала вполне определенной и обоснованной областью математики, а в настоящее время она приобрела фундаментальное значение. Теория множеств является основанием для всех разделов дискретной математики и компьютерных наук в целом, является одной из основ функционального анализа, топологии, общей алгебры, и в настоящее время ведутся глубокие исследования в самой теории множеств, связанные с основаниями математики.

Теория множеств вместе с другими разделами дискретной математики имеет множество полезных приложений в программировании. Она используется для построения систем управления базами данных, при построении и организации работы компьютерных сетей, в частности сети Интернет.

Множество является настолько общим и одновременно изначальным понятием, что его строгое определение через более простые понятия дать затруднительно. Поэтому вслед за Г. Кантором мы

принимаем интуитивное представление о ***множестве*** как о совокупности некоторых ***элементов***, вполне определенных в случае каждого конкретного множества.

Множества обозначают заглавными, а элементы множеств — строчными латинскими буквами или строчными латинскими буквами с индексами. Элементы множеств обычно заключаются в фигурные скобки. Например, запись $A = \{a, b, d, h\}$ означает, что множество A состоит из четырех элементов a, b, d, h. В общем виде утверждение, что конечное множество A состоит из n элементов, записывается так: $A = \{a_1, a_2, ..., a_n\}$. Принадлежность элемента множеству обозначается символом $\in$: $a \in A$ (читают: элемент a принадлежит множеству A). В противном случае обозначают $a \notin A$ (читают: элемент a не принадлежит множеству A).

В дальнейшем используются следующие общепринятые обозначения основных числовых множеств:

N — ***множество натуральных чисел***, $N = \{1, 2, 3, ...\}$;

Z — ***множество целых чисел***, $Z = \{..., -3, -2, -1, 0, 1, 2, 3, ...\}$;

Q — ***множество рациональных чисел***. Всякое рациональное число можно представить в виде дроби: a/b, где $a, b \in Z$, $b \neq 0$;

R — ***множество действительных чисел***. Всякое действительное число можно представить в виде бесконечной десятичной дроби $a, b_1 b_2 b_3 ... b_n ...$ целой частью $a \in Z$ и $b_k \in \{0, ..., 9\}$. Множеству действительных чисел соответствует множество точек на числовой прямой.

Элементами множеств могут быть другие множества, тогда эти элементы будут обозначаться заглавными буквами.

Пример. $A = \{D, C\}$, $D = \{a, b\}$, $C = \{c, d, e\}$. При этом $D \in A$, $C \in A$, но $a \notin A$ и $c \notin A$.

Пример. $E = \{\{1, 2\}, 3\}$. Эта запись означает, что множество E содержит два элемента: множество $\{1, 2\}$ и элемент 3.

> **Определение**
>
> Множество называется ***конечным,*** если оно содержит конечное число элементов, и ***бесконечным***, если оно содержит неограниченное число элементов.

Пример. Множество $A = \{1, 2, 3, 4, 5, 6, 7, 8, 9, 0\}$ цифр в десятичной системе счисления конечно, а множество точек окружности бесконечно.

Упорядоченным считается такое множество, в котором важны не только его элементы, но и порядок их следования в множестве.

Например, упорядоченным является множество, в котором каждый элемент имеет свой порядковый номер. Обозначают упорядоченное множество, как правило, либо круглыми, либо треугольными скобками.

$A = <1, 2, 3>$, в общем случае:

$A = <a_1, a_2, ..., a_n>$, $n \in N$;

$B = (a, b, c)$.

Пример. Рассмотрим упорядоченное множество $A = (x, y)$ — географических координат долготы x и широты y. Если долготу и широту поменять местами, то в результате можно попасть в другую точку света.

Указать порядковый номер для всякого вещественного числа на множестве R невозможно, и порядок в R задается с помощью сравнений $<$ и $\leq$. Поэтому общее определение упорядоченного множества X предполагает, что для всех пар элементов из X определено ***отношение порядка*** (см. п. 2.4)

Способы задания множеств

1. Множество можно задать простым ***перечислением элементов***

$$A = \{a_1, a_2, ..., a_n\}.$$

Пример. Множество отличников в группе обозначим O и зададим его перечислением: $O =$ {Иванов, Петров, Сидоров, Кукушкина}.

Способ задания множества перечислением его элементов не пригоден для задания бесконечных множеств, а в случае конечных множеств часто практически нереализуем. Так, невозможно перечислить множество рыб в Тихом океане, хотя их число конечно.

2. Другой способ задания множества состоит в описании элементов ***определяющим свойством***: $X = \{x \mid P(x)\}$, где $P(x)$ означает, что элемент x обладает свойством $P(x)$.

Пример. Множество N_{10} всех натуральных чисел, меньших 10, можно задать так: $N_{10} = \{x \mid x \in N,\ x < 10\}$.

Свойства элементов могут быть заданы не формально, а с помощью описания на естественном языке.

Пример. Множество S студентов группы ПМ-04-1, получающих стипендию.

Пример. В геометрии часто приходится иметь дело с множествами, заданными своими характеристическими свойствами. Так, окружность — геометрическое место точек плоскости, равноудаленных от центра данной точки этой плоскости.

3. Множество может быть задано ***рекурсивно*** указанием способа последовательного порождения его элементов.

Пример. Множество значений рекурсивной функции является рекурсивно-заданным множеством $F = \{\varphi_1, \varphi_2, \varphi_3, ...\}$, где $\varphi_i \in N$, $i = 1, 2, 3, ...$ Пусть $\varphi_1 = 1$, $\varphi_2 = 2$, а каждое последующее число зависит от двух предыдущих следующим образом:

$$\varphi_n = 3\varphi_{n-2} + \varphi_{n-1},\ n = 3, 4, ...$$

Тогда

$$\varphi_3 = 3\varphi_1 + \varphi_2 = 3 \times 1 + 2 = 5;$$
$$\varphi_4 = 3\varphi_2 + \varphi_3 = 3 \times 2 + 5 = 11 \text{ и т.д.}$$

При задании множеств могут возникать ошибки и противоречия. Множество задано корректно, если для любого элемента можно определить, принадлежит он множеству или нет.

Пример. Определение множества A как множества, содержащего любые пять натуральных чисел, не является корректным, поскольку невозможно определить точно элементы A. Множество всех простых чисел определено корректно. Для любого натурального числа можно проверить, является ли оно простым, хотя практически на это может потребоваться очень много времени.

Пример. Множество всех динозавров, живших на Земле, является множеством, заданным верно. Хотя практически невозможно определить элементы этого множества, но теоретически ясно, что если животное, когда-либо жившее на Земле, является динозавром, то оно принадлежит к этому множеству, в противном случае — нет.

Некорректность задания множества часто связана с ***противоречием*** при проверке принадлежности некоторого элемента множеству. Приведем два классических примера.

Пример. Определим множество G как множество всех множеств, которые не являются элементами самих себя. Но тогда нельзя выяснить, является ли само множество G элементом множества G. Если да, то приходим к противоречию, поскольку G содержит в качестве элементов только множества, которые не являются элементами самих себя. Если множество G не является элементом самого себя, то тогда, по определению, оно должно быть элементом множества G, что является противоречием.

Пример. Единственный парикмахер в городке N определяет множество K жителей, которых он должен брить, как совокупность всех тех жителей N, которые не бреются сами. Но тогда для самого парикмахера получается противоречие и при включении его в множество K, и при отнесении его к жителям, которые бреются сами.

Такие противоречия называются ***логическими парадоксами*** и изучаются в математической логике.

Вопросы

1. Запишите с помощью обозначений утверждение, что элемент a принадлежит множеству A, а элемент b не принадлежит множеству A.
2. Приведите примеры множеств, элементами которых являются множества.
3. Приведите примеры конечных и бесконечных множеств.
4. Какое множество называется упорядоченным?
5. Назовите известные вам способы задания множеств. В каком случае не применим тот или иной способ?
6. В каких случаях множество задано некорректно?
7. Какие противоречия (парадоксы) могут возникнуть при определении множеств? Приведите примеры.

Задания

1. Задайте перечислением элементов следующие множества:
   а) множество натуральных чисел, не больших 7;
   б) множество букв вашего имени;
   в) множество, единственным элементом которого является название вашего города;
   г) множество простых чисел между 10 и 20;
   д) множество положительных чисел, кратных 12.
2. Задайте в виде $X = \{x \mid P(x)\}$ следующие множества:
   а) множество натуральных чисел, не больших 100;
   б) множество четных положительных чисел;
   в) множество натуральных чисел, кратных 10.
3. Перечислите элементы множеств:
   а) $\{x \mid x \in N,\ 3 \le x \le 12\}$;
   б) $\{x \mid x$ — десятичная цифра$\}$.
4. Определите, элементом каких из приведенных множеств является 2:
   а) $\{x \mid x \in N,\ x > 1\}$;
   б) $\{x \mid x = y^2,\ y \in Z\}$;
   в) $\{2, \{2\}\}$.

1.2. Основные понятия теории множеств

Равенство множеств, включение множеств,
универсальное и пустое множества, степень множества

Рассмотрим понятие равенства множеств.

Определение
Два множества ***равны***, если они содержат одинаковый набор элементов. Обозначается $A = B$. Если множества не равны, это обозначается $A \neq B$. Число элементов конечного множества A обозначим через $|A|$.

Для множеств A и B с бесконечным или большим числом элементов проверка совпадения наборов всех элементов может быть практически затруднительной. Более эффективной оказывается логическая проверка ***двухстороннего включения***. А именно, $A = B$ тогда и только тогда, когда из $x \in A$ следует $x \in B$ и из $y \in B$ следует $y \in A$.

Рассмотрим пример.

Пример. Пусть заданы множества

$A = \{1, 2, 3, 4, 5\}$;

B — множество натуральных чисел от 1 до 5;

$C = \{c \mid 1 \leq c \leq 5, c \in N\}$;

$D = \{4, 1, 5, 2, 3\}$.

Эти множества содержат один набор элементов, поэтому $A = B = C = D$.

При задании множеств могут присутствовать неточности, которые необходимо устранять. Рассмотрим примеры.

Пример. Рассмотрим множество A остатков, получаемых при последовательном делении натуральных чисел $\{3, 4, 5, 6, ...\}$ на 3: $A = \{0, 1, 2, 0, 1, 2, 0, 1, 2, 0, ...\}$. Это множество содержит всего три элемента: 0, 1, 2. Поэтому его можно записать в виде $A = \{0, 1, 2\}$.

Пример. Пусть B — множество всех видов шахматных фигур, а C — множество всех шахматных фигур, участвующих в одной игре. Тогда $|B| = 6$ (пешка, ладья, слон, конь, ферзь, король), а $|C| = 32$ (16 белых и 16 черных).

Определение
Множество A, все элементы которого принадлежат множеству B, называется ***подмножеством*** множества B.

Обозначение. ***Нестрогое включение*** обозначается $A \subseteq B$, означает, что A — ***подмножество*** множества B, возможно, совпадающее с B. ***Строгое включение*** обозначается $A \subset B$ и означает, что A — подмножество множества B, не совпадающее с B. Символьное выражение $A \subset B$ читают «A включено в B».

Выполнение соотношений $A \subseteq B$ и $B \subseteq A$ возможно только при $A = B$. И обратно, $A = B$, если $A \subseteq B$ и $B \subseteq A$ одновременно. Заметим, что иногда в литературе символом $\subset$ обозначают «нестрогое» включение, допускающее и равенство множеств. В этом случае символ $\subseteq$ не используется, а строгое включение записывают двумя соотношениями $A \subset B$, $A \neq B$.

Пример. Для множества положительных чисел R_+ используется знак строгого включения относительно множества действительных чисел: $R_+ \subset R$.

Пример. Обозначим множество учеников некоторого класса через X, множество отличников в этом классе — через Y. Тогда $Y \subseteq X$, поскольку множество отличников в классе включено во множество учеников этого класса и теоретически может быть равным ему.

> **Определение**
> ***Универсальным*** называется множество, которое содержит все возможные элементы, встречающиеся в данной задаче. Универсальное множество обозначается символом $\boldsymbol{U}$.

Заметим, что универсальное множество U может быть индивидуальным для каждой отдельной задачи и определяется в ее условии.

Пример. Рассмотрим некоторую группу студентов. Пусть A — множество юношей группы, B — множество отличников. В данной задаче универсальным является множество студентов группы, а множества A и B являются его подмножествами: $A \subseteq U$, $B \subseteq U$.

> **Определение**
> ***Пустым*** называется такое множество, которое не содержит никаких элементов. Пустое множество обозначается специальным символом $\varnothing$.

Роль пустого множества $\varnothing$ аналогична роли числа нуль. Это понятие можно использовать для определения заведомо несуществующей совокупности элементов (например, множества зеленых слонов). Более существенным мотивом введения пустого множества является то, что заранее не всегда известно (или неизвестно вовсе),

существуют ли элементы, удовлетворяющие характеристическому свойству какого-то множества. Например, множество выигрышей в следующем тираже спортлото на купленные билеты может оказаться пустым. Пустое множество $\varnothing$ является подмножеством любого множества A, $\varnothing \subseteq A$. Следует помнить, что пустое множество является множеством, поэтому если некоторое множество A не содержит ни одного элемента, то $A = \varnothing$; $|A| = 0$. Запись $A = \{\varnothing\}$ означает, что A содержит один элемент — $\varnothing$, $|A| = 1$.

Таким образом, любое непустое множество A обязательно имеет, как минимум, два подмножества — пустое множество и само это множество.

Определение

Множество всех подмножеств множества X назовем ***множеством-степенью*** или ***булеан*** X и обозначим $\mathbf{2}^X$.

Пример. Пусть задано множество $A = \{a, b, c\}$. Система всех его подмножеств есть

$$2^A = \{\varnothing, \{a\}, \{b\}, \{c\}, \{a, b\}, \{b, c\}, \{a, c\}, \{a, b, c\}\},$$

так что 2^A содержит 8 элементов.

Пустое множество имеет только одно подмножество — само пустое множество, поэтому $2^{\varnothing} = \{\varnothing\}$. Для произвольного множества X из n элементов количество всех его подмножеств (т.е. $|2^X|$) равно 2^n:

$$|2^X| = 2^{|X|} = 2^n.$$

Вопросы

1. Какие множества считаются равными?
2. Могут ли два элемента одного множества быть одинаковыми?
3. Определите понятия подмножества и включения множеств.
4. Приведите примеры множеств A и B для случаев $A \subset B$ и $A \subseteq B$.
5. Чем отличается строгое включение от нестрогого? Приведите пример.
6. Как определяется равенство множеств через понятие нестрогого включения?
7. Какое множество называется универсальным?
8. Какое множество называется пустым?
9. Запишите отношения включения между универсальным множеством U, произвольным его подмножеством A и множеством $\varnothing$.
10. Как обозначается множество всех подмножеств некоторого множества? Сколько элементов оно содержит?

Задания

1. Определите, какие из следующих утверждений справедливы:
   а) $|\{\varnothing\}| = 1$; в) $|\{\{\varnothing\}\}| = 2$; д) $\{x\} \subseteq \{x\}$;
   б) $\{\{\varnothing\}\} \in \{\{\{\varnothing\}\}\}$; г) $x \in \{x\}$; е) $\{x\} \in \{x\}$; ж) $\{x\} \in \{\{x\}\}$.
2. Сколько элементов содержат следующие множества:
   а) $\{x\}$;
   б) $\{\{x\}\}$;
   в) $\{x, \{x\}\}$;
   г) $\{\{x\}, x, \{\{x, \{x\}\}\}\}$.
3. Какие из приведенных утверждений верны? Докажите.
   а) если $A \subset B$ и $B \subset C$, то $A \subset C$;
   б) если $A \subseteq B$ и $B \subseteq A$, то $A = B$;
   в) если $A \subseteq B$ и $B \subseteq C$, то $A \subseteq C$.
4. Дано множество $D = \{7, 13, 25, 34, 101, 112\}$. Какие из приведенных множеств являются подмножествами множества D?
   а) $\{1, 7, 13\}$;
   б) $\{0, 1, 12\}$;
   в) $\{25, 112, 34\}$;
   г) $\{a, b, c, n\}$;
   д) $\{7, 13, 25, 34, 101, 112\}$.
   е) $\varnothing$.
5. Определите, какие из приведенных множеств равны друг другу:
   а) $A = \{x \mid$ существует y такой, что $x = 2y, y \in N\}$;
   б) $C = \{1, 2, 3\}$;
   в) $D = \{0, 2, -2, 3, -3, 4, -4, \ldots\}$;
   г) $E = \{2x \mid x \in Z\}$.
6. Постройте 2^A для множества A, если
   а) $A = \{\{\varnothing\}\}$;
   б) $A = \{1, 2, 3, 4\}$;
   в) $A = \{$«день», «ночь»$\}$;
   г) $A = \{1, \{2, 3\}, 4\}$.
7. Сколько подмножеств содержит
   а) множество дней недели;
   б) множество месяцев года.
8. Пусть заданы множества S_{n-1} и S_n, такие, что $S_{n-1} = \{a_0, a_1, \ldots, a_{n-1}\}$, $S_n = \{a_0, a_1, \ldots, a_n\}$. Объясните, как получить из множества 2^{S-1} множество 2^S.
9. Составьте алгоритм, который в качестве входных данных получает два множества и определяет, равны ли эти множества, является ли одно из них подмножеством другого.
10. Составьте алгоритм, который в качестве входного данного получает множество и конструирует список всех возможных подмножеств данного множества.

1.3. Геометрическая интерпретация множеств

Диаграммы Венна, круги Эйлера

Для наглядного изображения соотношений между подмножествами универсального множества используются диаграммы Венна и круги Эйлера.

Построение ***диаграммы Венна*** заключается в разбиении плоскости на 2^n ячеек с помощью n фигур. Каждая фигура на диаграмме представляет отдельное множество, n — число изображаемых множеств. При этом каждая последующая фигура должна иметь одну и только одну общую область с каждой из ранее построенных фигур. Плоскость, на которой изображаются фигуры, представляет универсальное множество U. Таким образом, точки, не принадлежащие ни одной из фигур, принадлежат только U. Диаграмма Венна для двух множеств A и B выглядит следующим образом.

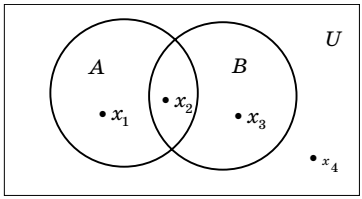


Рис. 1.1. Диаграмма Венна для двух множеств A и B

С помощью диаграмм Венна можно графически показать, принадлежит ли некоторый элемент $x \in U$ рассматриваемым множествам или нет. Например, на рис. 1.1 элемент x_1 принадлежит A и не принадлежит B, x_2 принадлежит A и B, x_3 принадлежит B и не принадлежит A, x_4 не принадлежит ни A, ни B. Любой элемент принадлежит универсальному множеству U.

Диаграмму Венна для трех множеств A, B и C представлено на рис. 1.2, где элемент x_1 принадлежит множествам A, B и C, x_2 принадлежит B и C и не принадлежит A.

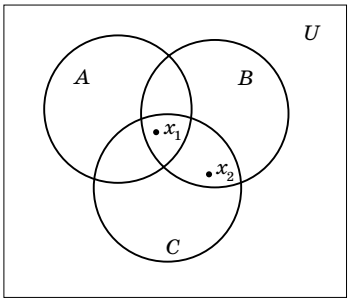
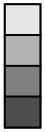


Рис. 1.2. Диаграмма Венна для трех множеств A, B и C

Диаграмму Венна для четырех множеств A, B, C и D представлено на рис. 1.3, на котором в качестве примера изображен элемент x_1, принадлежащий всем четырем множествам: A, B, C и D. Для более ясного представления заштрихуем каждую

область этой диаграммы, используя более густую штриховку там, где точки принадлежат большему числу множеств:



- принадлежит только одному из множеств;
- принадлежит только двум из множеств;
- принадлежит только трем из множеств;
- принадлежит всем четырем множествам.

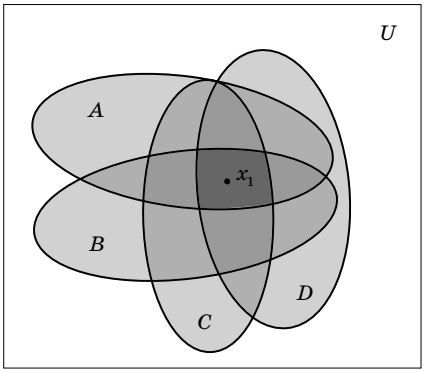
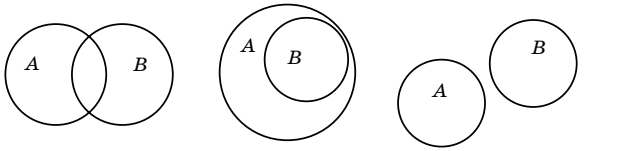


Рис. 1.3. Диаграмма Венна для четырех множеств A, B, C и D

Диаграммы Венна не отражают реальные отношения включения, установленные между множествами, а рассматривают их в общем случае. Индивидуальные отношения между заданными множествами изображают с помощью ***кругов Эйлера***. В этом случае множества, не имеющие общих элементов, изображают не пересекающимися фигурами. Отношение включения на множествах изображают, располагая одну фигуру вложенной в другую. Рассмотрим построение кругов Эйлера на примере рис 1.4.



$A = \{1, 4, 6\}$; $B = \{1, 5, 8\}$; Общий элемент — 1

$A = \{1, 4, 6\}$; $B = \{1, 6\}$; $B \subseteq A$

$A = \{1, 4, 6\}$; $C = \{3, 5, 8\}$; Нет общих элементов A и B

Рис. 1.4. Изображение множеств с помощью кругов Эйлера

Вопросы

1. В чем различия между диаграммами Венна и кругами Эйлера?
2. Попытайтесь изобразить диаграмму Венна для четырех множеств, используя окружности. Возможно ли такое изображение?
3. Является ли фигура, изображенная на рис. 1.5, диаграммой Венна для четырех множеств? Аргументируйте ответ.
4. Как выглядят круги Эйлера для множеств, которые не имеют общих элементов?
5. Как с помощью кругов Эйлера изобразить подмножества данного множества?

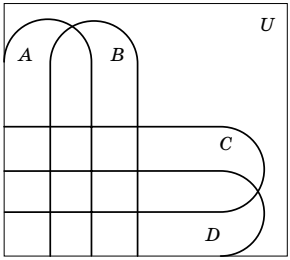


Рис. 1.5. Условие вопроса 3

Задания

1. Определите, каким множествам принадлежат элементы $x_1 \ldots x_7$, расположенные на диаграмме Венна, изображенной на рис 1.6.
2. Изобразите следующие множества в виде кругов Эйлера:
   а) $A = \{0, 1, 2\}$,
   $B = \{1, 2, 3, 4, 5\}$;
   б) $A = \{a, b, c, d, e\}$,
   $B = \{d, a, e\}$;
   в) N — натуральные числа,
   Z — целые чисел, R — действительные числа;
   г) X — множество птиц, Y — множество зверей, Z — множество млекопитающих, F — множество кроликов, G — множество живых организмов, обитающих в морях и океанах.
3. Изобразите с помощью кругов Эйлера множества A, B, C, если $A \subseteq B$, $B \subseteq C$. Покажите, что если $A \subseteq B$, $B \subseteq C$, то $A \subseteq C$.

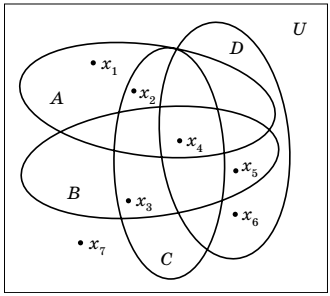


Рис. 1.6. Диаграмма Венна

1.4. Операции на множествах

Объединение, пересечение, разность, дополнение

Для наглядного представления операций будем использовать диаграммы Венна, в которых круги изображают множества, участвующие в операции, а заштрихованная часть — результат операции.

1. ***Объединение (сумма)*** $A \cup B$ есть множество, состоящее из тех и только тех элементов, которые входят либо в A, либо в B, либо в A и B одновременно (рис. 1.7).

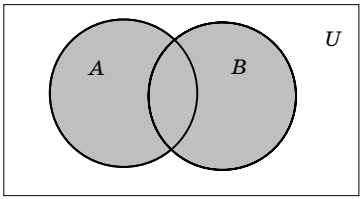


Рис. 1.7. Диаграмма Венна для $A \cup B$

Пример. Пусть даны множества $A = \{a, b, m\}$; $B = \{m, c, p\}$, тогда их объединение $A \cup B = \{a, b, c, m, p\}$.

2. ***Пересечение (произведение)*** $A \cap B$ есть множество, состоящее только лишь из элементов, входящих в A и B одновременно (рис. 1.8).

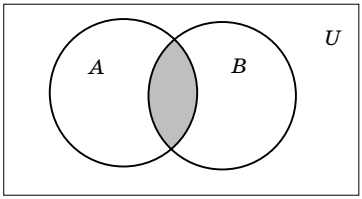


Рис. 1.8. Диаграмма Венна для $A \cap B$

Пример. Для множеств A и B из предыдущего примера $A \cap B = \{m\}$.

3. ***Разность*** $A \backslash B$ есть множество, состоящее в точности из всех элементов A, не входящих в B (рис. 1.9).

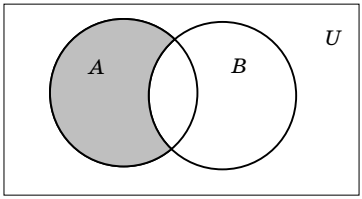


Рис. 1.9. Диаграмма Венна для $A \backslash B$

Пример. Для вышерассмотренных здесь множеств A и B

$$A \backslash B = \{a, b\}.$$

4. ***Дополнение (отрицание)*** $\overline{A}$ (читается ***«не A»***) есть множество $U \backslash A$ (рис. 1.10).

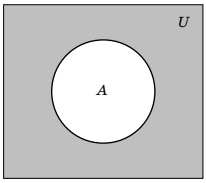


Рис. 1.10. Диаграмма Венна для $\overline{A}$

Разность множеств можно выразить через операции отрицания и пересечения следующим образом:

$$A \backslash B = A \cap \overline{B}.$$

Пример. Будем производить действия на множестве целых чисел $Z = \{..., -2, -1, 0, 1, 2, ...\}$, так что $Z_- = \{..., -2, -1, 0\}$. Дополнением к множеству Z_- является множество натуральных чисел $N = \{1, 2, ...\}$: $\overline{Z}_- = N$.

Вопросы

1. Дайте определение операциям на множествах.

2. Приведите самостоятельно примеры множеств и выполните каждую из четырех операций.
3. При выполнении какой операции используется универсальное множество?
4. Могут ли некоторые из этих операций быть выражены одна через другую? Каким образом?

Задания

1. Для множеств $A = \{1, 2, 3, 4, 5\}$, $B = \{0, 3, 6\}$ найдите:
   а) $A \cup B$; б) $A \cap B$; в) $A \backslash B$; г) $B \backslash A$.
2. С помощью диаграмм Венна докажите, что
   $$\overline{\overline{A}} = A, \quad A \backslash B = A \cap \overline{B}.$$
3. Пусть A — некоторое множество. Найдите значения выражений:
   а) $A \cup \varnothing$; в) $A \backslash \varnothing$; д) $A \cap \varnothing$; ж) $\varnothing \backslash A$;
   б) $A \cup A$; г) $A \cup U$; е) $A \cap A$; з) $A \cap U$.
4. Найдите множества A и B, если $A \backslash B = \{1, 5, 7, 8\}$, $B \backslash A = \{2, 10\}$, $A \cap B = \{3, 6, 9\}$.
5. Пусть A, B и C — множества. Покажите, что
   а) $(A \cap B) \subseteq A$; в) $(A \backslash B) \subseteq A$;
   б) $A \subseteq (A \cup B)$; г) $(A \cup B) \subseteq (A \cup B \cup C)$; д) $(A \cap B \cap C) \subseteq (A \cap B)$.
6. Изобразите с помощью диаграмм Венна пересечение и объединение трех множеств.
7. Докажите, что $A \subseteq B$ верно тогда и только тогда, когда верно $\overline{B} \subseteq \overline{A}$.
8. Какие выводы можно сделать о множествах A и B, если верно одно из следующих равенств:
   а) $A \cup B = A$;
   б) $A \cap B = A$;
   в) $A \backslash B = A$;
   г) $A \backslash B = B \backslash A$.

1.5. Алгебра множеств

Приоритет операций, тождества алгебры множеств, тождественные преобразования выражений

Множество 2^U всех подмножеств универсального множества U с заданными на нем четырьмя операциями составляют ***алгебру множеств***.

В общем случае алгебру может составлять любой класс $\mathcal{A} \subset 2^U$ подмножеств универсального множества U, замкнутый относительно всех четырех операций. Определение алгебры, не содержащее избыточных (точнее, зависимых) ограничений, выглядит следующим образом.

Определение

Класс множеств cl называется ***алгеброй*** (множеств), если:

1. $U \in cl$.
2. Из $A, B \in cl$ следует $A \cup B \in cl$.
3. Из $A, B \in cl$ следует $A \backslash B \in cl$.

Алгебра множеств широко применяется в программировании, в частности, при работе с разнообразными базами данных и составляет основу для построения многих математических структур. Вместе с тем, что алгебра множеств имеет основополагающее значение в математике, она очень проста и близка к реальной жизни. Мы ежедневно применяем операции и законы алгебры множеств, не задумываясь об этом. Мы вычитаем из множества задач, которые необходимо решить, множество решенных и приступаем к решению оставшихся. Из них мы, вероятно, в первую очередь выберем те, которые относятся к множеству легких. Мы готовим завтрак, определяя пересечение множества имеющихся продуктов с множеством продуктов, которые нам нравятся. Да и вообще, вся наша жизнь проходит среди множеств, которые как-то взаимосвязаны.

Мы имеем достаточно операций, чтобы составлять сложные алгебраические выражения. Для этого необходимо определить, каким приоритетом обладают операции относительно друг друга.

Приоритет операций в алгебре множеств следующий:

1. $\overline{A}$.
2. $A \cap B$.
3. $A \cup B$.
4. $A \backslash B$.

Рассмотрим пример.

Пример. Пусть нужно расставить скобки (определить последовательность выполнения операций) в формуле:

$$E = (A \backslash B \cup \overline{A} \cap D) \backslash B,$$

с учетом приоритетов это следует сделать так:

$$E = (A \backslash (B \cup ((\overline{A}) \cap D))) \backslash B.$$

В алгебре множеств cl автоматически выполняются следующие тождества, которые позволяют отнести cl к классу так называемых ***булевых алгебр*** (см. раздел 4).

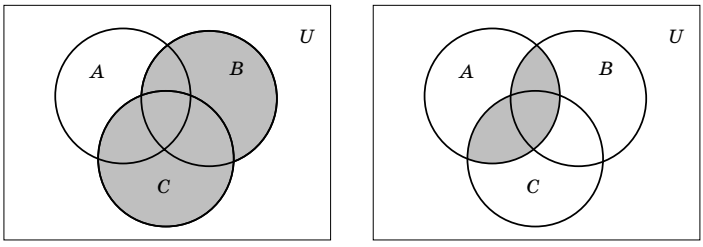
1. ***Коммутативные законы***
   1.1. $A \cup B = B \cup A$. 1.2. $A \cap B = B \cap A$.
2. ***Ассоциативные законы***
   2.1. $A \cup (B \cup C) = (A \cup B) \cup C$.
   2.2. $A \cap (B \cap C) = (A \cap B) \cap C$.

3. ***Дистрибутивные законы***
   3.1. $A \cup (B \cap C) = (A \cup B) \cap (A \cup C)$.
   3.2. $A \cap (B \cup C) = (A \cap B) \cup (A \cap C)$.
4. ***Свойства пустого и универсального множеств***
   4.1. $A \cup \varnothing = A$. 4.2. $A \cap U = A$.
   4.3. $A \cup U = U$. 4.4. $A \cap \varnothing = \varnothing$.
5. ***Законы идемпотентности***
   5.1. $A \cup A = A$. 5.2. $A \cap A = A$.
6. ***Закон инволюции***
   $\overline{\overline{A}} = A$.
7. ***Закон противоречия***
   $A \cap \overline{A} = \varnothing$.
8. ***Закон исключенного третьего***
   $A \cup \overline{A} = U$.
9. ***Закон элиминации***
   9.1. $A \cap (A \cup B) = A$. 9.2. $A \cup (A \cap B) = A$.
10. ***Законы де Моргана***
    10.1. $\overline{A \cup B} = \overline{A} \cap \overline{B}$. 10.2. $\overline{A \cap B} = \overline{A} \cup \overline{B}$.

Все перечисленные тождества можно наглядно представить и доказать, используя диаграммы Венна.

Пример. Доказать с помощью диаграмм Венна дистрибутивный закон 3.2. $A \cap (B \cup C) = (A \cap B) \cup (A \cap C)$.

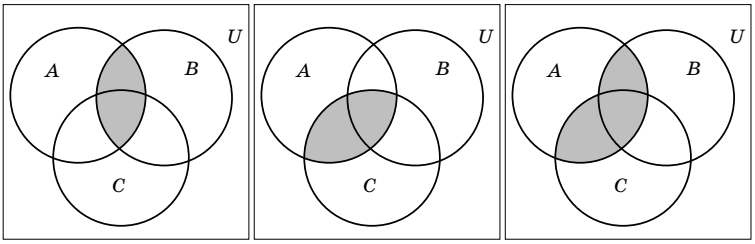
Проиллюстрируем на диаграмме левую часть тождества, выполнив сначала объединение множеств B и C, а затем пересечение с A (рис. 1.11).



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

Рис. 1.11. Построение диаграммы Венна для $A \cap (B \cup C)$

Теперь построим диаграмму для правой части тождества — $(A \cap B) \cup (A \cap C)$ (рис. 1.12).



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

Рис. 1.12. Построение диаграммы Венна для $(A \cap B) \cup (A \cap C)$

Как видим, правые диаграммы на рис. 1.11 и 1.12 совпадают, следовательно тождество 3.2 верно.

С помощью тождеств алгебры множеств можно производить эквивалентные преобразования выражений. Рассмотрим такие преобразования на примере.

Пример. Упростить выражение

$\overline{(\overline{A} \cup B \cup C)} \cap (A \cap (B \cup \overline{C})) \cap \overline{B} =$ (применим закон де Моргана)

$= (A \cap \overline{B} \cap \overline{C}) \cap (A \cap (B \cup \overline{C})) \cap \overline{B} =$ (ассоциативность и коммутативность)

$= A \cap \overline{B} \cap \overline{C} \cap A \cap \overline{B} \cap (B \cup \overline{C})) =$ (используем закон идемпотентности)

$= A \cap \overline{B} \cap \overline{C} \cap (B \cup \overline{C})) =$ (используем дистрибутивный закон)

$= A \cap \overline{B} \cap \overline{C} \cap B \cup A \cap \overline{B} \cap \overline{C} \cap \overline{C} =$ (согласно свойствам пустого множества)

$= \varnothing \cup A \cap \overline{B} \cap \overline{C} = A \cap \overline{B} \cap \overline{C}.$

Ответ: $A \cap \overline{B} \cap \overline{C}$.

Вопросы

1. Расположите операции алгебры множеств в соответствии с их приоритетом.
2. Назовите тождества алгебры множеств, запишите соответствующие формулы.
3. Каким образом можно графически представить и доказать законы и тождества алгебры множеств? Приведите пример такого доказательства.

4. Каким образом производятся эквивалентные преобразования формул алгебры множеств? Приведите примеры.

Задания

1. Докажите при помощи диаграмм Венна:
   а) ассоциативные законы;
   б) дистрибутивные законы;
   в) свойства пустого и универсального множеств.
2. Докажите при помощи эквивалентных преобразований законы элиминации.
3. Исходя из свойств пустого множества, с помощью эквивалентных преобразований получите свойства универсального множества.
4. Упростите выражения
   а) $\overline{\bar{A} \cup C} \cup (B \cup B \cap C) \cap (\bar{B} \cup \overline{B \cup C})$;
   б) $(A \cap \bar{B} \cup C) \cap (A \cup B) \cap \bar{C}$;
   в) $A \cap ((B \cap \bar{C} \cup \overline{C \cup B}) \cap C) \cap \bar{A})$;
   г) $(A \cap B \cap C) \cup (\bar{A} \cap B \cap C)$.
5. В каком отношении находятся множества A и B, если $A \backslash B = B \backslash A = \varnothing$?
6. Является ли алгеброй класс множеств, замкнутых относительно:
   а) операций $\cup$, $\cap$;
   б) операций $\cup$, дополнения.

1.6. Бесконечные множества

Счетные и континуальные множества, мощность

В дискретной математике, как и во всей математике, в естествознании очень важна роль натурального ряда чисел $N = \{1, 2, ...\}$. Хотя практические вычисления всегда производятся с конечным отрезком $\{1, 2, ..., n\}$ этого бесконечного множества, нет возможности указать наибольшее число n, которое не будет превышено во всех случаях жизни. Поэтому приходится исследовать свойства всего бесконечного множества чисел N. На вещественной числовой оси R натуральные и целые числа образуют довольно «разреженные» подмножества N, Z (рис. 1.13), которые интуитивно можно назвать «дискретные». Множество рациональных чисел Q, образующихся при делении целых чисел m/n, плотно располагается на вещественной оси R: интервал (a, b) сколь угодно малой длины $\varepsilon = b - a$ $(a \neq b)$ содержит бесконечное множество различных рациональных

точек. Можно ли множество Q считать «дискретным»? Оказывается да, хотя интуиция отказывается это признать.

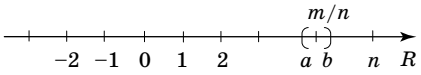


Рис. 1.13. Натуральные, целые и рациональные числа на оси R

Наконец, вся вещественная ось R — явно непрерывное множество, и здесь интуитивное заключение нас не подводит. Для строгого определения дискретных и непрерывных бесконечных множеств используются сопоставления дискретных множеств натуральному ряду чисел, а непрерывных множеств — отрезку $[0, 1]$ вещественной оси. Необходимо только уточнить термин «сопоставить». Имеется в виду установить взаимно однозначное соответствие.

Понятие взаимно однозначного соответствия является первичным в сознании человека при дифференцированном восприятии предметов внешнего мира. Если зрительный зал заполнен и нет свободных мест, нам не обязательно считать количество сидящих зрителей, оно равно числу кресел в зале. Чтобы сделать этот вывод, мы интуитивно используем наличие взаимно однозначного соответствия между зрителями и креслами. Отметим одну особенность: фактически на данном концерте реализовано конкретное взаимно однозначное соответствие зрители — кресла (каждому зрителю отвечает одно и только одно определенное кресло и наоборот). После перерыва некоторые зрители могут поменяться местами, и конкретное соответствие станет другим, но вывод останется прежним: число зрителей равно числу мест.

Определение

Взаимно однозначным называется такое ***соответствие*** между множествами A и B, при котором каждому элементу $a \in A$ отвечает один и только один элемент $b \in B$ и каждому элементу $b \in B$ отвечает один и только один элемент $a \in A$. Функция, определяющая взаимно однозначное соответствие, называется ***биективной функцией*** или ***биекцией***.

Определение

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

В примере с заполненным зрительным залом множество зрителей эквивалентно множеству кресел. Таким образом,

эквивалентными друг другу оказываются все конечные множества с одинаковым числом элементов n, и число n считается ***мощностью*** этих множеств.

Для бесконечного множества строгое понятие мощности не вводится, но сам термин ***мощность*** используется для обозначения свойства, общего для всех эквивалентных множеств. Если два бесконечных множества A и B эквивалентны ($A \sim B$), то равенство их мощностей формально записывается как $|A| = |B|$.

Определение

Множество A называется ***счетным***, если оно эквивалентно натуральному ряду N ($A \sim N$). Термин «***счетность***» является точным заменителем интуитивного понятия «***дискретность***». С помощью биекции $\varphi = N \to A$ можно пересчитать все элементы a из A, присвоив им индексы по правилу $\varphi(n) = a_n$. Можно записать, что $A = \{a_n, n = 1, 2, ...\}$. Множества четных натуральных чисел $N_ч = \{2, 4, ..., m, ...\}$, всех натуральных чисел $N = \{1, 2, ..., n, ...\}$, целых чисел Z и рациональных чисел Q последовательно вложены: $N_ч \subset N \subset Z \subset Q$. Хотя для любых двух из этих множеств нет равенства, они ***эквивалентны друг другу***, то есть имеют одинаковую мощность и являются счетными: $|N_ч| = |N| = |Z| = |Q|$. Поэтому в соответствии с нашими соглашениями множества $N_ч$, N, Z, Q являются ***дискретными***.

Действительно, эквивалентность $N \sim N_ч$ аргументируется с помощью биекции $\varphi(n) = 2n : 2n = m$. Множество целых чисел Z можно «пересчитать» (то есть присвоить его элементам натуральные индексы) по правилу:

$$0 = z_1; \quad 1 = z_2; \quad -1 = z_3; \quad 2 = z_4; \quad -2 = z_5; \quad 3 = z_6; \ ...$$

Следовательно $Z \sim N$.

Для доказательства эквивалентности множеств Q и N достаточно указать правило нумерации множества Q_+ положительных рациональных чисел, образующихся делением m/n натуральных чисел m и n. Пронумеруем столбцы и строки бесконечной таблицы индексами m и n соответственно (рис. 1.14).

n \ m	1	2	3	4	...
1	1/1	2/1	3/1	4/1	...
2	1/2	2/2	3/2	4/2	...
3	1/3	2/3	3/3	4/3	...
4	1/4	2/4	3/4	4/4	...
...	...	...	...	...	...

Рис. 1.14. Таблица множества Q_+

Будем нумеровать последовательно числа n/m вдоль пунктирных наклонных стрелок, начиная с левого верхнего угла таблицы и переходя после прохождения каждой стрелки к соседней, более длинной. При этом пропускаются отношения m/n, численные значения которых встречались ранее:

$$q_1 = 1 = 1/1, \quad q_2 = 1/2, \quad q_3 = 2 = 2/1, \quad q_4 = 1/3,$$
$$q_5 = 3 = 3/1, \quad q_6 = 1/4, \quad q_7 = 2/3, \quad q_8 = 3/2, \quad q_9 = 4 = 4/1, \ldots$$

Таким образом, множество Q рациональных чисел счетно:

$$Q = \{q_n\}, \quad n = 1, 2, \ldots.$$

Разумеется, существуют бесконечные ***несчетные*** множества, и их мощность естественно считать большей, чем $|N|$. Так, множество точек отрезка $[0, 1] = \{x \in R;\ 0 \le x \le 1\}$ *не является счетным* (теорема Г. Кантора). Его мощность называется ***континуум*** и обозначается малой буквой c: $|[0, 1]| = c$. Само множество $[0, 1]$ и любое эквивалентное ему множество называются ***континуальными***.

Оказывается, что на вещественной оси R континуальными (то есть эквивалентными друг другу и отрезку $[0, 1]$) являются, например, множества $[a, b]$, (a, b), при любом $a < b$; $(0, +\infty)$; множество $(-\infty, +\infty)$, равное R.

Континуальны также множества точек любого квадрата и круга на плоскости R^2, параллелепипеда и шара в пространстве R^3 и самого пространства R^3: $|R| = |R^2| = |R^3| = c$.

Задания

1. Доказать следующие утверждения:

   1.1. Из всякого бесконечного множества можно выделить счетное подмножество.

   1.2. Всякое бесконечное подмножество счетного множества также счетно.

   1.3. Сумма (объединение) конечного и счетного множеств есть счетное множество.

   1.4. Объединение конечного числа счетных множеств есть счетное множество.

2. Какова мощность множества многочленов любых степеней с целыми коэффициентами?

3. Вещественное число называется *алгебраическим*, если оно является корнем некоторого многочлена с целыми коэффициентами. Все остальные числа называются *трансцендентными*. Какова мощность множества алгебраических чисел?

раздел 2

Отношения

2.1. Понятие отношения. Задание отношений

Декартово произведение множеств, n-арное отношение, бинарное отношение, способы задания отношений. Частные случаи отношений

Отношения реализуют в математических терминах на абстрактных множествах реальные связи между реальными объектами. Отношения применяются при построении компьютерных баз данных, которые организованы в виде таблиц данных. Связи между группами данных в таблицах описываются на языке отношений. Сами данные обрабатываются и преобразуются с помощью операций, математически точно определенных для отношений. Такие базы данных называются реляционными и широко применяются для хранения и обработки самой разнообразной информации: производственной, коммерческой, статистической и др. Отношения также часто используются в программировании. Такие составные структуры данных, как списки, деревья и др. обычно используются для описания некоторого множества данных вместе с отношением между элементами этого множества.

Пример. Некоторые предприятия, занимающиеся дизайном интерьера помещений, используют виды продукции ряда производственных фирм, расположенных в двух городах. Для анализа необходимо составить «отношение» реальных комбинаций трех параметров: название фирмы, места ее расположения (город), вида выпускаемой продукции.

Пусть известно, что ЧПП «Orion» (Одесса) продает мебель, ООО «День» (Харьков) продает светильники, ЧКП «Sit» (Одесса) торгует мебелью и светильниками, ООО «House» (Харьков) продает светильники и материалы для ремонта.

В этом отношении участвуют три множества:

Фирмы = {ЧПП «Orion», ООО «День», ЧКП «Sit», ООО «House»} — множество фирм.

Города = {Одесса, Харьков} — множество городов.

Продукция = {мебель, светильники, материалы для ремонта} — множество видов продукции.

Это отношение можно формально представить списком элементов следующим образом: {(ЧПП «Orion» (Одесса) — мебель), (ООО «День» (Харьков) — светильники), (ЧКП «Sit» (Одесса) — мебель), (ЧКП «Sit» (Одесса) — светильники), (ООО «House» (Харьков) — светильники), (ООО «House» (Харьков) — материалы для ремонта)}. Мы видим, что полученное отношение — это множество, состоящее из упорядоченных групп типа (ЧПП «Orion» (Одесса) — мебель). Первый элемент группы принадлежит множеству «Фирмы», второй — множеству «Города», третий — множеству «Продукция». Кроме того, не все возможные комбинации элементов этих множеств принадлежат нашему отношению, а только некоторые из них, например, такая группа, как (ЧПП «Orion» (Одесса) — материалы для ремонта) не принадлежит нашему отношению, поскольку ЧПП «Orion» торгует только мебелью, а группа (ЧПП «Orion» (Харьков) — мебель) не принадлежит нашему отношению, поскольку ЧПП «Orion» находится в Одессе, а не в Харькове. Получается, что наше отношение — это некоторое подмножество множества всевозможных комбинаций фирм, городов, продукций.

Для формального описания всевозможных комбинаций из элементов множеств, входящих в отношение, используется понятие декартова произведения множеств. Рассмотрим это понятие и затем, используя его, определим формально понятие отношения.

Определение

Декартовым произведением множеств $X_1 \times X_2 \times \ldots \times X_n$ называется множество всех возможных упорядоченных наборов $(x_1, x_2, \ldots, x_n)$ из n элементов, в которых первый элемент принадлежит множеству X_1, второй — множеству X_2, ..., n-й — множеству X_n. Декартово произведение $X \times X \times \times \ldots \times X$, в котором одно и то же множество X умножается

n раз само на себя, называют ***декартовой степенью*** множества и обозначают X^n. При этом $X^1 = X$. Множество X^2 называют ***декартовым квадратом*** множества X, множество X^3 — ***декартовым кубом*** множества X.

Пример. Пусть $A = \{a_1, a_2, a_3\}$, $B = \{b_1, b_2\}$, $C = \{c_1, c_2\}$. Тогда

$$A \times B = \{(a_1, b_1), (a_1, b_2), (a_2, b_1), (a_2, b_2), (a_3, b_1), (a_3, b_2)\}.$$

$$B \times A = \{(b_1, a_1), (b_1, a_2), (b_1, a_3), (b_2, a_1), (b_2, a_2), (b_2, a_3)\}.$$

$$A \times B \times C = \{(a_1, b_1, c_1), (a_1, b_1, c_2), (a_1, b_2, c_1), (a_1, b_2, c_2), (a_2, b_1, c_1), (a_2, b_1, c_2), (a_2, b_2, c_1), (a_2, b_2, c_2), (a_3, b_1, c_1), (a_3, b_1, c_2), (a_3, b_2, c_1), (a_3, b_2, c_2)\}.$$

$$B^2 = \{(b_1, b_1), (b_1, b_2), (b_2, b_1), (b_2, b_2)\}.$$

Порядок следования пар может быть произвольным, но размещение элементов в каждой паре определяется порядком следования перемножаемых множеств, т.е. $A \times B \neq B \times A$, если $A \neq B$.

Определение

n-арное отношение R на множествах $X_1, X_2, \ldots, X_n$ — это подмножество декартова произведения этих n множеств: $R \subseteq X_1 \times X_2 \times, \ldots, \times X_n$.

Если набор элементов $(x_1, x_2, \ldots, x_n)$ принадлежит отношению R, то говорят, что элементы $x_1, x_2, \ldots, x_n$ находятся в отношении R. Под ***n-арным отношением R на множестве X*** понимается подмножество n-й степени этого множества: $R \subset X^n$. Если $n = 1$, то отношение называется ***унарным***, если $n = 2$ — ***бинарным***. Отметим, что унарное отношение R на множестве X — это подмножество в самом X: $R \subset X$.

Пример. Отношением на множествах A, B, C из предыдущего примера является всякое подмножество множества $A \times B \times C$, в частности

$$R_1 = \{(a_1, b_1, c_1), (a_2, b_1, c_1), (a_2, b_1, c_2), (a_3, b_1, c_1), (a_3, b_1, c_2), (a_3, b_2, c_2)\};$$

$$R_2 = \{(a_2, b_2, c_1), (a_2, b_2, c_2), (a_3, b_1, c_1)\}.$$

Перейдем к исследованию бинарных отношений, которые являются «базисными» в том смысле, что любое n-арное отношение можно представить в виде цепочки последовательно конструируемых бинарных отношений. Этот очевидный факт является следствием ассоциативности декартового произведения множеств.

Способы задания бинарных отношений

Если R — бинарное отношение на множествах X, Y, то факт $(x, y) \in R$ часто записывается в виде xRy, и говорят, что элемент $x \in X$ находится в отношении R с элементом $y \in Y$.

1. Любое бинарное отношение может быть задано в виде ***списка***, элементами которого являются пары, из которых состоит отношение.

Пример. На множествах чисел $A = \{1, 2, 3, 4, 5, 6, 7, 8, 9\}$; $B = \{24, 25, 26\}$ построим отношение «делитель», которое состоит из упорядоченных пар вида (x, y), где x — делитель y, $x \in A$, $y \in B$. Обозначим это отношение через R:

$$R = \{(1, 24), (1, 25), (1, 26), (2, 24), (2, 26), (3, 24), (4, 24), (5, 25), (6, 24), (8, 24)\}.$$

2. Бинарное отношение R на множествах X и Y может быть задано с помощью ***матрицы*** ($W = W(R)$), строки которой отвечают элементам множества X, столбцы — элементам множества Y. Если $n = |X|$, $m = |Y|$ — количества элементов множеств X и Y соответственно, то произвольная матрица W имеет размерность $n \times m$. Элемент w_{ij} матрицы соответствует паре $(x_i, y_j) \in A \times B$ декартового произведения множеств, причем $w_{ij} = 1$, если $(x_i, y_j) \in R$, и $w_{ij} = 0$, если $(x_i, y_j) \notin R$, то есть отношение R не содержит пару (x_i, y_j).

Пример. Следующая матрица W задает отношение R «делитель» для числовых множеств A и B из предыдущего примера:

$W =$

A \ B	**24**	**25**	**26**
1	1	1	1
2	1	0	1
3	1	0	0
4	1	0	0
5	0	1	0
6	1	0	0
7	0	0	0
8	1	0	0
9	0	0	0

3. Бинарное отношение R на множествах X, Y может быть задано ***графически***. На плоскости изобразим точками x_i и y_j элементы множеств X и Y. Если пара (x_i, y_j) принадлежит отношению R, соединяем изображающие точки x_i, y_j линией, направленной от первого элемента пары ко второму. Обозначив таким образом все пары, принадлежащие отношению R, получим фигуру, которая

называется ***графом*** отношения. Направленные линии, соединяющие пары точек, называются ***дугами***, а точки, изображающие элементы множеств, — ***вершинами*** графа. Если бинарное отношение R задано на одном множестве X ($R \subseteq X^2$), то вершинами графа будут элементы множества X.

Пример. В качестве примера графического задания отношения рассмотрим генеалогическое дерево некоторой семьи, изображенное на рис. 2.1

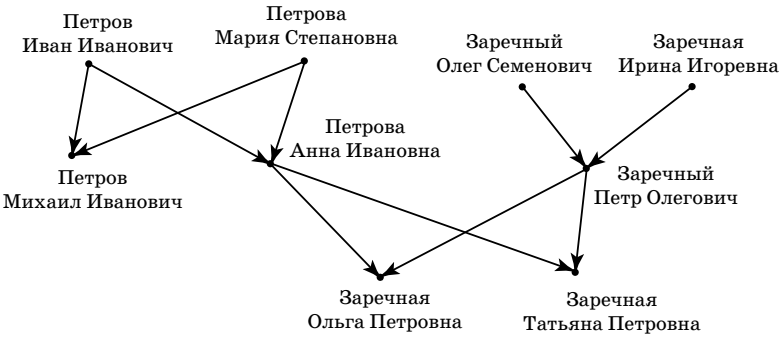


Рис. 2.1. Граф отношения «родитель»

Отношение «родитель» в данном случае задано на множестве, состоящем из 9 человек. Рис 2.1 представляет собой граф отношения «родитель». Упорядоченные пары элементов множества, принадлежащие отношению, соединены дугами графа. Первый элемент такой пары является родителем, второй — его ребенком. Дуга графа направлена от первого элемента ко второму.

Рассмотрим некоторые частные случаи отношений. Пусть задано бинарное отношение R на множестве A: $R \subseteq A^2$. Возможен случай, когда $R = A^2$, — такое отношение называется ***полным***. Для пятиэлементного множества A граф полного отношения представлен на рис. 2.2,*б*. Может случиться, что $R = \varnothing$ — такое отношение называется ***пустым***. При $|A| = 5$ граф представлен на рис 2.2,*в*. Если отношение содержит все возможные пары вида (a, a) и не содержит других пар элементов, то такое отношение называется ***тождественным***. Граф этого отношения представлен на рис. 2.2,*а* для $|A| = 5$.

Если ***полное отношение*** задано с помощью матрицы, то все элементы этой матрицы равны единице. Матрица ***пустого отношения*** состоит из нулевых элементов.

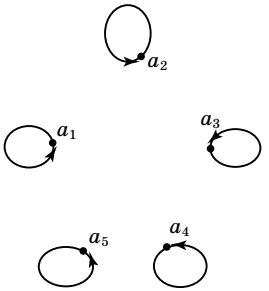


Рис. 2.2,*а*.
Граф тождественного
отношения

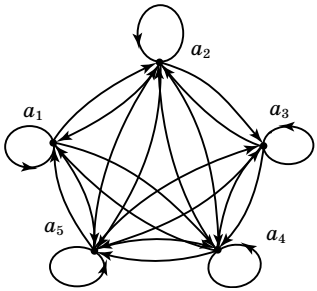


Рис. 2.2,*б*.
Граф полного
отношения

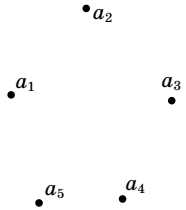


Рис. 2.2,*в*.
Граф пустого
отношения

Мы изучили три способа задания отношений. Перечислением элементов возможно задавать n-арные отношения при $n \geq 2$. С помощью матрицы и графа удобно задавать только бинарные отношения.

Вопросы

1. Как связаны теория отношений и теория множеств?
2. Дайте определение декартова произведения множеств.
3. Пусть A — некоторое множество. Что означает запись A^2, A^3, A^n?
4. Пусть A и B — множества. Объясните, почему $A \times B \neq B \times A$.
5. Что называется отношением? Что такое n-арное отношение, бинарное, унарное?
6. Перечислите способы задания отношений. Какие из них применимы для n-арных отношений при $n > 2$?
7. Объясните понятия «граф», «вершина», «дуга».
8. Что такое тождественное отношение, полное отношение, пустое отношение? Изобразите графы этих отношений, постройте их матрицы.

Задания

1. Постройте матрицу и граф для следующих отношений, определенных на множестве $A = \{1, 2, 3\}$:
   а) $\{(1, 1), (1, 2), (1, 3)\}$;
   б) $\{(1, 1), (2, 1), (2, 2), (2, 3)\}$;
   в) $\{(1, 1), (1, 2), (1, 3), (2, 2), (2, 3), (3, 3)\}$;
   г) $\{(1, 3), (3, 1)\}$.
2. Постройте граф и список элементов для следующих отношений, определенных на множестве $A = \{a, b, c\}$ матрицами:

а)

	a	*b*	*c*
a	1	0	1
b	0	1	0
c	1	0	1

б)

	a	*b*	*c*
a	0	1	0
b	0	1	0
c	0	1	0

в)

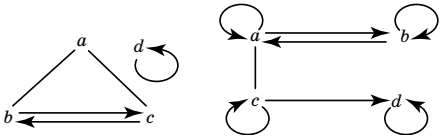
	a	*b*	*c*
a	1	1	1
b	1	0	1
c	1	1	1

г)

	a	*b*	*c*
a	1	0	1
b	0	1	0
c	1	0	1

3. Постройте матрицу и список элементов для следующих отношений, заданных графически:

а) *б)*



4. Запишите список элементов для 3-арного отношения R, заданного на множестве натуральных чисел следующим образом: $(a, b, c) \in R$, если $0 < a < b < c < 5$.
5. Запишите список элементов для 4-арного отношения R, заданного на множестве натуральных чисел следующим образом: $(a, b, c, d) \in R$, если $abcd = 6$.
6. Задайте закон, определяющий отношение H, связывающее клетки шахматной доски, расположенные на расстоянии одного хода конем, т.е. $(x, y) \in H$, если конь может перейти с клетки x на клетку y за один ход.
7. Докажите, что на множестве A, состоящем из n элементов, может быть задано 2^{n^2} различных бинарных отношений. Сколько 3-арных отношений может быть задано на этом множестве?
8. Сколько различных отношений может быть задано между множествами A и B, если $|A| = n$, $|B| = m$?
9. Составьте 16 различных отношений на множестве $\{0, 1\}$.
10. Определите алгоритм составления матрицы отношения по заданному списку элементов и наоборот.

2.2. Операции над отношениями

Обратное отношение, композиция отношений, степень отношения, сечение отношения, фактор-множество

Вернемся к отношению «родитель». Если упорядоченная пара элементов (x, y) принадлежит этому отношению, это значит, что x является родителем y. Но если x является родителем y, то y является ребенком x. Из этого следует, что существует и отношение «ребенок», которому принадлежит пара (y, x). Очевидно, что граф отношения «ребенок» можно получить из графа отношения «родитель» изменением направления дуг, как показано на рис. 2.3, который «обращает» рис. 2.1.

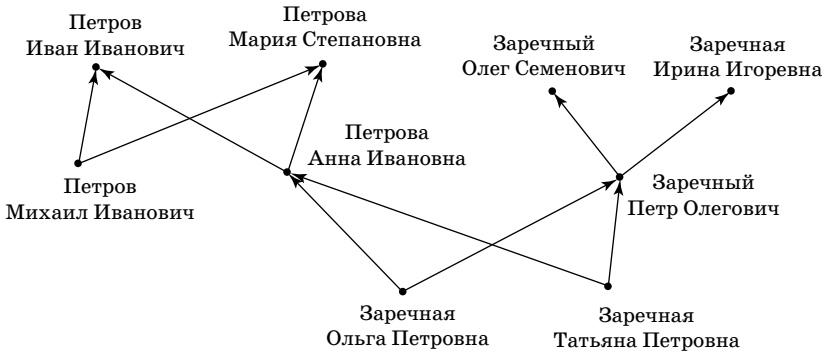


Рис 2.3. Граф отношения «ребенок»,
обратного отношению «родитель»

Этот пример поясняет понятие обратного отношения.

Определение

Пусть R-бинарное отношение. ***Обратное*** отношение к R обозначается $\boldsymbol{R^{-1}}$. ***Упорядоченная пара*** (x, y) принадлежит R^{-1} тогда и только тогда, когда (y, x) принадлежит R.

Если $R \subseteq X^2$, то $R^{-1} \subseteq X^2$, где X — некоторое множество. Если $R \subseteq X \times Y$, то $R^{-1} \subseteq Y \times X$.

Если бинарное отношение задано на двух множествах, то граф отношения можно построить следующим образом. Вершины графа, соответствующие элементам первого множества, располагаются слева, вершины графа, соответствующие элементам второго множества, располагаются справа. Таким образом, дуги графа направлены слева направо. Рассмотрим пример. Пусть заданы множества $A = \{a, b, c, d, e, f\}$, $B = \{1, 2, 3, 4\}$ и отношение $R = \{(a, 1), (a, 2), (b, 4), (d, 1), (f, 4)\}$. Граф этого отношения изображен на рис. 2.4.

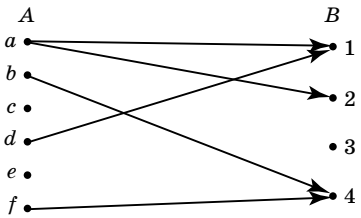


Рис. 2.4. Граф отношения $R = \{(a, 1), (a, 2), (b, 4), (d, 1), (f, 4)\}$

Для того чтобы построить граф отношения R^{-1}, изменим направления дуг (рис. 2.5).

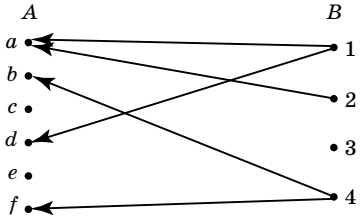


Рис. 2.5. Граф отношения $R^{-1} = \{(1, a), (2, a), (4, b), (1, d), (4, f)\}$

Теперь изучим способ получения отношения из двух других отношений, используя операцию композиции. Пусть имеется множество $C = \{w, x, y, z\}$ и $B = \{1, 2, 3, 4\}$ и отношение $S = \{(1, x), (2, y), (3, x), (3, z)\}$, $S \subseteq B \times C$. Дополним рис. 2.4, изобразив на нем, кроме графа отношения R, граф отношения S.

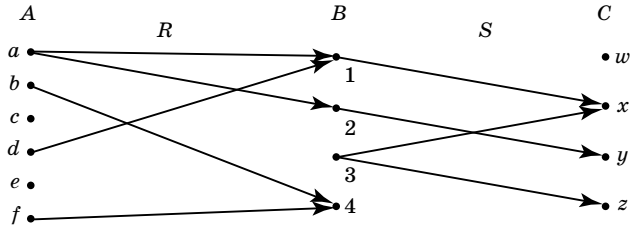


Рис. 2.6. Граф отношения R и отношения $S = \{(1, x), (2, y), (3, x), (3, z)\}$

Определение

Пусть R и S — отношения, такие, что $R \subseteq X \times Y$, $S \subseteq Y \times Z$, где X, Y, Z — некоторые множества. ***Композицией отношений*** R и S называется отношение, состоящее из упорядоченных пар (x, z), $x \in X$, $z \in Z$, для которых существует элемент $y \in Y$, такой, что выполняются условия $(x, y) \in R$, $(y, z) \in S$. Композиция отношений R и S обозначается $S \circ R$.

В частности, для отношений $R \subseteq A \times B$ и $S \subseteq B \times C$, изображенных на рис. 2.6, композиция $S \circ R$ есть отношение, изображенное на рис. 2.7 и являющееся подмножеством декартового произведения $A \times C$.

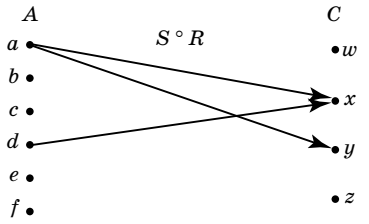


Рис. 2.7. Граф отношения $S \circ R$

Заметим, что для пары $(x, z) \in S \circ R$ «промежуточных» элементов Y может быть несколько, однако их количество (если оно не нулевое) не влияет на вид композиции $S \circ R$.

Операция композиций отношений позволяет ввести понятие степени бинарного отношения, заданного на одном множестве.

Определение

Пусть R — некоторое отношение, определенное на множестве X: $R \subseteq X \times X$. Тогда ***n*-ая степень** отношения R обозначается R^n и определяется рекурсивно так:

R^0 — тождественное отношение на множестве X;

$R^n = R^{n-1} \circ R$, для $n = 1, 2, \ldots$

Из определения следует, что $R^1 = R$, $R^2 = R \circ R$, $R^3 = R^2 \circ R$, и т.д.

Пример. Пусть отношение R на множестве $A = \{a, b, c, d\}$ задано графом, изображенным на рис. 2.8.

Рис. 2.8. Граф отношения R

Построим степени отношения R. Квадрат отношения R^2 можно построить, используя правило: в графе R^2 присутствует дуга (x, y), если существует такая вершина z, что верно xRz и zRy (рис. 2.9).

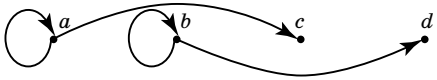


Рис. 2.9. Граф отношения R^2

Степень R^3 можно построить аналогично, используя правило: в графе R^3 присутствует дуга (x, y), если существует такая вершина z, что верно xR^2z и zRy (рис. 2.10).

Рис 2.10. Граф отношения R^3

Построим также отношение R^4. Правило построения аналогично: xR^4y верно, если существует такая вершина z, что верно xR^3z и zRy (рис. 2.11).

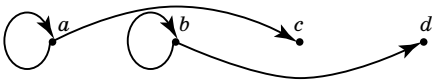


Рис 2.11. Граф отношения R^4

Мы видим, что построенный граф R^4 не отличается от графа отношения R^2, поэтому в результате дальнейших построений отношений R^5, R^6..., мы получим отношения, совпадающие с R^2, R^3.

Ясно, что справедлива следующая графическая трактовка степени отношения: в графе отношения R^n имеется дуга из x в y, если в графе R из вершины x в вершину y ведет хотя бы один маршрут длины n (состоящий из n дуг), причем одна и та же дуга может быть включена в маршрут сколько угодно раз.

Рассмотрим еще одну операцию, которая называется сечением бинарного отношения.

Определение

Пусть $R \subset X \times Y$ — отношение на множествах X и Y. Если $x \in X$, то ***сечение*** отношения R по x, обозначаемое $R(x)$, есть множество $R(x) \subseteq Y$, состоящее из элементов $y \in Y$, таких, что $(x, y) \in R$. Объединение сечений по элементам некоторого подмножества $Z \subset X$ называется ***сечением $R(Z)$ относительно подмножества Z***. Множество, состоящее из сечений отношения $R \subseteq X \times Y$ по каждому элементу из X, называется ***фактор-множеством*** множества Y по отношению R и обозначается Y/R. Формально можно записать, что $Y/R = \{R(x), \forall x \in X\}$.

Пример. Рассмотрим сечения отношения R на множествах $A = \{a, b, c, d, e, f\}$, $B = \{1, 2, 3, 4\}$, $R \subseteq A \times B$, заданного графом (рис. 2.4): $R = \{(a, 1), (a, 2), (b, 4), (d, 1), (f, 4)\}$. Можно получить следующие сечения: $R(a) = \{1, 2\}$, $R(b) = \{4\}$, $R(d) = \{1\}$, $R(e) = \varnothing$, $R(f) = \{4\}$. Фактор-множество $B/R = \{\{1, 2\}, \{4\}, \{1\}, \varnothing\}$. Рассмотрим множество $D = \{d, e, f\}$, $D \subset A$. Сечение отношения R по

множеству D выглядит следующим образом: $R(D) = R(d) \cup R(e) \cup \cup R(f) = \{\{1\}, \{4\}, \varnothing\}$.

Кроме перечисленных операций, к отношениям применимы обычные теоретико-множественные операции, как объединение, пересечение, дополнение и разность. Рассмотрим два отношения R и S, определенные на множествах X и Y, $R \subseteq X \times Y$, $S \subseteq X \times Y$. В результате выполнения операций $R \cup S$, $R \cap S$ и $R \backslash S$ получаем множества упорядоченных пар, являющиеся соответственно объединением, пересечением и разностью множеств R и S. Результаты выполнения этих операций также являются отношениями на множествах X и Y. Дополнение отношения R обозначается $\overline{R}$ и содержит все упорядоченные пары множества $X \times Y$, кроме тех пар, которые принадлежат R.

Вопросы

1. Что называется обратным отношением, и как оно обозначается?
2. Как получить граф отношения, обратного данному?
3. Что называется композицией отношений?
4. Пусть R — некоторое отношение. Что означает запись R^2, R^3, R^n (степень отношения рассматривается относительно операции композиции отношений)? Как построить эти отношения, и всегда ли это возможно?
5. Дайте определения сечения отношения R и фактор-множества по отношению R.
6. Перечислите теоретико-множественные операции, которые применимы к отношениям. Объясните, каким образом эти операции применяются к отношениям.

Задания

1. Пусть R_1 и R_2 — бинарные отношения на множестве $A = \{a, b, c, d\}$, где $R_1 = \{(a, a), (a, b), (b, d)\}$, $R_2 = \{(a, d), (b, c), (b, d), (c, d)\}$:
   а) постройте отношения $R_1 \circ R_2$, $R_2 \circ R_1$, $R_1^2 R_2^3$;
   б) постройте сечения отношений R_1, R_2 по элементам a, d и подмножеству $\{a, b\}$;
   в) постройте фактор-множество по отношению R_2.
2. Найдите отношение R^{-1}, если отношение R задано следующим образом:
   а) $(a, b) \in R$, если $a, b \in N$, $a > b$;
   б) $(a, b) \in R$, если $a, b \in N$, a — делитель b;
   в) A — множество стран мира; $(a, b) \in R$, если $a, b \in A$ и страна a граничит с b.

3. Пусть A — некоторое множество людей, R — отношение «родитель», определенное на этом множестве так, что $(a, b) \in R$, если a — родитель b. Далее S — отношение «брат-сестра», определенное на множестве A по правилу $(a, b) \in S$, если a и b имеют одну и ту же пару родителей. Опишите отношения $R \circ S$, $S \circ R$, R^2, R^{-1}, $S \circ R^{-1}$. Выразите через отношения S, R отношение C — «двоюродный брат-сестра», где $(a, b) \in C$, если a — двоюродный брат или сестра b.
4. Пусть даны множества $A = \{1, 2, 3\}$, $B = \{1, 2, 3, 4\}$ и отношения $R_1, R_2 \subseteq A \times B$: $R_1 = \{(1, 2), (2, 3), (3, 4)\}$, $R_2 = \{(1, 1), (1, 2), (2, 1), (2, 2), (2, 3), (3, 1), (3, 2), (3, 3), (3, 4)\}$. Определите:
   а) $R_1 \cup R_2$;
   б) $R_1 \cap R_2$;
   в) $R_1 \backslash R_2$;
   г) $R_2 \backslash R_1$.
5. Пусть A — множество студентов университета, B — множество книг в библиотеке. Пусть заданы отношения $R_1, R_2 \subseteq A \times B$, такие, что $(a, b) \in R_1$, если студент a, согласно учебной программе, должен в ходе обучения прочесть книгу b, и $(a, b) \in R_2$, если студент a в ходе обучения уже прочел книгу b. Дайте словесное описание отношений, получаемых в результате выполнения операций:
   а) $R_1 \cup R_2$;
   б) $R_1 \cap R_2$;
   в) $R_1 \backslash R_2$;
   г) $R_2 \backslash R_1$.

2.3. Свойства бинарных отношений

Рефлексивность, антирефлексивность, симметричность, асимметричность, антисимметричность, транзитивность, антитранзитивность

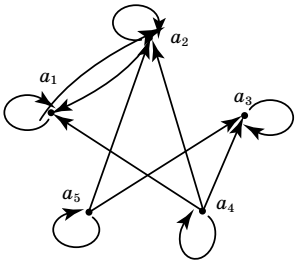
Каждое бинарное отношение может обладать одним или несколькими из перечисленных свойств. Эти свойства определяют вид матрицы и графа отношения.

1. Рефлексивность

Отношение R на множестве X называется ***рефлексивным***, если для любого $x \in X$ имеет место xRx, то есть каждый элемент $x \in X$ находится в отношении R к самому себе.

Свойство рефлексивности при задании отношения матрицей характеризуется тем, что все диагональные элементы матрицы равны 1; при задании отношения графом каждый элемент имеет петлю — дугу (x, x).

Пример. Рефлексивным является отношение, представленное на рис. 2.12.



	a_1	a_2	a_3	a_4	a_5
a_1	1	1	0	0	0
a_2	1	1	0	0	0
a_3	0	0	1	0	0
a_4	1	1	1	1	0
a_5	0	1	1	0	1

Рис. 2.12. Пример графа и матрицы рефлексивного отношения

2. **Антирефлексивность**

Отношение R на множестве X называется ***антирефлексивным***, если из $x_1 R x_2$ следует, что $x_1 \neq x_2$.

Свойство антирефлексивности при задании отношения матрицей характеризуется тем, что все диагональные элементы являются нулевыми. При задании такого отношения графом ни одна вершина не имеет петли — нет дуг вида (x, x).

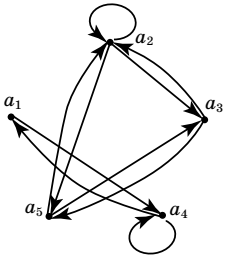
Отношение $\leq$ на множестве вещественных чисел рефлексивно, отношение $<$ на множестве вещественных чисел — антирефлексивно. Отношение «иметь общий делитель» на множестве целых чисел рефлексивно, отношение «быть сыном» на множестве людей — антирефлексивно. Отношение «быть симметричным относительно оси X на множестве точек координатной плоскости» не является ни рефлексивным, ни антирефлексивным: точка плоскости симметрична сама себе, если она лежит на оси x, и несимметрична сама себе в противном случае.

3. **Симметричность**

Отношение R на множестве X называется ***симметричным***, если для пары $(x_1, x_2) \in X^2$ из $x_1 R x_2$ следует $x_2 R x_1$ (иначе говоря, для любой пары R выполняется либо в обе стороны, либо не выполняется вообще).

Матрица симметричного отношения является симметричной относительно главной диагонали, а в задающем графе для каждой дуги из x_i в x_k существует противоположно направленная дуга из x_k в x_i.

Пример. Пример симметричного отношения представлен на рис. 2.13.



	a_1	a_2	a_3	a_4	a_5
a_1	0	0	0	1	0
a_2	0	1	1	0	1
a_3	0	1	0	0	1
a_4	1	0	0	1	0
a_5	0	1	1	0	0

Рис. 2.13. Пример графа и матрицы симметричного отношения

4. Асимметричность

Отношение R называется ***асимметричным***, если для пары $(x_1, x_2) \in X^2$ из $x_1 R x_2$ следует, что не выполняется $x_2 R x_1$ (иначе говоря, для любой пары R выполняется либо в одну сторону, либо не выполняется вообще).

5. Антисимметричность

Отношение R называется ***антисимметричным***, если из $x_1 R x_2$ и $x_2 R x_1$ следует, что $x_1 = x_2$.

Пример. Отношение «быть симметричным относительно оси X на множестве точек координатной плоскости» является симметричным: если первая точка симметрична второй, то и вторая симметрична первой. Пример антисимметричного отношения — отношение $\leq$ на множестве действительных чисел: если $a \leq b$ и $b \leq a$, то $a = b$. Отношение $<$ на вещественной оси является асимметричным, т.к. если $a < b$, то $b < a$ не выполняется.

6. Транзитивность

Отношение R называется ***транзитивным***, если из $x_1 R x_2$ и $x_2 R x_3$ следует $x_1 R x_3$.

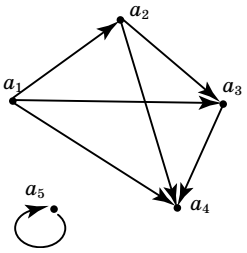
В графе, задающем транзитивное отношение R, для всякой пары дуг, таких, что конец первой совпадает с началом второй, существует третья дуга, имеющая общее начало с первой и общий конец со второй.

Пример. Отношения $\leq$ и $<$ на множестве действительных чисел транзитивны: если $a \leq b$ и $b \leq c$, то $a \leq b$.

Пример. Пример транзитивного отношения представлен на рис. 2.14.

7. Антитранзитивность

Отношение R называется ***антитранзитивным***, если из $x_1 R x_2$ и $x_2 R x_3$ следует, что не выполняется $x_1 R x_3$.



	a_1	a_2	a_3	a_4	a_5
a_1	0	1	1	1	0
a_2	0	0	1	1	0
a_3	0	0	0	1	0
a_4	0	0	0	0	0
a_5	0	0	0	0	1

Рис. 2.14. Пример графа и матрицы транзитивного отношения

Пример. Отношения «равенство», «быть делителем», заданные на множестве целых чисел, и отношение «жить в одном городе», заданное на множестве людей, — транзитивны; отношение «быть сыном» — антитранзитивно.

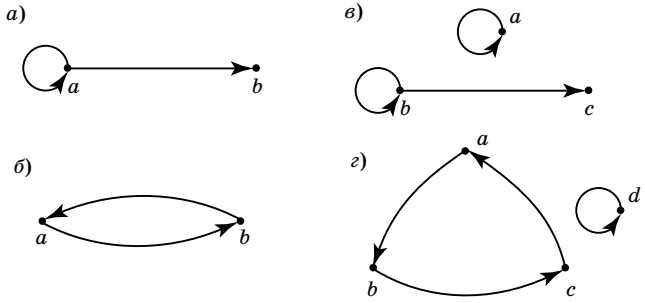
Вопросы

1. Верно ли высказывание: «Свойство отношения — это определенное условие, которому удовлетворяют элементы этого отношения»? Подтвердите ответ с помощью примера.
2. Дайте определение свойствам:
   – рефлексивности; – антирефлексивности;
   – симметричности; – асимметричности;
   – антисимметричности; – транзитивности;
   – антитранзитивности.
   Для каждого из перечисленных свойств объясните, чем граф и матрица отношений, обладающих этим свойством, отличаются от графа и матрицы отношений, не обладающих этим свойством.
3. Может ли отношение обладать не одним, а несколькими свойствами? Приведите примеры. Какие из известных вам свойств могут одновременно характеризовать отношение, а какие являются взаимоисключающими, т.е. не могут одновременно характеризовать одно отношение?

Задания

1. Определите, какими свойствами обладает каждое из приведенных отношений. Отношения заданы на множестве $A = \{1, 2, 3, 4\}$:
   а) {(2, 2), (2, 3), (2, 4), (3, 2), (3, 3), (3, 4)};
   б) {(1, 1), (1, 2), (2, 1), (2, 2), (3, 3), (4, 4)};
   в) {(2, 4), (4, 2)};
   г) {(1, 2), (2, 3), (3, 4)};
   д) {(1, 1), (2, 2), (3, 3), (4, 4)};
   е) {(1, 3), (1, 4), (2, 3), (2, 4), (3, 1), (3, 4)}.

2. Определите, какими свойствами обладает каждое из отношений, представленных следующими графами:



3. Пусть R_1 и R_2 — некоторые отношения. Заполните таблицу следующим образом. В ячейку таблицы поместите «+», если из того, что R_1 и R_2 обладают указанным свойством следует, что результат операции тоже обладает этим свойством, «−» — если при том, что R_1 и R_2 обладают указанным свойством, результат операции может не обладать этим свойством. Обоснуйте выбор каждого «+» и «−».

	$R_1 \cup R_2$	$R_1 \cap R_2$	$\overline{R}_1$	$R_1 \backslash R_2$	$R_1 \circ R_2$	R^n, $(n \in N)$	R^{-1}
Рефлексивность							
Антирефлексивность							
Симметричность							
Асимметричность							
Антисимметричность							
Транзитивность							
Антитранзитивность							

4. Определите, какими свойствами обладают следующие отношения, заданные на некотором множестве людей. Пусть $(a, b) \in R$, если:
   а) a выше ростом, чем b;
   б) a и b родились в один день;
   в) a является родственником b;
   г) a знаком с b.
5. Найдите ошибку в доказательстве следующего утверждения (это утверждение не верно).
   Утверждение: Если некоторое отношение R симметрично и транзитивно, то оно является также и рефлексивным.
   Доказательство: Возьмем пару элементов (a, b), такую, что $(a, b) \in R$, тогда и $(b, a) \in R$ (отношение симметрично). Если $(a, b) \in R$ и $(b, a) \in R$, то, по свойству транзитивности, $(a, a) \in R$, значит отношение рефлексивно.

6. Приведите пример отношения, которое
   а) является симметричным и антисимметричным;
   б) не является ни симметричным, ни антисимметричным.
7. Сколько существует отношений, заданных на множестве A, $|A| = n$, обладающих следующим свойством:
   а) симметричность;
   б) антисимметричность;
   в) асимметричность;
   г) антирефлексивность;
   д) симметричность и рефлексивность.
8. Пусть R — рефлексивное и транзитивное отношение. Верно ли, что $R^n = R$ для всех $n \in N$.
9. Пусть R_1 и R_2 — некоторые отношения, причем $R_1 \supseteq R_2$. Покажите, что если R_1 антисимметрично, то R_2 тоже антисимметрично.
10. Составьте алгоритм определения свойств отношения. Входным данным может быть матрица отношения или список элементов отношения.

2.4. Отношения эквивалентности, порядка, толерантности

Отношение эквивалентности, классы эквивалентности, путь в графе, частичный (нестрогий) порядок, строгий порядок, линейный порядок, сравнимые и несравнимые элементы, отношение толерантности

Рассмотрим некоторые наиболее часто используемые классы бинарных отношений.

Определение
Бинарное отношение, обладающее свойствами рефлексивности, симметричности и транзитивности, называется ***отношением эквивалентности*** (обозначается ~).

Пусть задано множество A и отношение эквивалентности, определенное на этом множестве: $R \subseteq A \times A$. Элементы a, $b \in A$, для которых выполняется aRb, называются ***эквивалентными***. Важное свойство любого отношения эквивалентности R, определенного на множестве A, заключается в том, что оно разбивает множество A на непересекающиеся подмножества, которые называются ***классами эквивалентности***. В случае конечного множества A разбиение его на классы эквивалентности происходит следующим образом. Пусть на множестве A задано отношение эквивалентности R. Выберем элемент $a_1 \in A$ и образуем класс C_1, состоящий из всех элементов $y \in A$, для которых выполняется отношение $a_1 R y$. Класс C_1

может состоять только из одного элемента a_1, если не существует других элементов y, таких, что $a_1 R y$. Заметим, что в силу рефлексивности отношения эквивалентности всегда выполняется $a_1 R a_1$. Если $C_1 \neq A$, то выберем из A элемент a_2, не вошедший в класс C_1, и образуем класс C_2, состоящий из элементов $y \in A$, для которых выполняется отношение $a_2 R y$. Если $(C_1 \cup C_2) \neq A$, то выберем из A элемент a_3, не вошедший в классы C_1 и C_2, и образуем класс C_3. Будем продолжать построение классов до тех пор, пока в A не останется ни одного элемента, не вошедшего в один из классов C_i. Получится система классов $C_1, C_2, \ldots, C_n$. Эта система классов называется системой классов эквивалентности и обладает следующими свойствами:

а) классы попарно не пересекаются;

б) любые два элемента из одного класса эквивалентны;

в) любые два элемента из разных классов не эквивалентны.

Примеры. Отношение равенства «=» на любом множестве чисел является отношением эквивалентности. Отношение «учиться в одном классе» на множестве учеников школы является отношением эквивалентности и разбивает все множество учеников школы на отдельные классы. Отношение «иметь одинаковое имя» на определенном множестве людей является отношением эквивалентности и разбивает все множество людей на классы эквивалентности — группы людей с одинаковыми именами.

Определение

Отношение называется ***отношением частичного порядка*** (обозначается $\leq$), если оно:

а) рефлексивно ($a \leq a$);

б) антисимметрично (если $a \leq b$ и $b \leq a$, то $a = b$);

в) транзитивно (если $a \leq b$ и $b \leq c$, то $a \leq c$).

Если на множестве задано отношение частичного порядка, то это множество называется ***частично упорядоченным***.

Существует способ наглядного представления конечного частично упорядоченного множества, который называют ***диаграммой Хассе***. Каждый элемент $a \in A$ обозначают точкой, и любую пару точек, соответствующих элементам $a \in A$ и $b \in A$, соединяют стрелкой, идущей из точки a в точку b, тогда и только тогда, когда $(a \leq b)$, $(a \neq b)$, и не существует $c \in A$ такого, что $a \leq c \leq b$, и элемент c отличается и от a, и от b.

Пример. Примером отношения частичного порядка является отношение включения множеств, справедливое для некоторых пар

подмножеств фиксированного множества. На рис. 2.15 изображен граф отношения включения на множестве 2^A всех подмножеств трехэлементного множества $A = \{a, b, c\}$.

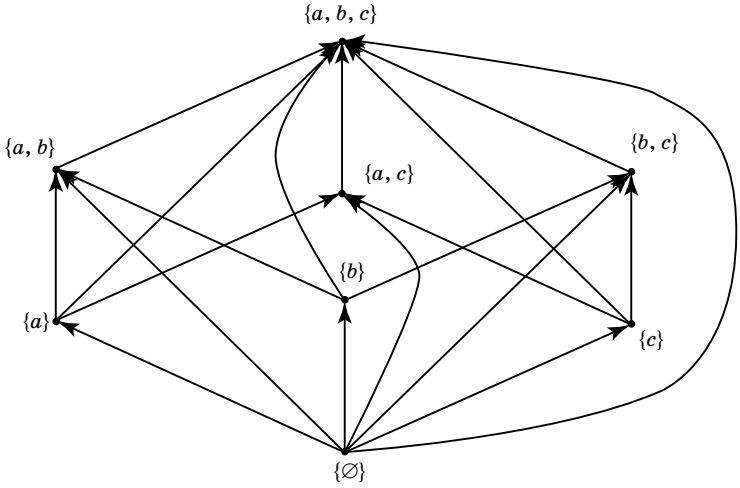


Рис. 2.15. Граф отношения включения множеств

Со свойством транзитивности связано понятие пути в графе.

Определение

Путь в графе отношения из вершины a в вершину b — это последовательность дуг (a, x_1), (x_1, x_2), (x_2, x_3),..., (x_{n-1}, b), $n \geq 1$. Число дуг n называется ***длиной пути***.

Используя понятие пути в графе, можно показать, что свойство транзитивности выполняется в отношении, заданном графом, если любые две вершины, между которыми существует путь, соединены дугой.

Часто при изображении графа отношения порядка опускают дуги, которые можно достроить, используя свойство транзитивности, то есть, если на графе существует путь из a в b, то дугу (a, b) не изображают. Кроме того, располагают вершины так, чтобы направление всех дуг было снизу-вверх. Тогда направление дуг можно не указывать. Полученная таким образом диаграмма достаточна для определения отношения порядка. Так, отношение включения множеств с графом на рис. 2.15 можно изобразить с помощью диаграммы Хассе, как показано на рис. 2.16.

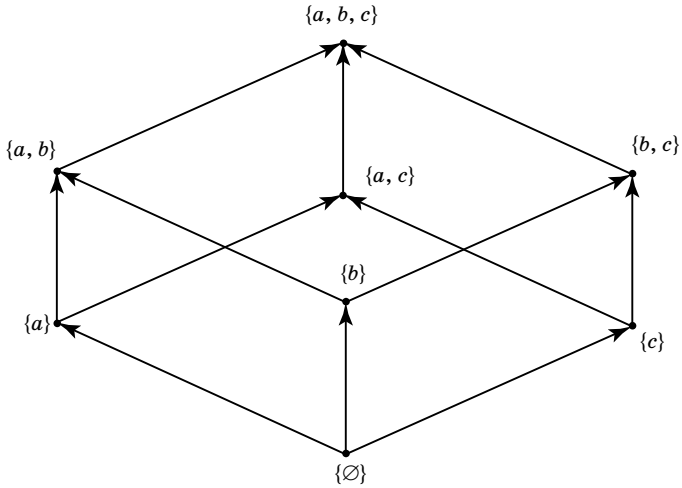


Рис. 2.16. Диаграмма Хассе отношения включения множеств

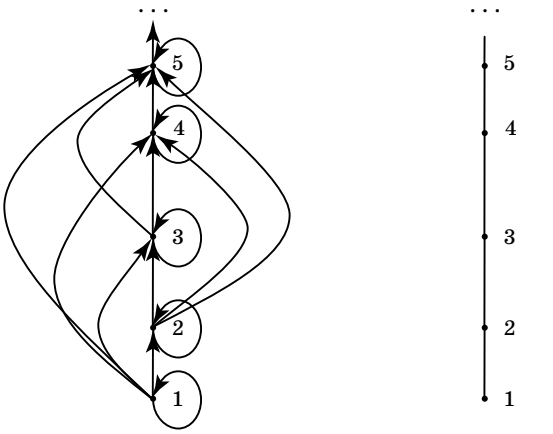
Из рассмотренного примера видно, что не все вершины графа отношения соединены между собой путями. Например, не существует пути между $\{a, b\}$ и $\{a, c\}$.

Определение

Элементы a и b называются ***сравнимыми*** в отношении частичного порядка R, если выполняется хотя бы одно из соотношений aRb или bRa. Множество A, на котором задано отношение частичного порядка R и для которого любые два элемента этого множества сравнимы, называется ***линейно упорядоченным*** или ***полностью упорядоченным***.

Полностью упорядоченное множество отличается от частично упорядоченного тем, что в частично упорядоченном множестве A могут присутствовать элементы, из которых можно составлять несравнимые пары.

Пример. Множество N натуральных чисел полностью упорядочено относительно естественной операции сравнения $\leq$, которая истинна на некотором подмножестве R пар (a, b) декартового квадрата N^2. Любые два элемента $a, b \in N$ сравнимы по R, так как хотя бы одно из неравенств $a \leq b$, $b \leq a$ истинно. На рис. 2.17 рассмотренное множество N задано с помощью графа отношений и диаграммы Хассе.



а) задание с помощью графа отношений

б) задание с помощью диаграммы Хассе

Рис. 2.17. Линейно упорядоченное множество натуральных чисел

Отношение частичного порядка, являющееся рефлексивным, антисимметричным и транзитивным, называется также ***отношением нестрогого порядка***. В отличие от него ***отношение строгого порядка*** (обозначается $<$) определяется свойствами:

а) антирефлексивность (если $a < b$, то $a \neq b$);

б) асимметричность (если $a < b$, то не верно $b < a$);

в) транзитивность (если $a < b$ и $b < c$, то $a < c$).

Таким образом, если в отношении нестрогого порядка свойство антисимметричности заменить асимметричностью, то получится строгий порядок.

Определение

Отношение называется ***отношением толерантности***, если оно: а) рефлексивно; б) симметрично и в) антитранзитивно.

Толерантность представляет собой формальное представление интуитивного понятия сходства. Сходство двух объектов не зависит от того, в каком порядке они сравниваются, в этом проявляется свойство симметричности. В то же время, если один объект сходен с другим, а другой сходен с третьим, то это не означает, что первый и третий объекты сходны, то есть свойство транзитивности может не выполняться.

Рассмотрим отношение толерантности на примере популярной задачи «превращения мухи в слона» как сходстве между

четырехбуквенными словами, если они отличаются одной буквой. Получим: муха — мура — тура — тара — кара — каре — кафе — кафр — каюр — каюк — крюк — крок — срок — сток — стон — слон.

Вопросы

1. Дайте определение отношения эквивалентности.
2. Каким образом происходит разбиение множества на классы эквивалентности? Какие элементы называются эквивалентными?
3. Какие свойства характерны для классов эквивалентности?
4. Приведите примеры отношений эквивалентности и получаемых при этом классов эквивалентности.
5. Дайте определение отношения частичного порядка. Как оно обозначается? Приведите пример.
6. Что такое путь в графе? Как понятие пути применяется для исследования графов отношения порядка?
7. Какое множество называется линейно упорядоченным? Что такое сравнимые и несравнимые элементы?
8. Чем отличается отношение полного порядка от отношения частичного порядка? Дайте определение отношения строгого порядка.
9. Дайте определение отношения толерантности. Приведите пример.

Задания

1. Докажите, что любое отношение эквивалентности на множестве A разбивает это множество на непересекающиеся подмножества — классы эквивалентности.
2. Какие из следующих наборов подмножеств множества $A = \{1, 2, 3, 4, 5, 6\}$ могут являться разбиениями этого множества на классы эквивалентности?
   а) $\{1, 2\}, \{2, 3, 4\}, \{4, 5, 6\}$; в) $\{2, 4, 6\}, \{1, 3, 5\}$;
   б) $\{1\}, \{2, 3, 6\}, \{4, 5\}$; г) $\{1, 4, 5\}, \{2, 6\}$.
3. Пусть R_1 и R_2 — отношения эквивалентности. Определите, являются ли следующие отношения отношениями эквивалентности:
   а) $R_1 \cup R_2$; б) $R_1 \cap R_2$; в) $\overline{R}_1$.
4. Пусть R_1 и R_2 — отношения частичного порядка. Определите, являются ли следующие отношения отношениями частичного порядка:
   а) $R_1 \cup R_2$; б) $R_1 \cap R_2$; в) $\overline{R}_1$.
5. Определите, какие отношения эквивалентности разбивают множество на наибольшее и наименьшее число классов эквивалентности.

6. Пусть R — отношение частичного порядка. Докажите, что R^{-1} — тоже отношение частичного порядка.

7. Какие из следующих матриц задают отношение частичного порядка? Для нахождения ответа исследуйте графы отношений.

	a	*b*	*c*	*d*
a	1	0	1	0
b	0	1	1	0
c	0	0	1	1
d	1	1	0	1

	a	*b*	*c*
a	1	0	1
b	1	1	0
c	0	0	1

	a	*b*	*c*
a	1	0	0
b	0	1	0
c	1	0	1

8. Какие из следующих матриц задают отношение эквивалентности? Для нахождения ответа исследуйте графы отношений.

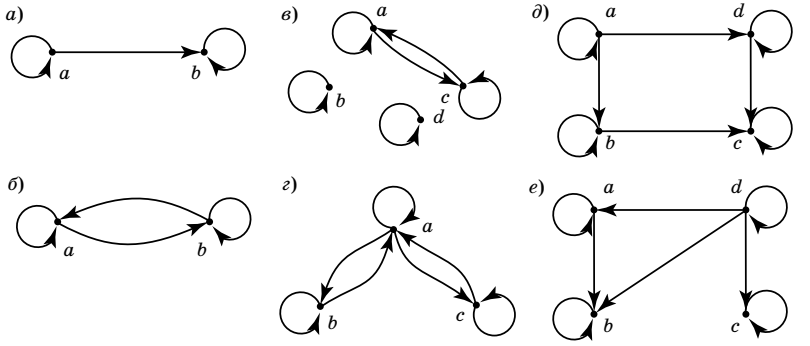
	a	*b*	*c*	*d*
a	1	0	1	0
b	0	1	0	0
c	1	0	1	0
d	0	1	0	1

	a	*b*	*c*
a	1	1	1
b	0	1	1
c	1	1	1

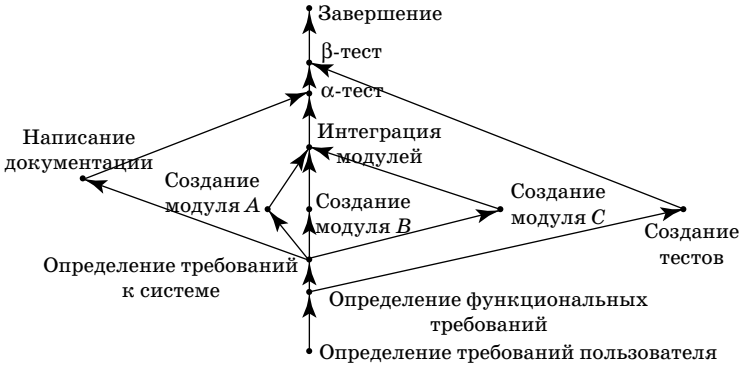
	a	*b*	*c*	*d*
a	1	1	1	0
b	1	1	1	0
c	1	1	1	0
d	0	0	0	1

9. Какие из следующих отношений на множестве людей являются отношениями толерантности, какие — отношениями эквивалентности?
   а) a и b одинакового возраста;
   б) a и b имеют общих родителей;
   в) a и b знакомы;
   г) a и b разговаривают на одном языке.

10. Какие из следующих отношений, заданных графами, являются отношениями порядка, какие — отношениями эквивалентности?



11. Составьте возможную последовательность выполнения задач по созданию некоторого программного продукта, если диаграмма, задающая частичный порядок на множестве задач, следующая:



12. Составьте алгоритм, определяющий по заданной матрице отношения, является ли отношение частичным или строгим порядком, эквивалентностью или толерантностью.

2.5. Функциональные отношения

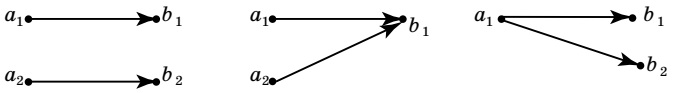
Функциональное отношение, области определения и значений, отображение, сюръекция, инъекция, биекция

Функциональная зависимость между переменными или функция является частным случаем отношения.

Определение

Отношение R между множествами X и Y ($R \subseteq X \times Y$) является ***функциональным***, если все его элементы (упорядоченные пары) различны по первому элементу: каждому $x \in X$ либо соответствует только один элемент $y \in Y$, такой, что xRy, либо такого элемента y вообще не существует.

Матрица функционального отношения, заданного на конечных множествах X и Y, содержит не более одной единицы в каждой строке. Если функциональное отношение задано в виде графа, то из каждой вершины, изображающей первую координату, выходит не более одной дуги. Примеры функционального и нефункционального отношений представлены на рис. 2.18.



а) функциональное отношение

б) функциональное отношение

в) нефункциональное отношение

Рис. 2.18. Примеры функционального и нефункционального отношений

Пример. Пусть имеется множество A кроликов и множество B клеток. Пусть R — отношение размещения кроликов по клеткам — «Кролики — Клетки». В таком случае, как бы мы ни размещали кроликов по клеткам, в любом случае отношение R будет функциональным, поскольку каждому кролику может соответствовать только одна клетка. Обозначим кроликов буквами, а клетки — номерами. Пусть $A = \{a, b\}$, $B = \{1, 2, 3\}$. На рис. 2.19 изображены функциональные отношения $R_1 = \{(a, 1), (b, 3)\}$ и $R_2 = \{(a, 1), (b, 1)\}$.

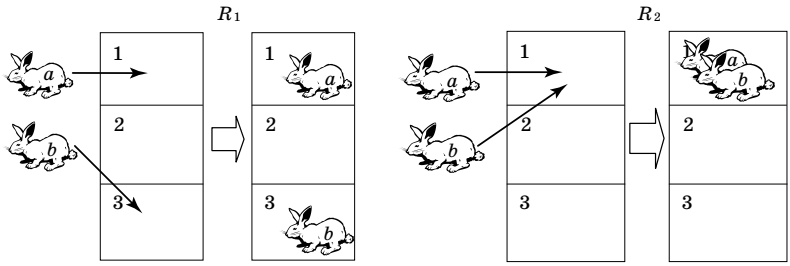


Рис. 2.19. Примеры функциональных отношений «Кролики — Клетки»

Примером нефункционального отношения является отношение $R_3 = \{(a, 1), (a, 2), (b, 3)\}$. Оно может быть представлено в терминах размещения кроликов по клеткам, как показано на рис. 2.20, где кролик a занимает одновременно две клетки 1, 2, благодаря сломанной перегородке.

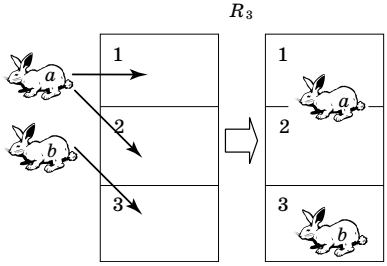


Рис. 2.20. Пример отношения «Кролики — Клетки», не являющегося функцией

Определение

Пусть R — некоторое отношение, $R \subseteq X \times Y$. ***Областью определения*** отношения R называется множество D_R, состоящее из всех элементов множества X, которые связаны отношением R

с элементами множества Y: $D_R \subseteq X$, $D_R = \{x: \exists\, y \in Y, (x, y) \in R\}$. ***Областью значений*** отношения R называется множество $\Re_R$, состоящее из всех элементов множества Y, которые связаны отношением R с элементами множества X: $\Re_R \subseteq Y$, $\Re_R = \{y: \exists\, x \in X, (x, y) \in R\}$.

Определение

Пусть F — функциональное отношение, $F \subseteq X \times Y$. Соответствие $x \to y$ от первого ко второму элементу каждой пары $(x, y) \in F$ отношения F называется ***функцией*** f или ***отображением*** f ***множества*** D_F в Y и обозначается как $f: D_F \to Y$, либо как $D_F \xrightarrow{f} Y$.

Множество D_F называется областью определения или задания функции (отображения) f и обозначается как $D_f (\equiv D_F)$. Говорят также, что функция f действует из X в Y и определена на подмножестве D_f из X. Если $D_f = X (= D_F)$, то пишут $f: X \to Y$ и говорят, что задано отображение $X \to Y$.

Если множество $A \subseteq X$, то через $f(A) = \{y \in Y: y = f(x), \forall x \in A\}$ обозначается образ множества A. Множество $f(X) \subseteq Y$ называется ***образом*** или ***областью значений*** отображения f и обозначается через $\Re_f = f(X)$.

Если множество $B \subseteq Y$, то множество $f^{-1}(B) = \{x \in X: f(x) \in B\}$ называется ***прообразом*** множества B относительно отображения f.

Запись $y = f(x)$ функциональной зависимости переменной y от переменной x (называемой ***аргументом функции***) менее информативна, чем запись в виде отображения $f: D_f \to Y$, так как должна дополняться информацией об области определения — подмножестве допустимых значений аргумента x.

Графиком функции (отображения) $f: X \to Y$ называется совокупность «двумерных» точек (x, y) вида $(x, f(x))$ в декартовом произведении $X \times Y$. Именно в этом смысле синусоида является графиком функции $\sin x$, парабола (охватывающая ось y) — графиком квадратичной функции x^2 и т.д. Таким образом, если $F \subset X \times Y$ — исходное функциональное отношение, порождающее фунцию (отображение) f, то F в точности есть ***график функции*** f. Не следует смешивать понятия «график функции f» и «граф отношения F»: граф с помощью дуг со стрелками описывает действие отображения f на каждом значении аргумента x (рис. 2.21).

Виды отображений

1. Функция $f\colon X \to Y$ называется ***сюръективным отображением***, если $\Re_f = Y$.

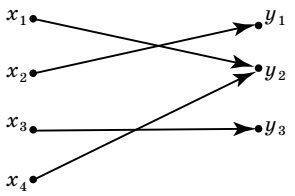


Рис. 2.21. Пример сюръективного отображения

На графе, представляющем сюръективное отображение $X \to Y$, из любой вершины $x \in X$ выходит в точности одна дуга, а в любую вершину, представляющую элемент множества Y, входит не менее одной дуги (рис. 2.21).

2. Функция $f\colon X \to Y$ называется ***инъективным отображением***, если из $x_1 \neq x_2$ следует $f(x_1) \neq f(x_2)$.

На графе, представляющем инъективное отображение $X \to Y$ из любой вершины $x \in X$ выходит в точности одна дуга, а в любую вершину, представляющую элемент множества Y, входит не более одной дуги (рис. 2.22).

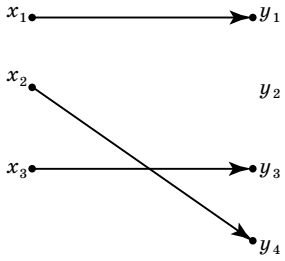


Рис. 2.22. Пример инъективного отображения

Если $f\colon X \to Y$ — инъективное отображение и $F = \{(x, f(x)), \forall x \in X\}$ — соответствующее функциональное отношение ($F \subset X \times Y$), то обратное отношение $F^{-1} = \{(y, x), \forall y \in \Re_f\}$ также является функциональным. Задаваемая отношением F^{-1} функция $x = f^{-1}(y)$, $f^{-1}\colon \Re_f \to X$ обладает свойствами: $f^{-1}(f(x)) = x$, $\forall x \in X$; $f(f^{-1}(y)) = y$, $\forall y \in \Re_f$ и называется ***обратной*** к функции f.

3. Функция $f\colon X \to Y$ называется ***биективным отображением***, если она сюръективна и инъективна.

На графе, представляющем биективное отображение $X \to Y$ конечных множеств, из любой вершины $x \in X$ выходит в точности одна дуга, а в любую вершину $y \in Y$ входит одна и только одна дуга (рис. 2.23). Таким образом,

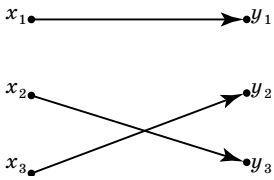


Рис. 2.23. Пример биективного отображения

биективное отображение $f: X \to Y$ осуществляет взаимно однозначное отображение между множествами X и Y, поэтому $X \sim Y$, $|X| = |Y|$ (см. п. 1.6).

Свойство биективности обеспечивает существование обратного отображения. Это означает, что если $f: X \to Y$ биективно, то существует обратное отображение $f^{-1}: Y \to X$, причем $D_{f^{-1}} = Y$.

Проиллюстрируем общее понятие функции и сюръективные, инъективные и биективные отображения на примерах функциональных отношения типа «Кролики — Клетки». Отношения R_1, R_2 на рис. 2.19 определяют отображения $f_1: A \to B$, $f_2: A \to B$ соответственно, причем f_1 инъективно, f_2 — нет. Оба отображения не являются сюръективными, так как после размещения кроликов остаются свободные клетки. На схеме, представленной на рис. 2.24, множество из шести кроликов $X = \{a, b, c, d, e, f\}$ размещается в множестве клеток $Y = \{1, 2, 3, 4\}$.

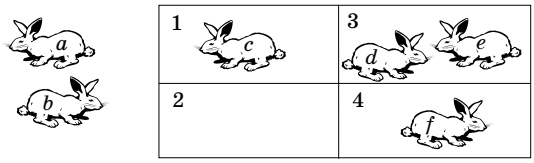


Рис. 2.24. Функция «Кролики — Клетки», не являющаяся отображением $X \to Y$

Эта схема размещения соответствует отношению R, граф которого представлен на рис. 2.25.

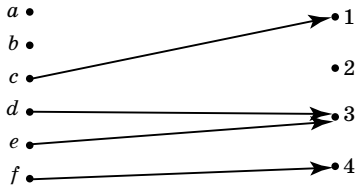


Рис. 2.25. Граф отношения $R = \{(c, 1), (d, 3), (e, 3), (f, 4)\}$

Это отношение R не является функциональным, но определяемая им функция f определена не на всем множестве X, так как не все кролики размещены в клетках. Областью определения функции f является множество $D_R = \{c, d, e, f\}$ и областью значений $\Re_R = \{1, 3, 4\}$.

Рассмотрим два других варианта размещения шести кроликов X по четырем клеткам Y, указанных на рис. 2.26 и 2.27.

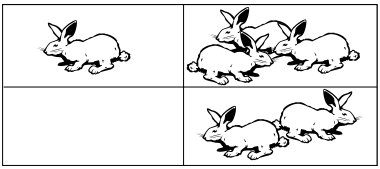


Рис. 2.26. Отображение «Кролики — Клетки»

R является сюръективным отображением, если все клетки заняты.

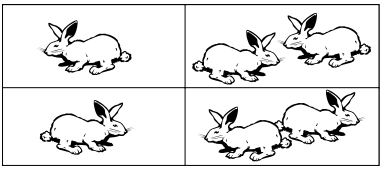


Рис. 2.27. Сюръективное отображение «Кролики — Клетки»; $|X| = 6, |Y| = 4$

Для реализации инъективного размещения шести кроликов ($|X| = 6$) необходимо увеличить число клеток так, чтобы оно было не меньше числа кроликов. Соответствующее отображение $f: X \to Y$ будет являться инъективным отображением, если в каждой клетке помещено не более одного кролика, но могут оставаться и пустые клетки, как показано на рис. 2.28.

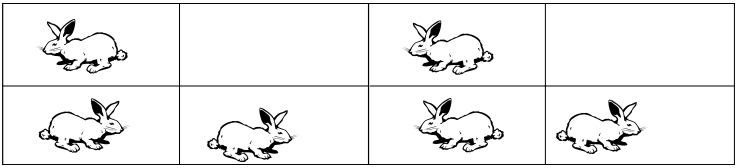


Рис. 2.28. Инъективное отображение «Кролики — Клетки»; $|X| = 6, |Y| = 8$

R является биективным отображением, если в каждой клетке находится точно по одному кролику (рис. 2.29).

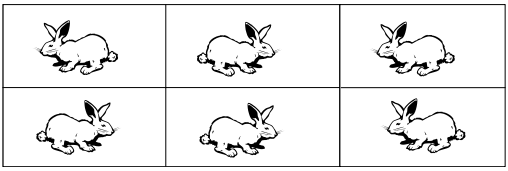


Рис. 2.29. Биективное отображение «Кролики — Клетки»; $|X| = 6, |Y| = 6$

Вопросы

1. Какое отношение называется функциональным?
2. Что такое область определения и область значения отношения?
3. Какое отношение $R \subset X \times Y$ задает отображение $X \to Y$?
4. Дайте определение сюръективного, инъективного и биективного отображений. Чем характеризуется граф каждого из этих отображений?
5. Какой вид отображения можно назвать «много к одному», какой — «один к одному»?
6. Какой вид отображения множества A на множество B ставит в соответствие каждому элементу множества A один и только один элемент множества B так, что каждый элемент множества B поставлен в соответствие одному и только одному элементу множества A?

Задания

1. Какие из следующих отношений $R \subset A^2$, заданных на множестве $A = \{-20, -19, ..., 0, 1, ..., 19, 20\}$, являются функциональными? Для функциональных отношений укажите соответствующие функции:
   а) $(a, b) \in R$, если $a = b^2$;
   б) $(a, b) \in R$, если $a^2 = b$;
   в) $(a, b) \in R$, если $a \le b$;
   г) $(a, b) \in R$, если $ab = 6$;
   д) $(a, b) \in R$, если $a = b^3$;
   е) $(a, b) \in R$, если $b = a^3$;
   ж) $(a, b) \in R$, если $a = b^4$;
   з) $(a, b) \in R$, если $a = 1/b$.
2. Какие из функций f, найденных при решении задачи 1, являются отображениями вида $f: A \to A$?
3. Пусть заданы множества A, B, C и отношения $R \subseteq A \times B$ и $S \subseteq B \times C$. Покажите на примерах, что следующие выводы не верны:

а) если R — функциональное отношение, то $S \circ R$ — функциональное отношение;

б) если R задает отображение $A \to B$, то $S \circ R$ — отображение $A \to C$;

в) если S определяет сюръекцию $B \to C$, то $S \circ R$ — сюръекцию $A \to C$;

г) если S определяет инъекцию $B \to C$, то $S \circ R$ — инъекцию $A \to C$.

4. Пусть заданы множества A, B, C и отношения $R \subseteq A \times B$ и $S \subseteq B \times C$. Докажите, что

   а) если R и S — функциональные отношения, то $S \circ R$ — функциональное отношение;

   б) если R и S определяют отображения $A \to B$ и $B \to C$ соответственно, то $S \circ R$ определяет отображение $A \to C$;

   в) если R и S определяют сюръекции, то $S \circ R$ — также сюръекцию;

   г) если R и S определяют инъекции, то $S \circ R$ — также инъекцию;

   д) если R и S определяют биекции, то $S \circ R$ — также биекцию.

5. Докажите, что если f — инъективная функция, то существует функция f^{-1}.

6. Пусть $F \subset X \times Y$ определяет сюръективную функцию. Определяет ли F^{-1} отображение $Y \to X$?

7. Найдите область определения и область значений следующих функций:

   а) функция, которая каждому неотрицательному целому числу x, $x \le n$, ставит в соответствие его последнюю цифру;

   б) функция, которая строке битов длиной $\le n$ ставит в соответствие количество единиц в этой строке;

   в) функция, которая строке битов длиной $\le n$ ставит в соответствие количество оставшихся битов при разбиении этой строки на байты;

   г) функция, которая для целого положительного числа x, $x \le n$, находит наибольший квадрат, не превосходящий это число.

8. Пусть A и B — конечные множества, содержащие соответственно n и m элементов: $|A| = n$, $|B| = m$. Определите соотношение между n и m, если:

   а) существует инъективное отображение из A в B;

   б) существует сюръективное отображение из A в B;

   в) существует биективное отображение из A в B.

2.6. Реляционная модель данных

Кортеж, домен, атрибут. Реляционная алгебра и операции в ней: объединение, пересечение, разность, прямое произведение, ограничение, проекция, естественное соединение, деление

Одна из наиболее важных областей применения компьютеров в настоящее время — это хранение и обработка большого объема информации, имеющей сложную внутреннюю структуру. Эта информация должна быть представлена в памяти ЭВМ так, чтобы ее было легко извлекать, пополнять и преобразовывать. Мы рассмотрим один из методов построения баз данных, разработанный в конце 60-х годов и широко используемый в настоящее время. Поскольку представление информации в виде таблиц является наиболее удобным и привычным для человека, оно было взято за основу и показано, что любое представление данных сводится к совокупности двумерных таблиц. С математической точки зрения табличное представление данных легко формулируется в терминах отношений, и поэтому к нему применим аппарат теории множеств. Такая модель данных называется ***реляционной*** (relation — отношение). Для работы с ***реляционной моделью*** была создана ***реляционная алгебра***. Каждая операция этой алгебры использует одну или несколько таблиц (отношений) в качестве ее операндов и продуцирует в результате новую таблицу, т.е. позволяет «разрезать» и «склеивать» таблицы. Основная идея реляционной алгебры: поскольку отношения являются множествами, то средства манипулирования отношениями могут базироваться на таких традиционных теоретико-множественных операциях, как объединение, пересечение, разность, декартово произведение, дополненных некоторыми специальными операциями.

Рассмотрим примеры реляционного представления данных. Пусть имеется информация о студентах университета, представленная в таблице 2.1.

Таблица 2.1

Фамилия	Инициалы	Группа
Алексеев	И. А.	ПО-01
Андреева	О. П.	ПМ-01
Баранов	Н. П.	ПМ-01
Быкова	Н. А.	ПО-01
Волков	В. В.	ПМ-01

Эта информация представляет собой некоторое отношение R_1, заданное на трех множествах — множестве фамилий, множестве

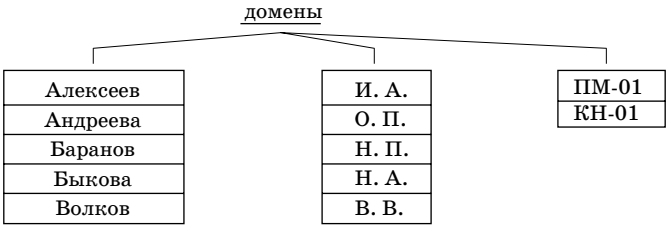
инициалов и множестве групп. Отношение R_1 можно задать списком его элементов:

R_1 = {(Алексеев, И. А., ПО-01), (Андреева, О. П., ПМ-01), ..., (Волков, В. В., ПМ-01)}.

Рассмотрим терминологию, применяемую при построении подобных таблиц данных. Элементы отношения, как (Алексеев, И. А., ПО-01) и (Андреева, О. П., ПМ-01), соответствующие строкам таблицы, называются ***кортежами***. Множества или области данных, на которых определено отношение, такие, как множество фамилий, множество инициалов и множество групп, соответствующие столбцам таблицы, называются ***доменами***. Наименования столбцов таблицы называют ***атрибутами***. Отношению присваивают имя, например, СТУДЕНТ 1. ***Схемой отношения*** является список атрибутов, например, схемой отношения СТУДЕНТ 1 является список: (Фамилия, Инициалы, Группа). Представим это отношение на рис. 2.30.

Для изменения содержимого таких таблиц в базах данных используют операции добавления, удаления кортежей и изменения значения атрибутов. Это простые операции по заполнению таблицы.

Рассмотрим теоретико-множественные операции, применяемые для преобразования отношений в базе данных, используя термины реляционной алгебры. Отношения, к которым применяется операция, будем называть отношениями-***операндами***.



СТУДЕНТ 1

Фамилия	Инициалы	Группа
Алексеев	И. А.	КН-01
Андреева	О. П.	ПМ-01
Баранов	Н. П.	ПМ-01
Быкова	Н. А.	КН-01
Волков	В. В.	ПМ-01

атрибуты

кортежи

Рис. 2.30. Отношение СТУДЕНТ 1

Теоретико-множественные операции реляционной алгебры

Объединение отношений. При выполнении операции объединения двух отношений (∪) получаем отношение, включающее все кортежи, входящие хотя бы в одно из отношений-операндов.

Пересечение отношений. При выполнении операции пересечения двух отношений (∩) получаем отношение, включающее только те кортежи, которые входят в оба отношения-операнда.

Разность отношений. Отношение, являющееся разностью (\) двух отношений включает все кортежи, входящие в отношение — первый операнд, такие, что ни один из них не входит в отношение, являющееся вторым операндом.

Эти операции имеют смысл для отношений, определенных на одинаковых доменах. Рассмотрим отношения СТУДЕНТ 1 и СТУДЕНТ 2 (рис. 2.31).

СТУДЕНТ 1

Фамилия	Инициалы	Группа
Алексеев	И. А.	КН-01
Андреева	О. П.	ПМ-01
Баранов	Н. П.	ПМ-01
Быкова	Н. А.	КН-01
Волков	В. В.	ПМ-01

СТУДЕНТ 2

Фамилия	Инициалы	Группа
Алексеев	И. А.	КН-01
Быкова	Н. А.	КН-01
Дроздов	И. К.	КН-01
Зайцев	О. Н.	ПМ-01
Кузнецова	Е. В.	КН-01

Рис. 2.31. Отношения СТУДЕНТ 1 и СТУДЕНТ 2

Применив операции ∩, ∪ и \ к отношениям СТУДЕНТ 1 и СТУДЕНТ 2, получим следующие результаты (рис. 2.32):

СТУДЕНТ 1 ∩ СТУДЕНТ 2

Фамилия	Инициалы	Группа
Алексеев	И. А.	КН-01
Быкова	Н. А.	КН-01

ВСЕ СТУДЕНТЫ = СТУДЕНТ 1 ∪ СТУДЕНТ 2

ВСЕ СТУДЕНТЫ

Фамилия	Инициалы	Группа
Алексеев	И. А.	КН-01
Андреева	О. П.	ПМ-01
Баранов	Н. П.	ПМ-01
Быкова	Н. А.	КН-01
Волков	В. В.	ПМ-01
Дроздов	И. К.	КН-01
Зайцев	О. Н.	ПМ-01
Кузнецова	Е. В.	КН-01

СТУДЕНТ 1 \ СТУДЕНТ 2

Фамилия	Инициалы	Группа
Андреева	О. П.	ПМ-01
Баранов	Н. П.	ПМ-01
Волков	В. В.	ПМ-01

Рис. 2.32. Результат выполнения операций ∩, ∪, \

Схематически выполнение операций ∩, ∪, \ изображено на рис. 2.33.

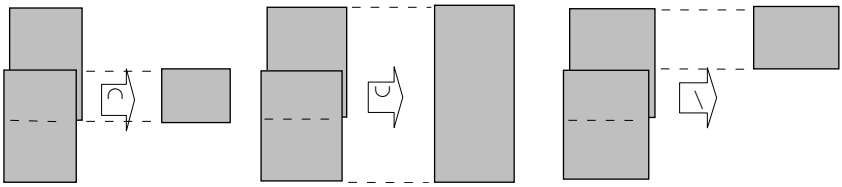


Рис. 2.33. Схема выполнения операций ∩, ∪, \

Рассмотрим операцию, которая представляет собой декартово или прямое произведение отношений.

Прямое произведение отношений. При выполнении прямого произведения (×) двух отношений получаем отношение, множество кортежей которого является декартовым произведением множеств кортежей первого и второго операндов. Рассмотрим отношения СТУДЕНТ 1 (рис. 2.30) и КУРС (рис. 2.34).

КУРС

Уч. год	Курс
2001–2002	1
2002–2003	2

Рис. 2.34. Отношение КУРС

Выполним прямое произведение отношения СТУДЕНТ 1 на отношение КУРС (рис. 2.35).

СТУДЕНТ 1 × КУРС

Фамилия	Инициалы	Группа	Уч. год	Курс
Алексеев	И. А.	КН-01	2001–2002	1
Алексеев	И. А.	КН-01	2002–2003	2
Андреева	О. П.	ПМ-01	2001–2002	1
Андреева	О. П.	ПМ-01	2002–2003	2
Баранов	Н. П.	ПМ-01	2001–2002	1
Баранов	Н. П.	ПМ-01	2002–2003	2
Быкова	Н. А.	КН-01	2001–2002	1
Быкова	Н. А.	КН-01	2002–2003	2
Волков	В. В.	ПМ-01	2001–2002	1
Волков	В. В.	ПМ-01	2002–2003	2

Рис. 2.35. Результат выполнения прямого произведения отношений СТУДЕНТ 1 и КУРС

Схематически выполнение операции × представлено на рис. 2.36.

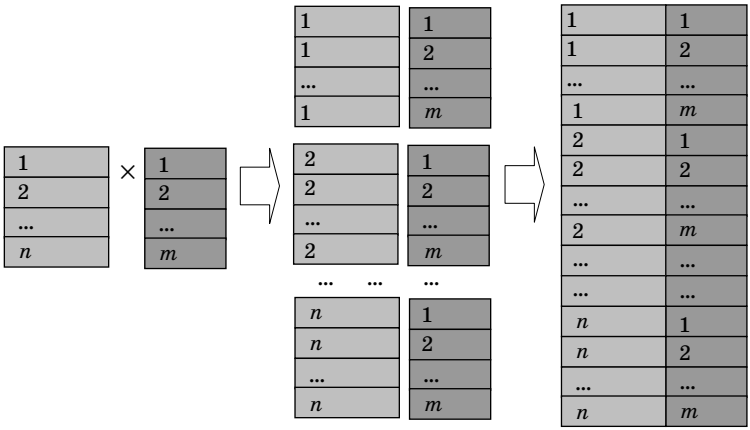


Рис. 2.36. Схема выполнения операции прямого произведения отношений

Теперь перейдем к рассмотрению специальных операций, применяемых в базах данных.

Специальные операции реляционной алгебры

Ограничение отношения. Результатом ограничения (σ) отношения по некоторому атрибуту или атрибутам σ является отношение, состоящее в точности из тех кортежей, которые удовлетворяют условию σ.

Выполним ограничение отношения ВСЕ СТУДЕНТЫ (рис. 2.32) по атрибуту Группа = КН-01. Назовем результат — СТУДЕНТ КН-01. В результате получим отношение, содержащее только кортежи, в которых значение атрибута Группа равняется КН-01 (рис. 2.37).

$$\sigma_{Группа = КН\text{-}01}\,(ВСЕ\ СТУДЕНТЫ) = СТУДЕНТ\ КН\text{-}01$$

СТУДЕНТ КН-01

Фамилия	Инициалы	Группа
Алексеев	И. А.	КН-01
Быкова	Н. А.	КН-01
Дроздов	И. К.	КН-01
Кузнецова	Е. В.	КН-01

Рис. 2.37. Ограничение σ по значению КН-01 атрибута Группа

Проекция отношения. При выполнении проекции отношения (π) на заданный набор его атрибутов отношение-результат получается путем удаления из отношения-операнда атрибутов, не указанных в заданном наборе.

Выполним проекцию отношения СТУДЕНТ 1 КУРС по атрибутам Группа, Уч. год, Курс (рис. 2.38).

$$\pi_{Группа,\ Уч.\ год,\ Курс}\,(СТУДЕНТ\ 1\ КУРС)$$

Группа	Уч. год	Курс
КН-01	2001–2002	1
КН-01	2002–2003	2
ПМ-01	2001–2002	1
ПМ-01	2002–2003	2

Рис. 2.38. Проекция по атрибутам Группа, Уч. год, Курс

Схематически выполнение операции π представлено на рис. 2.39.

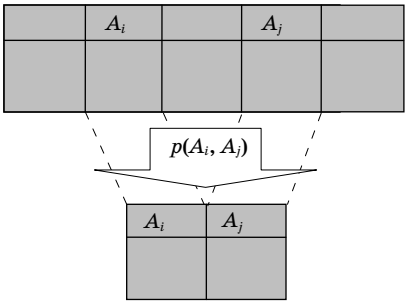


Рис. 2.39. Схема выполнения операции проекции по A_i, A_j

Естественное соединение отношений. При естественном соединении двух отношений (I⋈I) образуется результирующее отношение, кортежи которого являются соединением кортежей первого и второго отношений, если значение общих атрибутов совпадает.

Схематически выполнение операции I⋈I представлено на рис. 2.40.

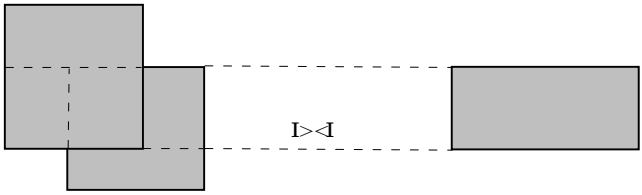


Рис. 2.40. Схема выполнения операции соединения отношений (I⋈I)

Рассмотрим отношение НОМЕР (рис. 2.41).

НОМЕР

Фамилия	Инициалы	Зачетка №	Студ №
Алексеев	И. А.	11197	784567
Андреева	О. П.	11215	784565
Баранов	Н. П.	11213	784598
Быкова	Н. А.	11216	784588
Волков	В. В.	11217	784599
Дроздов	И. К.	11193	784611
Зайцев	О. Н.	11191	784601
Кузнецова	Е. В.	11195	785587

Рис. 2.41. Отношение НОМЕР

Выполним естественное соединение отношений СТУДЕНТ КН-01 и НОМЕР (рис. 2.42).

СТУДЕНТ КН-01 I⋈I *НОМЕР*

Фамилия	Инициалы	Группа	Зачетка №	Студ №
Алексеев	И. А.	КН-01	11197	784567
Быкова	Н. А.	КН-01	11216	784588
Дроздов	И. К.	КН-01	11193	784611
Кузнецова	Е. В.	КН-01	11195	785587

Рис. 2.42. Естественное соединение СТУДЕНТ КН-01 и НОМЕР

Деление отношений. Операция деления отношений (÷) происходит следующим образом. Отношение — делитель должно иметь набор атрибутов, включенный в набор атрибутов делимого. Результирующее отношение содержит те атрибуты делимого, которые не присутствуют в делителе. Значения этих атрибутов берутся из тех кортежей делимого, которые включают в себя кортежи делителя.

Схематически выполнение операции деления отношений представлено на рис. 2.43.

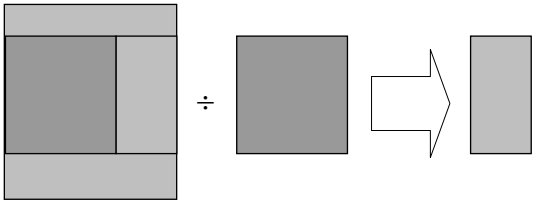


Рис. 2.43. Схема выполнения операции деления отношений (÷)

Выполним проекцию отношения СТУДЕНТ КН-01 по атрибутам Фамилия, Инициалы (рис. 2.44).

$$ФАМИЛИЯ\ СТ\ КН\text{-}01 = \pi_{Фамилия,\ Инициалы}(СТУДЕНТ\ КН\text{-}01)$$

ФАМИЛИЯ СТ КН-01

Фамилия	Инициалы
Алексеев	И. А.
Быкова	Н. А.
Дроздов	И. К.
Кузнецова	Е. В.

Рис. 2.44. Проекция СТУДЕНТ КН-01 по атрибутам Фамилия, Инициалы

Теперь произведем деление отношения НОМЕР (рис. 2.41) на отношение ФАМИЛИЯ СТ КН-01 (рис. 2.44), используя схему рис. 2.43.

В результате деления мы получили таблицу номеров студентов группы КН-01 (рис. 2.45).

НОМЕР ÷ ФАМИЛИЯ СТ КН-01

Зачетка №	Студ №
11197	784567
11216	784588
11193	784611
11195	785587

Рис. 2.45. Деление НОМЕР на ФАМИЛИЯ СТ КН-01

Кроме рассмотренных нами операций реляционной алгебры, в набор операций базы данных обычно включается операция присваивания, позволяющая сохранить результаты вычисления алгебраических выражений, операция переименования атрибутов, дающая возможность корректно сформировать заголовок (схему) результирующего отношения и другие полезные операции.

Вопросы

1. Что представляет собой реляционный метод построения баз данных?
2. Что называется кортежем, атрибутом, доменом? Приведите пример.
3. Перечислите теоретико-множественные операции, применяемые при работе с реляционными базами данных. Объясните принцип их выполнения.
4. Перечислите специальные реляционные операции, которые вам известны. Объясните принцип выполнения каждой из них.
5. Как реализованы рассмотренные в этом разделе операции над отношениями в СУБД, с которой вы работаете?

Задания

Вернемся к отношению, которое рассматривалось в п. 1 и было составлено для некоторой дизайнерской фирмы в г. Харькове.

Фирма-продукция-1

Название	Город	Продукция
ЧПП «Orion»	Одесса	мебель
ООО «День»	Харьков	светильники
ЧКП «Sit»	Одесса	мебель
ЧКП «Sit»	Одесса	светильники
ООО «House»	Харьков	светильники
ООО «House»	Харьков	материалы для ремонта

Допустим, существует еще одно отношение, составленное некоторой фирмой в г. Киеве.

Фирма-продукция-2

Название	Город	Продукция
ООО «Днепр»	Киев	материалы для ремонта
ЧКП «Virta»	Киев	сантехника
ЧКП «Virta»	Киев	плитка
ЧПП «Orion»	Одесса	мебель
ЧКП «София»	Киев	сантехника
ЧКП «София»	Киев	плитка
ООО «День»	Харьков	светильники

1. Выполните операции объединения, пересечения и разности над отношениями Фирма-продукция-1 и Фирма-продукция-2. Какой смысл может иметь применение этих операций? Назовите таблицу, полученную в результате выполнения операции объединения, Фирма-продукция-3.
2. Выполните соединение отношений Фирма-продукция-3 и Фирма-телефон. Отношение Фирма-телефон следующее:

Фирма-телефон

Название	Телефон
ЧПП «Orion»	784-556, 784-557
ООО «День»	145-134
ЧКП «Sit»	356-777
ООО «House»	122-890, 126-713
ООО «Днепр»	555-78-12, 555-77-09
ЧКП «Virta»	233-77-77, 233-77-78
ЧКП «София»	550-09-09

Назовите отношение, полученное в результате операции соединения Фирма-продукция-телефон.

3. Выберите два варианта выполнения операции проекции отношения Фирма-продукция-телефон. По каким атрибутам вы производили проекции?
4. Выполните ограничение отношения Фирма-продукция-телефон по условиям:
   а) Продукция = мебель;
   б) Продукция = материалы для ремонта, Город = Харьков;
   в) Город = Одесса.
5. Как изменяется число кортежей и (или) атрибутов отношения при выполнении изученных нами операций. Рассмотрите каждую операцию отдельно.

раздел 3

Алгебраические структуры

3.1. Алгебраические операции и их свойства

Унарная операция, бинарная операция, n-арная операция, операнд, записи $infix$, $prefix$, $postfix$, таблица Кэли, коммутативность, ассоциативность, дистрибутивность, единица, обратный элемент, операции сложения и умножения по модулю

Понятие алгебраической структуры включает в себя определенное множество объектов и операций над этими объектами. Поскольку практически в любой задаче обработки данных с помощью компьютера выделяется множество самих данных и операции, которые применимы к этим данным, очевидно, что при этом формируются определенные алгебраические структуры. Мы уже знакомы с алгебраическими структурами на множестве натуральных, целых и действительных чисел, на множествах и отношениях. Мы рассмотрим такие алгебраические структуры, как полугруппы, моноиды и группы, которые, в частности, используются для преобразований строк символов и участвуют в формировании более сложных структур — колец и полей. Эти понятия являются базовыми для общей и линейной алгебры, используются при работе с матрицами, при кодировании информации и обработке данных.

Определение

Операцией на множестве S называется функция f, которая является отображением вида $S^n \to S$, $n \in N$, где S^n — декартово произведение $S \times S \times ... \times S$, в которое S входит n раз.

В этом определении есть два важных момента. Во-первых, поскольку операция является функцией, то результат применения

операции определен однозначно. Поэтому данный упорядоченный набор из n элементов множества S функция f переводит только в один элемент из S. Во-вторых, операция замкнута на S в том смысле, что область определения и область значений операции лежат в S^n и S соответственно.

Говорят, что операция $S^n \to S$ имеет порядок n или является n-арной операцией. Чаще встречается ситуация, когда порядок равен 1 или 2. Операции вида $S \to S$ называют ***унарными***, а операции $S^2 \to S$ называют ***бинарными***. Элементы упорядоченного набора из n элементов в области определения S^n называют ***операндами***. Операции обычно обозначают символами, называемыми ***операторами***. В случае унарных операций обычно символ оператора ставят перед или над операндом.

Примеры. Примерами унарных операций являются операции изменения знака (−) на множестве действительных чисел R: $-2{,}678 - 56$, операция возведения в степень (например, в квадрат) на множестве R: 56^2, 7^2. В алгебре множеств примером унарной операции является операция дополнения множества. Бинарными операциями на множестве действительных чисел R являются арифметические операции — сложение, вычитание, умножение, деление ($+$, $-$, $*$, $/$). В алгебре множеств бинарными являются операции — объединение ($\cup$), пересечение ($\cap$), разность ($\setminus$).

Операции записывают одним из трех способов. В первом случае оператор ставится между операндами (***infix***), во втором — перед операндами (***prefix***) и в третьем — после операндов (***postfix***). Рассмотрим три варианта записи бинарной операции арифметического выражения $a + b$.

$$infix\colon a + b,$$

$$prefix\colon +ab,$$

$$postfix\colon ab+.$$

В соответствии с большинством математических текстов мы будем использовать обозначение $infix$. Формы записи $postfix$ и $prefix$ имеют то преимущество, что не требуют скобок при определении порядка вычислений сложных выражений, и это делает их особенно удобными для автоматической обработки, они часто используются для представления выражений в памяти компьютера. Мы рассмотрим их более подробно на примере $postfix$. Алгоритм вычисления значения выражения, записанного в форме $postfix$, выглядит так:

1) При просмотре записи слева направо выполняется первая найденная операция, которой непосредственно предшествует достаточное для нее количество операндов.
2) На место выполненной операции и использованных для этого операндов в строку записывается результат выполнения операции.
3) Возвращаемся к п.1.

Пример. Пусть имеется выражение, которое в стандартной привычной для нас *infix*-форме выглядит так:

$$1 + 2 * 3 + (4 + 5 * (6 + 7)).$$

Результат перевода его в *postfix* будет следующим:

$$123 * + 4567 + * + +.$$

Вычислим теперь значение выражения, используя приведенный алгоритм:

$$1\ \underline{2\ 3\ *}\ +\ 4\ 5\ 6\ 7\ +\ *\ +\ + = \underline{1\ 6\ +}\ 4\ 5\ 6\ 7\ +\ *\ +\ + =$$
$$= 7\ 4\ 5\ \underline{6\ 7\ +}\ *\ +\ + = 7\ 4\ \underline{5\ 13\ *}\ +\ + = 7\ \underline{4\ 65\ +}\ +\ =$$
$$= \underline{7\ 69\ +} = 76.$$

Аналогично записываются выражения с нечисловыми переменными в алгебраической форме. Следующие пары выражений записаны в формах *infix* и *postfix* соответственно:

а) $(a + b) * c + d + e * f + g$, $a\ b + c * d + e\ f * + g +$;

б) $a + (b * (c + d) + e) * f + g$, $a\ b\ c\ d + * e + f * + g +$.

Кроме стандартных известных нам операций (например, $+$, $-$, $*$), существует много других. Мы будем использовать символы $\otimes$ и $\oplus$ для обозначения абстрактных бинарных операций. Иначе говоря, символы $\otimes$ и $\oplus$ используются в качестве переменных для обозначения любых операций. Бинарные операции, определенные на конечных множествах, удобнее задавать при помощи таблиц. Таблица, задающая некоторую бинарную операцию $\otimes$ на некотором множестве A, называется ***таблицей Кэли***, ее строки и столбцы нумеруются элементами множества A, а элементом таблицы, стоящем на пересечении строки a_i и столбца a_j, является элемент $a_k = a_i \otimes a_j$.

Пример. Пусть операция $\otimes$ определена на множестве $\{a, b, c\}$ при помощи таблицы.

$\otimes$	a	b	c
a	a	a	b
b	b	a	c
c	a	b	b

Следовательно, $a \otimes b = a$, $b \otimes b = a$, $c \otimes b = b$, ...

Очевидно, что использование таблиц имеет большое значение, так как некоторые операции, с которыми приходится иметь дело в компьютерной математике, не пригодны для словесного задания.

Перечислим важные свойства, которыми могут обладать операции.

Определение

Пусть дано множество A, на котором определена некоторая бинарная операция $\otimes$. Если $a \otimes b = b \otimes a$ для всех $a, b \in A$, то говорят, что бинарная операция $\otimes$ на множестве A обладает свойством ***коммутативности***.

Если $(a \otimes b) \otimes c = a \otimes (b \otimes c)$ для всех $a, b, c \in A$, то говорят, что бинарная операция $\otimes$ на множестве A обладает свойством ***ассоциативности***.

Пусть на множестве A определены две бинарные операции $\otimes$ и $\oplus$. Если для всех $a, b, c \in A$ $a \otimes (b \oplus c) = (a \otimes b) \oplus \oplus (a \otimes c)$, то говорят, что операция $\otimes$ обладает свойством ***дистрибутивности*** по отношению к операции $\oplus$.

Заметим, что в определении ассоциативности порядок операндов a, b и c сохранен (операция может быть некоммутативной) и использованы круглые скобки, чтобы указать порядок выполнения операций. Таким образом, выражение $(a \otimes b) \otimes c$ требует, чтобы сначала вычислялось $a \otimes b$ и затем результат этого (скажем, x) участвовал в операции $x \otimes c$ в качестве первого операнда. Если операция ассоциативна, то порядок вычислений несуществен и, следовательно, скобки не требуются.

Пример. Обычная бинарная операция сложения $(+)$ на множестве действительных чисел R коммутативна и ассоциативна, а операция вычитания $(-)$ — некоммутативна и неассоциативна, т.е.

$$a + b = b + a, \text{ но } a - b \neq b - a;$$
$$(a + b) + c = a + (b + c), \text{ но } (a - b) - c \neq a - (b - c).$$

Кроме того, на множестве действительных чисел R умножение дистрибутивно по отношению к сложению, а сложение не дистрибутивно по отношению к умножению, т.е.

$$a * (b + c) = a * b + a * c, (a + b) * c = a * c + b * c,$$
$$\text{но } a + (b * c) \neq (a + b) * (a + c).$$

Для решения уравнений относительно каждой операции во множестве-носителе алгебраической структуры выделяется особый элемент, называемый единичным элементом.

Определение

Если для бинарной операции $\otimes$ на множестве A существует элемент $e \in A$ такой, что для всех $a \in A$

$$e \otimes a = a \otimes e = a,$$

тогда e называется ***единицей*** по отношению к операции $\otimes$. Пусть $\otimes$ — операция на A с единицей e и элементы $x, y \in A$ удовлетворяют равенствам

$$x \otimes y = e = y \otimes x.$$

Тогда y называется ***обратным элементом*** к x по отношению к операции $\otimes$, и x называется ***обратным элементом*** к y по отношению к операции $\otimes$.

Иногда различают левые и правые единицы ($e_{лев.} \otimes a = a$ либо $a \otimes e_{прав.} = a$ для любого $a \in A$) и левые и правые обратные элементы, однако, в большинстве случаев единицы являются двусторонними, как в нашем определении.

В случаях, когда бинарная операция считается аналогичной умножению ($*$), единичный элемент обозначается 1, а обратный к элементу x элемент записывается в виде x^{-1}. Когда бинарная операция считается аналогичной сложению ($+$), единичный элемент обозначается 0, а обратный к элементу x элемент записывается в виде $-x$. Обозначим обратный элемент к x как x'.

Приведем примеры единиц и обратных элементов. На множестве действительных чисел R правой единицей по отношению к вычитанию и единицей по отношению к сложению является 0, так как

$$a - 0 = a, \text{ но } 0 - a \neq a, \text{ если } a \neq 0;$$

$$a + 0 = a \text{ и } 0 + a = a \text{ для всех } a.$$

В алгебре множеств для операции объединения $\cup$ единичным элементом является пустое множество $\varnothing$, для операции пересечения единицей является универсальное множество U.

Для дальнейшего необходимо определить операции сложения и умножения по модулю n на множестве целых чисел.

Определение

Пусть n — произвольное натуральное число. ***Сложением по модулю n*** целых чисел a и b называется алгебраическая операция, результатом которой является остаток от деления суммы $a + b$ на n. ***Умножением по модулю n*** чисел a и b называется алгебраическая операция, результатом которой

является остаток от деления произведения $a * b$ на n. Эти операции (обозначим их $\oplus_n$, $\otimes_n$) определены на множестве целых неотрицательных чисел Z_+:

$a \oplus_n b = c$, так, что $a + b = k * n + c$, $0 \le c < n$; $a, b, k \in Z_+$
$a \otimes_n b = d$, так, что $a * b = f * n + d$, $0 \le d < n$; $a, b, f \in Z_+$

Областью значений этих операций является множество целых неотрицательных чисел, меньших n, обозначим его Z_n, $Z_n = \{0, 1, ..., n-1\}$.

Часто используется обозначение

$$a + b \equiv c \ (mod\ n),$$
$$a \times b \equiv d \ (mod\ n)$$

для сложения и умножения по модулю n.

Пример. Приведем примеры сложения и умножения по модулю n.

$2 \oplus_3 2 = \text{Ост}\ (4/3) = 1$, $2 \otimes_3 2 = \text{Ост}\ (4/3) = 1$,
$2 \oplus_4 2 = \text{Ост}\ (4/4) = 0$, $2 \otimes_4 2 = \text{Ост}\ (4/4) = 0$,
$7 \oplus_{10} 8 = \text{Ост}\ (15/10) = 5$, $7 \otimes_{10} 8 = \text{Ост}\ (56/10) = 6$,
$7 \oplus_{12} 8 = \text{Ост}\ (15/12) = 3$, $7 \otimes_{12} 8 = \text{Ост}\ (56/12) = 8$.

Вопросы

1. Дайте определение операции и приведите примеры операций, заданных на различных множествах.
2. Какие из перечисленных действий, заданных на множестве натуральных чисел N, можно назвать операциями:
   а) увеличение на 1;
   б) нахождение числа, кратного данному;
   в) сложение по модулю 5;
   г) нахождение числа, меньшего, чем данное;
   д) нахождение наибольшего общего делителя;
   е) нахождение числа, которое при делении на 7 дает в остатке 6.
3. Что называется порядком операции? Приведите примеры унарной, бинарной, n-арной операции.
4. Что называется оператором? Приведите пример.
5. Что представляют собой формы записи *infix*, *prefix* и *postfix*? Приведите примеры. Почему форма записи *infix* менее удобна для автоматической обработки?
6. Какие операции можно и удобно задавать с помощью таблиц. Почему?
7. Дайте определение свойств ассоциативности, дистрибутивности и коммутативности операции. Приведите примеры.

8. При выполнении какого условия элемент e называется единицей по отношению к некоторой операции $\oplus$?
9. Что такое обратный элемент к некоторому элементу x по отношению к некоторой операции $\oplus$?
10. Является ли существование единицы необходимым и (или) достаточным условием для существования обратных элементов?

Задания

1. Определите, замкнуты ли данные операции относительно данных множеств. Заполните таблицу обозначениями «З» — замкнута или «Н» — не замкнута.

	$x + y$	$x * y$	$x - y$	$\lvert x - y\rvert$	$\max(x, y)$	$\min(x, y)$	$-x$	$\lvert x\rvert$
Z (целые числа)								
N (натуральные числа)								
$\{x \mid 0 \le x \le 10,\ x \in Z\}$								
$\{x \mid -5 \le x \le 5,\ x \in Z\}$								
$\{x \mid -10 \le x \le 0,\ x \in Z\}$								
$\{2x \mid 0 \le x \le 10,\ x \in Z\}$								

2. Определите, обладают ли приведенные операции перечисленными свойствами. Заполните таблицу обозначениями «Да» — обладает и «Нет» — не обладает. Операции рассматриваются на множестве действительных чисел.

	$x + y$	$x * y$	$x - y$	$\lvert x - y\rvert$	$\max(x, y)$	$\min(x, y)$
Ассоциативна						
Коммутативна						
Имеет единицу						

3. Опишите алгоритм, который определяет, ассоциативна бинарная операция или нет. Входным данным является таблица операции (операция задана на конечном множестве).
4. Опишите алгоритм, который определяет, коммутативна бинарная операция $\otimes$ относительно $\oplus$ или нет. Входными данными являются таблицы операций $\otimes$ и $\oplus$ (операции заданы на конечном множестве).

3.2. Понятие алгебраической структуры

Алгебраическая структура, подструктура, гомоморфизм, изоморфизм

Определение

Алгебраической структурой (кратко — ***структурой***) называется множество вместе с заданными операциями, определенными и замкнутыми на этом множестве. Это множество называется ***носителем*** алгебраической структуры.

Пример. Алгебраическая структура с операцией сложения на множестве N натуральных чисел обозначается $(N, +)$.

Пример. Множество $Z_7 = \{0, 1, 2, 3, 4, 5, 6\}$ вместе с обычной операцией сложения $(+)$ не будет являться алгебраической структурой, поскольку результат выполнения операции может не принадлежать множеству Z_7, например, $6 + 3 = 9$, $9 \notin Z_7$. Но $(Z_7, \oplus_7)$ является алгебраической структурой, поскольку область значений операции $\oplus_7$ лежит в Z_7.

Определение

Структура S': $(A', \oplus')$ является ***подструктурой*** алгебраической структуры S: $(A, \oplus)$, если:

1. $A' \subset A$.
2. $\oplus'$ и $\oplus$ операции одного порядка и сужение операции $\oplus$ на подмножестве A' совпадает с операцией $\oplus'$ (например, бинарных операций $a \oplus b = a \oplus' b$ для всех $a, b \in A'$).

Очевидно, что самой крупной подструктурой структуры S является сама структура S. В некоторых случаях других подструктур может не быть.

Примеры. Пусть E — множество четных натуральных чисел, тогда $(E, +)$ будет подструктурой структуры $(N, +)$, где N — множество натуральных чисел.

Структура $(\{0, 1\}, *)$ является подструктурой структуры $(Z, *)$, где Z — множество целых чисел.

Отношения между алгебраическими структурами могут быть не только включающие (структура — подструктура). Возможны и другие отношения, позволяющие осуществлять переход от структуры к структуре, с потерей некоторой информации или без потери информации.

Определение

Пусть заданы две структуры $(A, \otimes)$, $(C, \oplus)$ с операциями $\otimes$, $\oplus$ одного порядка n. Отображение $\varphi: A \to C$ называется ***гомоморфизмом*** из структуры $(A, \otimes)$ в структуру $(C, \oplus)$, если оно перестановочно с операциями в следующем смысле:

$$\varphi \cdot \otimes = \oplus \cdot \vec{\varphi},$$

где отображение $\vec{\varphi}: A^n \to C^n$ действует по правилу

$$\vec{\varphi}(a_1, a_2, \ldots, a_n) = (\varphi(a_1), \varphi(a_2), \ldots, \varphi(a_n)), \ \forall a_i \in A.$$

Для ***бинарных*** операций $(n = 2)$, в частности,

$$\varphi(x \otimes y) = \varphi(x) \oplus \varphi(y) \text{ для любых } x, y \in A.$$

Графическое определение гомоморфизма для случая бинарных операций поясняет рис. 3.1.

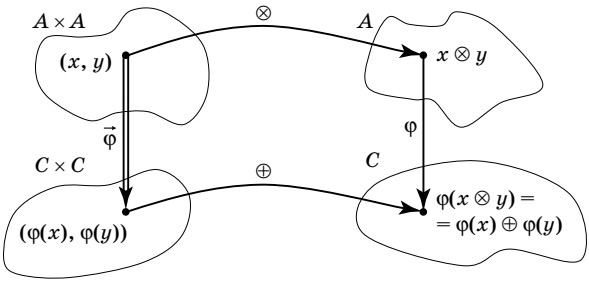


Рис. 3.1. Гомоморфизм φ: φ $(x \otimes y)$ = φ $(x) \oplus \varphi(y)$

Если упростить приведенную иллюстрацию, то получим коммутативную диаграмму, которая связывает отдельные элементы множеств, изображенную на рис. 3.2.

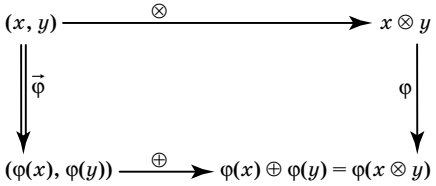


Рис. 3.2. Связь между отдельными элементами множеств при гомоморфизме φ

Часто такие диаграммы изображают в еще более упрощенном виде, указывая только необходимые множества, как показано на рис. 3.3.

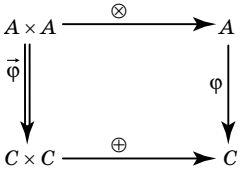


Рис. 3.3. Коммутативная диаграмма, изображающая гомоморфизм φ

Подобные диаграммы часто используются для изображения связей между структурами, они называются ***коммутативными***,

поскольку показывают возможность перехода к результату различными способами (по направлению стрелок).

Пример. Пусть задано отображение θ: $Z_+ \to Z_{10}$, переводящее любое целое неотрицательное число в остаток от деления этого числа на 10. Тогда

$$\theta(20) = 0,\ \theta(17) = 7,\ldots$$

Если $(Z_+, +)$ и $(Z_{10}, \oplus_{10})$ структуры с операцией обычного сложения $+$, определенной на Z_+ и сложением по модулю 10 на Z_{10}, то θ является гомоморфизмом из первой структуры во вторую. Например,

$$\theta(24 + 38) = \theta(62) = 2,$$

$$\theta(24) \oplus_{10} \theta(38) = 4 \oplus_{10} 8 = 2.$$

Получение данного результата двумя разными способами можно проиллюстрировать коммутативной диаграммой (рис. 3.4):

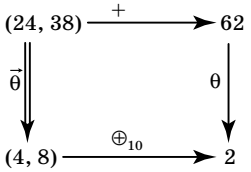


Рис. 3.4. Коммутативная диаграмма. Действие гомоморфизма θ из $(Z_+, +)$ в $(Z_{10}, \oplus_{10})$ для элементов 24 и 38 множества Z

В общем случае для гомоморфизма θ: $(Z_+, +) \to (Z_{10}, \oplus_{10})$ коммутативная диаграмма будет выглядеть так, как это изображено на рис. 3.5.

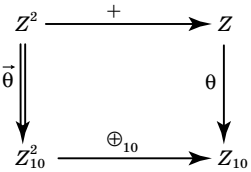


Рис. 3.5. Коммутативная диаграмма действия гомоморфизма θ из $(Z_+, +)$ в $(Z_{10}, \oplus_{10})$

Определение

Гомоморфизм, который является биекцией, называют ***изоморфизмом***. Если существует изоморфизм между двумя структурами, то говорят, что они ***изоморфны*** друг другу.

Таким образом, для любого изоморфизма φ существует обратное отображение φ^{-1}, также взаимно однозначное. Если существует изоморфизм структуры S в структуру Q, то существует и изоморфизм Q в S.

Отношение изоморфизма — это *отношение эквивалентности* на множестве алгебраических структур, поэтому изоморфизм разбивает множество всех алгебраических структур на классы эквивалентности. Используя изоморфизм, можно осуществлять эквивалентные преобразования алгебраических структур. Если алгебраические структуры S и Q *изоморфны*, то элементы и операции Q можно переименовать так, что Q совпадет с S. Любое соотношение в структуре S сохраняется в любой изоморфной ей структуре Q. Это позволяет, получив определенные соотношения в структуре S, автоматически распространить их на все структуры, изоморфные S. Поэтому алгебраические структуры часто рассматриваются с точностью до изоморфизма, то есть рассматриваются классы эквивалентности по отношению изоморфизма.

Пример. Рассмотрим способ измерения длины в дюймах и сантиметрах. Если добавить бинарную операцию сложения, то получим две структуры: (inch, +), (см, +). Определим изоморфизм γ: x (см) = 2,54 * x (inch).

Как указано на диаграмме (рис. 3.6), мы можем провести вычисления (сложение) в дюймах, а затем перевести результат в сантиметры, и также возможно сначала представить те же операнды в сантиметрах и затем провести сложение. В обоих случаях будет получен один и тот же результат. Например, пусть необходимо определить длину d некоторого изделия, состоящего из частей a и b. Измерив части a и b в дюймах, получили, что $a = 10''$, $b = 15''$. Найдем d двумя способами:

$$d = 10'' + 15'' = 25'',\ 2{,}54 * 25'' = 63{,}5 \text{ см};$$

$$d = 10'' * 2{,}54 + 15'' * 2{,}54 = 25{,}4 \text{ см} + 38{,}1 \text{ см} = 63{,}5 \text{ см}.$$

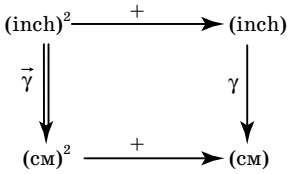


Рис. 3.6. Изоморфизм γ из (inch, +) в (см, +)

Для этого примера коммутативная диаграмма выглядит следующим образом (рис. 3.7):

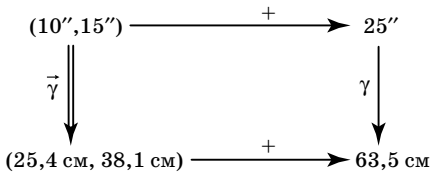


Рис. 3.7. Преобразование отдельных элементов при изоморфизме γ из (inch, +) в (см, +)

Отображение является изоморфизмом, поскольку γ — однозначное соответствие и существует обратное отображение γ′: x (inch) = 0,39 * x (см).

Приведем еще два примера изоморфизмов.

Пример. Пусть Z — множество всех целых чисел, Z_{2N} — множество всех четных чисел. Алгебры $(Z; +)$ и $(Z_{2N}; +)$ изоморфны. Изоморфизмом является отображение φ_{2N}: $n \to 2n$ для всех $n \in Z$.

Пример. Изоморфизмом между алгебраическими структурами $(R_+, *)$ и $(R, +)$, где R_+ — положительное подмножество R и является отображением $a \to \log a$. Условие гомоморфизма $\varphi(x \otimes y) = \varphi(x) \oplus \varphi(y)$ в этом случае имеет вид $\log(a * b) = \log a + \log b$.

Вопросы

1. Дайте определение алгебраической структуры.
2. Что называется подструктурой алгебраической структуры? Каким отношением связаны множества-носители алгебраической структуры и ее подструктуры? Приведите примеры подструктур.
3. Дайте определение гомоморфизма и изоморфизма. Чем они отличаются?
4. С помощью чего осуществляется разбиение алгебраических структур на классы эквивалентности?
5. Приведите примеры изоморфных алгебраических структур.
6. Приведите пример гомоморфизма, не являющегося изоморфизмом. Объясните, почему выбранное вами отношение не изоморфизм.

Задания

1. Рассмотрим алгебраические структуры $(\{a, b, c, d\}, \oplus)$ и $(\{a, b, c\}, \otimes)$, где операции $\oplus$ и $\otimes$ заданы таблицами:

$\oplus$	a	b	c	d
a	a	b	c	d
b	b	c	d	a
c	c	d	a	b
d	d	a	b	c

$\otimes$	a	b	c
a	a	a	c
b	a	b	c
c	c	c	b

Для каждой алгебраической структуры определить:

а) коммутативна ли операция?

б) ассоциативна ли операция?

в) определите, существует ли единица для каждой операции, если существует, то какой это элемент?

г) если единица существует, определите, какие элементы имеют обратные?

2. Определите алгебраическую структуру $(S, \oplus)$, такую, что для любого подмножества $T \subset S$ верно, что $(T, \oplus)$ — подструктура структуры $(S, \oplus)$.

3. Определите, корректно ли задана алгебраическая структура. Если да, определите, коммутативна ли операция и, если возможно, найдите единицу и обратные элементы. Здесь N — множество натуральных чисел, Z — множество целых чисел, R_+ — положительное подмножество множества действительных чисел.

   а) $(N, \otimes)$; $x \otimes y = x - y$;

   б) $(Z, \otimes)$; $x \otimes y = x * y - 1$;

   в) $(N, \otimes)$; $x \otimes y = \max(x, y)$;

   г) $(R_+, \otimes)$; $x \otimes y = x/y$.

4. Докажите, что

   а) две алгебраические структуры не могут быть изоморфны, если множества-носители этих структур имеют разные мощности;

   б) две алгебраические структуры могут не быть изоморфными, если множества-носители этих алгебр имеют одинаковые мощности.

5. Пусть C — множество структур с одной бинарной операцией, $C = \{S_1, S_2, \ldots\}$. Определим отношение $\sim$ на множестве C, такое, что $S_i \sim S_j$ выполняется, если S_i и S_j изоморфны. Докажите, что $\sim$ является отношением эквивалентности на множестве C.

3.3. Простейшие алгебраические структуры

Полугруппа, моноид, группа, абелева группа

Рассмотрим основные типы алгебраических структур, которые имеют только одну бинарную операцию.

Определение

Полугруппой называется алгебраическая структура с множеством-носителем A и бинарной операцией $\otimes$: $A^2 \to A$, которая удовлетворяет только свойству ассоциативности

$$x \otimes (y \otimes z) = (x \otimes y) \otimes z; \qquad x, y, z \in A.$$

Определение

Моноидом называют алгебраическую структуру с множеством-носителем M и бинарной операцией $\otimes$: $M^2 \to M$ такой, что

1. $\otimes$ ассоциативна:

$$x \otimes (y \otimes z) = (x \otimes y) \otimes z, \text{ для всех } x, y, z \in M.$$

2. Существует $e \in M$ — единица по отношению к $\otimes$:

$$e \otimes x = x = x \otimes e \text{ для всех } x \in M.$$

Таким образом, моноид — это полугруппа с единицей.

Полугруппы и моноиды используются при обработке строк символов. Рассмотрим пример.

Пример. При обработке строк символов вводится операция конкатенации — слияния строк. Обозначим ее $\bullet$. Конкатенация определяется как $\alpha \bullet \beta = \alpha\beta$, где α и β — строки символов. Возьмем строки: «пар», «сам», «о», «воз», «ход», «лет». Применив операции конкатенации, получаем следующие строки:

«пар» • «о» = «паро»
«сам» • «о» = «само»
«паро» • «воз» = «паровоз»
«паро» • «ход» = «пароход»
«само» • «лет» = «самолет».

Очевидно, что эта операция ассоциативна, поскольку
(«пар» • «о») • «воз» = «пар» • («о» • «воз») = «паровоз» и т.д.

Следовательно $(A^+, \bullet)$ является полугруппой, где A^+ — множество всевозможных строк, состоящих из букв русского алфавита. Если теперь мы обозначим через A^* множество всевозможных строк, состоящих из букв русского алфавита и пустой строки ε, то получим структуру $(A^*, \bullet)$, которая является моноидом с единичным элементом ε. Поскольку ε обозначает пустую строку, то можем записать ε = «».

«самолет» • «» = «» • «самолет» = «самолет» и т.д.

Определение

Группой называют множество G с бинарной операцией $\otimes$, замкнутой в G такой, что

1. $\otimes$ ассоциативна:

$$x \otimes (y \otimes z) = (x \otimes y) \otimes z \text{ для всех } x, y, z \in G.$$

2. Существует элемент $e \in G$ — единица по отношению к $\otimes$:

$$e \otimes x = x \otimes e = x \text{ для всех } x \in G.$$

3. Каждому элементу $x \in G$ соответствует обратный элемент $x' \in G$ по отношению к $\otimes$:

$$x' \otimes x = x \otimes x' = e \text{ для всех } x \in G.$$

Из определения следует, что группа — это моноид, в котором все элементы обратимы.

Часто к словам «группа» и «моноид» приписывают термин «коммутативный». Это означает, что операция в рассматриваемой структуре удовлетворяет свойству коммутативности, т.е.

$$y \otimes x = x \otimes y \text{ для всех } x, y \in M \text{ или } G.$$

Коммутативная группа называется ***абелевой группой*** (в честь норвежского математика Абеля).

Примеры. 1. Группой является множество действительных чисел вместе с операцией сложения: $(R, +)$, подгруппой этой группы является $(Z, +)$, где Z — множество целых чисел. Структура $(K, +)$, где K — множество целых чисел, кратных k, $k \in N$, и является подгруппой группы $(Z, +)$. Для этих групп единицей является 0, обратный элемент получается с помощью применения унарной операции изменения знака «$-$». Перечисленные группы являются абелевыми группами, поскольку сложение коммутативно.

2. Структура $(N, +)$, где N — множество натуральных чисел, не является группой, поскольку не существует обратных элементов и единицы. На самом деле $(N, +)$ — полугруппа.

3. Структуры $(R, *)$, $(Z, *)$ и $(N, *)$ не являются группами, а являются моноидами. Единичным элементом для операции умножения является 1. Обратные элементы существуют на множестве действительных чисел R для всех элементов, кроме 0: (не существует 0^{-1}, такого, что $0 * 0^{-1} = 1$). Таким образом, операция умножения задает группу на множестве действительных чисел, кроме нуля $(R\backslash\{0\}, *)$. Положительное подмножество множества действительных чисел с операцией умножения $(R_+, *)$ тоже является группой — подгруппой

группы $(R\backslash\{0\}, *)$. Умножение коммутативно, следовательно, эти группы являются абелевыми.

4. Обозначим $M_n(R)$ множество всех квадратных матриц порядка n с элементами из множества действительных чисел. Структура $(M_n(R), +)$ — коммутативный моноид с единицей — нулевой матрицей. Структура $(M_n(R), *)$ — некоммутативный моноид с единицей — единичной матрицей.

5. Структура $(Z_n, \oplus_n)$ — группа с единицей — 0 и обратным элементом — $x' = n - x$; $(Z_n, \otimes_n)$ — моноид с единицей — 1.

Для решения уравнений необходимо существование и единственность единиц и обратных элементов. Докажем эти утверждения.

Утверждение 1. Пусть $\otimes$ — операция на множестве A и существует единица e по отношению к $\otimes$, тогда ***единичный элемент единственный***.

Доказательство. Пусть e и e' — две единицы по отношению к $\otimes$. Тогда для любых $a, b \in A$ верно $a = e' \otimes a$, $b = b \otimes e$. Подставляя $a = e$, $b = e'$, получаем $e = e' \otimes e = e'$.

Утверждение 2. Пусть $\otimes$ — ассоциативная операция на множестве A и e — единица по отношению к $\otimes$. Тогда, если $x \in A$ и x имеет обратный элемент, то ***обратный элемент единственный*** по отношению к $\otimes$.

Доказательство. Допустим, что x' и x'' — обратные элементы к x, так что

$$x \otimes x' = x' \otimes x = e \text{ и } x \otimes x'' = x'' \otimes x = e,$$

тогда

$$x' = x' \otimes e = x' \otimes (x \otimes x'') = (x' \otimes x) \otimes x'' = e \otimes x'' = x''.$$

Внутри группы $(G, \otimes)$ можно решить уравнение $a \otimes x = b$.

К равенству $a \otimes x = b$ применяем слева a' — обратный к a элемент и последовательно получаем:

$a' \otimes (a \otimes x) = a' \otimes b$,

$(a' \otimes a) \otimes x = a' \otimes b$ ($\otimes$ ассоциативна),

$e \otimes x = a' \otimes b$ (свойство обратных элементов),

$x = a' \otimes b$ (свойство единицы); x — решение.

Существование единицы и обратных элементов относительно некоторой операции накладывает существенное ограничение на вид таблицы Кэли для этой операции. Рассмотрим группу $(Z_7, \oplus_7)$. Таблица Кэли (см. п. 3.1) для операции $\oplus_7$ выглядит следующим образом.

$\oplus_7$	0	1	2	3	4	5	6
0	0	1	2	3	4	5	6
1	1	2	3	4	5	6	0
2	2	3	4	5	6	0	1
3	3	4	5	6	0	1	2
4	4	5	6	0	1	2	3
5	5	6	0	1	2	3	4
6	6	0	1	2	3	4	5

Единицей относительно операции $\oplus_7$ является 0. Таблица симметрична относительно диагонали — это условие определяет коммутативность операции. Заметим, что каждый столбец таблицы содержит все элементы группы.

Утверждение 3. Любой столбец таблицы Кэли для операции конечной группы содержит все элементы группы.

Доказательство. Рассмотрим некоторую группу $(G, \otimes)$, G — конечное множество. Предположим, что некоторый столбец a_i таблицы Кэли для операции $\otimes$ не содержит какого-нибудь элемента множества G. Тогда какой-то другой элемент a_j в этом столбце должен встретиться дважды, скажем, в k-ой и i-ой строках. Но тогда $a_k \otimes a_i = a_j$, $a_l \otimes a_i = a_j$ и следовательно,

$$a_k \otimes a_i = a_l \otimes a_i,$$

$$a_k \otimes a_i \otimes a_i' = a_l \otimes a_i \otimes a_i'$$

(умножаем обе части равенства на a_i'),

$$a_k \otimes e = a_l \otimes e,$$

($a_i \otimes a_i' = e$ по определению обратного элемента),

$$a_k = a_l \quad (a_i \otimes e = a_i \text{ по определению единицы}).$$

Получаем $a_k = a_i$, что невозможно, так как здесь a_k, a_i — элементы группы, задающие разные строки таблицы. Таким образом, i-й столбец таблицы Кэли является перестановкой на множестве элементов группы.

Вопросы

1. Дайте определение полугруппы, моноида и группы.
2. Приведите примеры полугруппы, не являющейся моноидом, и моноида, не являющегося группой.
3. Приведите примеры группы.
4. Какая группа называется абелевой? Приведите пример.
5. Какие свойства элементов группы делают возможным решение уравнений внутри группы?

6. Какими свойствами обладает таблица Кэли для конечной группы?
7. Какими свойствами обладает операция абелевой группы?

Задания

1. Докажите, что в группе $(G, \otimes)$ для всех $a, b, c \in G$ выполняется:
   а) если $a \otimes b = a \otimes c$, то $b = c$, и, если $b \otimes a = c \otimes a$, то $b = c$;
   б) $(a')' = a$ — обратный элемент к обратному элементу к a равен a.
2. Постройте таблицу Кэли для группы $(Z_5, \oplus_5)$. Для каждого элемента определите обратный.
3. Постройте группу с бинарной операцией max, результатом выполнения которой является наибольший из двух операндов.
4. Пусть множество $E = \{0, -2, 2, -4, 4, ...\}$. Покажите, что $(E, +)$ — подгруппа группы $(Z, +)$.
5. Пусть задана алгебраическая структура $(T, \otimes)$, где операция $\otimes$ ассоциативна. Пусть существует элемент $0 \in T$, такой, что для любого $x \in T$ верно $0 \otimes x = x \otimes 0 = 0$. Покажите, что $(T, \otimes)$ не может быть группой, за исключением случая $T = \{0\}$.
6. Пусть некоторый компьютер использует регистры по k бит для представления неотрицательных целых чисел. Единственная выполняемая операция — сложение. Какую алгебраическую структуру представляет собой данный компьютер, если:
   а) при возникновении переполнения значение старшего бита теряется;
   б) при возникновении переполнения результатом является наибольшее из представимых чисел.
   Какова мощность множества-носителя этой алгебраической структуры?
7. Составьте алгоритм проверки. Является ли структура $(A, \otimes)$ группой, полугруппой или моноидом. Входными данными являются элементы таблицы Кэли операции $\otimes$.
8. Составьте алгоритм нахождения обратного элемента к a и решения уравнения $a \otimes x = b$ относительно x в группе $(G, \otimes)$. Входными данными являются элементы таблицы Кэли операции $\otimes$.

3.4. Кольца и поля

В этом разделе рассматриваются алгебраические структуры с двумя бинарными операциями $\otimes$ и $\oplus$. Операцию $\otimes$ называют умножением, а операцию $\oplus$ — сложением. Для $\otimes$ единичный элемент обозначается 1, а обратный к элементу x относительно $\otimes$ записывается в виде x^{-1}. Для $\oplus$ единичный элемент обозначается 0,

а обратный к элементу x относительно $\oplus$ записывается в виде $-x$. Понятно, что для различных структур эти операции определяются по-разному, хотя часто называются одинаково. Например, умножение матриц, умножение целых чисел, логическое умножение и т.д. Рассмотрим вначале структуры, называемые кольцами. Многие математические конструкции, которые естественно возникают в линейной алгебре (особенно в теории матриц), являются кольцами или содержат кольца как подструктуры.

Определение

Кольцом $(R, \oplus, \otimes)$ называется множество R с определенными на нем бинарными операциями $\oplus$ и $\otimes$, такими, что

1. $\oplus$ ассоциативна:
$$x \oplus (y \oplus z) = (x \oplus y) \oplus z \text{ для всех } x, y, z \in R.$$
2. $\oplus$ коммутативна:
$$x \oplus y = y \oplus x \text{ для всех } x, y \in R.$$
3. $\oplus$ имеет единицу, которая называется нулем и обозначается 0:
$$0 \oplus x = x \text{ для всех } x \in R.$$
4. Существует обратный элемент относительно $\oplus$ для каждого $x \in R$:
$$(-x) \oplus x = x \oplus (-x) = 0 \text{ для всех } x \in R.$$
5. $\otimes$ ассоциативна:
$$x \otimes (y \otimes z) = (x \otimes y) \otimes z \text{ для всех } x, y, z \in R.$$
6. $\otimes$ дистрибутивна по отношению к $\oplus$ слева и справа:
$$x \otimes (y \oplus z) = (x \otimes y) \oplus (x \otimes z),$$
$$(x \oplus y) \otimes z = (x \otimes z) \oplus (y \otimes z) \text{ для всех } x, y, z \in R.$$

Другими словами, если $(R, \oplus)$ — абелева группа, $(R, \otimes)$ — полугруппа и $\otimes$ дистрибутивна по $\oplus$, то $(R, \oplus, \otimes)$ — кольцо.

Будем говорить, что кольцо ***коммутативно***, если умножение $\otimes$ коммутативно, и является ***кольцом с единицей***, если существует единица относительно умножения. Кольцо с единицей называется ***алгеброй***. Как обычно, ее обозначают символом 1. Легко показать, что в кольце $(R, \oplus, \otimes)$ для любых $a, b \in R$ выполняются соотношения

$$0 \otimes a = a \otimes 0 = 0,$$
$$a \otimes (-b) = (-a) \otimes b = -(a \otimes b),$$
$$(-a) \otimes (-b) = a \otimes b.$$

В кольце $(R, \oplus, \otimes)$ фактически присутствует ***некоммутативная*** бинарная операция вычитания «$\ominus$», определяемая по правилу $a \ominus b = a \oplus (-b)$. Она является правой обратной относительно сложения в том смысле, что $(a \oplus b) \ominus b = a$. Дествительно,

$$(a \oplus b) \ominus b = (a \oplus b) \oplus (-b) = a \oplus b \oplus (-b) = a \oplus 0 = a.$$

Определение

Поле $(R, \oplus, \otimes)$ — это коммутативное кольцо с единицей 1 (отличной от 0), в котором каждый элемент a (отличный от 0) обратим по умножению.

Поле можно определить как объединение двух абелевых групп, связанных одним законом дистрибутивности.

Пример. Рассмотрим алгебраическую структуру $(Z_n, \oplus_n, \otimes_n)$ (см. разделы 3.2, 3.3). Для случая $n = 6$ таблицы Кэли операций $\oplus_6$ и $\otimes_6$ выглядят следующим образом.

$\oplus_6$	0	1	2	3	4	5
0	0	1	2	3	4	5
1	1	2	3	4	5	0
2	2	3	4	5	0	1
3	3	4	5	0	1	2
4	4	5	0	1	2	3
5	5	0	1	2	3	4

$\otimes_6$	0	1	2	3	4	5
0	0	0	0	0	0	0
1	0	1	2	3	4	5
2	0	2	4	0	2	4
3	0	3	0	3	0	3
4	0	4	2	0	4	2
5	0	5	4	3	2	1

В Z_6 существует 4 случая, когда произведение двух ненулевых элементов может давать нуль, а именно: (2, 3), (3, 4), (3, 2), (4, 3), следовательно $(Z_6, \oplus_6, \otimes_6)$ не является полем. Таким образом, $(Z_n, \oplus_n, \otimes_n)$ является коммутативным кольцом с единицей при любом $n \in N$. Если p — простое число, то алгебра $(Z_p, \oplus_p, \otimes_p)$ является полем.

Пример. $(R, *, +)$ — множество действительных чисел со стандартными операциями умножения и сложения является полем и состоит из абелевых групп $(R, +)$, $(R\backslash\{0\}, *)$, операции в которых связаны соответствующим законом дистрибутивности.

Структуру $(R, *, +)$ называют ***полем действительных чисел***.

Вопросы

1. Какими свойствами обладают операции кольца?
2. Дайте определение кольца, используя понятия группы и полугруппы.
3. Выполнение каких соотношений можно доказать, используя свойства операций кольца?
4. Какими свойствами обладают операции поля?

5. Чем поле отличается от кольца?
6. Дайте определение поля, используя понятие группы.

Задания

1. Докажите, что в поле $(F, \oplus, \otimes)$ выполняются соотношения:
   а) $-a = a \otimes (-1)$;
   б) $-(a + b) = (-a) + (-b)$;
   в) $-(-a) = a$;
   г) если $a \neq 0$, то $(a^{-1})^{-1} = a$;
   д) $(-a) \otimes (-b) = a \otimes b$;
   е) $a \otimes b = 0 \quad \Rightarrow \quad a = 0$ или $b = 0$.
2. Покажите, что алгебраическая структура $(Z_p, \oplus_p, \otimes_p)$ является полем, если p — простое число. Приведите пример.
3. Постройте подструктуры поля действительных чисел, являющиеся полями, кольцами.
4. Докажите двустороннюю дистрибутивность умножения $\otimes$ относительно вычитания $\ominus$ в кольце $(R, \oplus, \otimes)$:
   $a \times (b \ominus c) = (a \otimes b) \ominus (a \otimes c)$;
   $(b \ominus c) \otimes a = (b \otimes a) \ominus (c \otimes a)$.

3.5. Решетки

Верхняя и нижняя грань в частично упорядоченном множестве, решетка, полная решетка, единица и нуль решетки

Рассмотрим алгебраическую структуру $(M, \leq)$ с множеством-носителем M и бинарным отношением R ***частичного порядка***, которые будем обозначать $\leq$. Напомним, что два элемента $a, b \in M$ могут быть ***сравнимыми*** по отношению $\leq$, или ***несравнимыми*** (п. 2.4).

Определение

Верхней гранью для пары элементов $a, b \in M$ называется элемент $c_i \in M$, такой, что $a \leq c_i$, $b \leq c_i$. ***Нижней гранью*** для пары элементов $a, b \in M$ называется элемент $d_i \in M$, такой, что $d_i \leq a$, $d_i \leq b$.

Наименьшей верхней гранью для пары элементов $a, b \in M$ называется элемент $c \in M$, наименьший из всех верхних граней для a, b. Это означает, что для любого $c_i \in M$, такого, что $c_i \neq c$, $a \leq c_i$, $b \leq c_i$, выполняется $c \leq c_i$. ***Наибольшей нижней гранью*** для элементов $a, b \in M$ называется элемент $d \in M$ наибольший из всех верхних граней для a, b. Это означает, что для любого $d_i \in M$, отличного от d и такого, что $d_i \neq d$, $d_i \leq a$, $d_i \leq b$, выполняется $d_i \leq d$.

Определение

Частично упорядоченное множество — алгебраическая структура $(A, \leq)$, в котором каждая пара элементов имеет наименьшую верхнюю и наибольшую нижнюю грани, называется ***решеткой***.

Определив в решетке $(A, \leq)$ бинарные операции $\wedge$ и $\vee$, где $\wedge$ — нахождение наибольшей нижней грани и $\vee$ — нахождение наименьшей верхней грани, получим алгебраическую структуру решетки с двумя бинарными операциями: $(A, \wedge, \vee)$.

Структуру конечных частично упорядоченных множеств можно изобразить при помощи диаграмм Хассе, в которых элементы изображаются точками, расположенными на разных горизонталях, причем бóльшие элементы соединяются с непосредственно меньшими с помощью опускающихся линий. В результате элемент a оказывается больше элемента b тогда и только тогда, когда на диаграмме можно перейти от a к b по опускающимся вниз линиям. Такая диаграмма является подграфом графа бинарного отношения ≥ 0, обратного к $\leq$, причем в этом подграфе не указаны направления дуг, поскольку известно, что для каждой линии подразумевается направление сверху-вниз (сравнить рис. 2.15 и 2.16).

Пример. Рассмотрим множество B_n двоичных векторов длины n, частично упорядоченное следующим образом: $v \leq w$, если в векторе w единицы стоят на всех тех местах, на которых они стоят в v (и, возможно, еще на некоторых). Например, (010) ≤ (011), а (010) и (100) несравнимы. Изобразим граф отношения порядка $\geq$ и диаграмму для случая $n = 3$ (рис. 3.8).

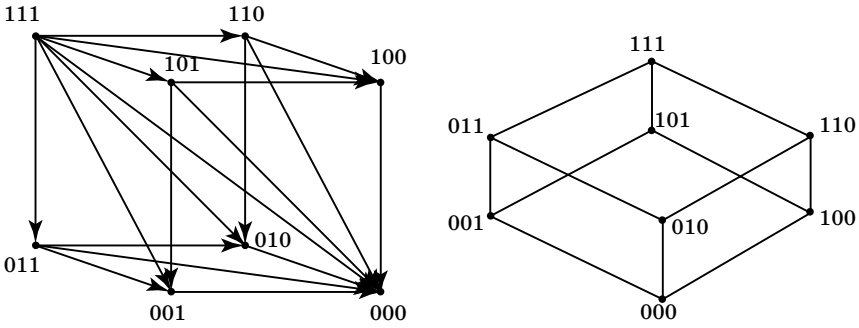


Рис. 3.8. Граф отношение порядка $\geq$
и диаграмма для множества двоичных векторов B_3

Множество B_n и введенный на нем частичный порядок $\leq$ образует решетку; в ней наименьшая верхняя грань $v \vee w$ — это вектор, в котором единицы стоят на тех и только тех местах, где единицы есть либо в v, либо в w, а наибольшая нижняя грань $v \wedge w$ — это вектор, в котором единицы стоят на тех и только тех местах, где есть единицы одновременно и в v и в w. Например, $(010) \vee (100) = (110)$, $(010) \wedge (100) = (000)$.

Таким образом, $v \vee w$ на диаграмме определяется как ближайшая вершина, которая находится над v и w и из которой есть путь вниз и в v и в w. Однако, если есть путь из v вниз к w, то $v \vee w$ будет равно v. Аналогично, $v \wedge w$ на диаграмме определяется как ближайшая вершина, которая находится под v и w и в которую есть путь вниз и из v и из w. Однако, если есть путь из v вниз в w, то $v \wedge w$ будет равно w.

Не всякая алгебраическая структура, задаваемая диаграммой Хассе, является решеткой.

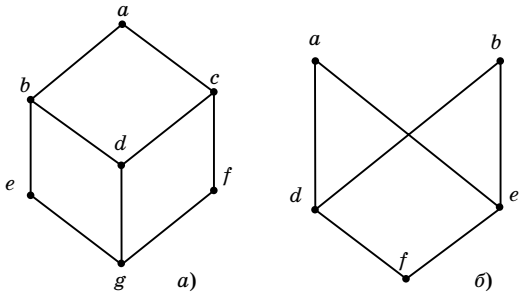


Рис. 3.9. Диаграммы, определяющие отношения частичного порядка

На рис. 3.9,*а* изображена структура, являющаяся решеткой, так как в ней каждая пара элементов имеет наименьшую верхнюю и наибольшую нижнюю грани. Для этой решетки справедливы выражения: $b \wedge c = d$, $d \wedge a = d$, $e \vee g = e$, $e \vee f = a$.

Структура, диаграмма которой представлена на рис. 3.9,*б*, не является решеткой, поскольку пара элементов (a, b) не имеет верхней грани. Кроме того, пара (a, b) имеет три нижние грани — f, d и e, но из них невозможно выбрать единственную наибольшую, поскольку d и e несравнимы. Таким образом, пара элементов (a, b) не имеет наибольшей нижней грани.

Несложно показать, что $\wedge$ и $\vee$ ассоциативны и коммутативны, поэтому можно рассматривать применение этих операций для любого конечного подмножества элементов решетки.

Определение

Решетка, в которой наибольшая нижняя грань и наименьшая верхняя грань существуют для любого подмножества ее элементов, называется ***полной***. Ввиду ассоциативности операций $\wedge$ и $\vee$ конечная решетка всегда полна. Наименьшая верхняя грань 1 для всех элементов полной решетки — это максимальный элемент решетки, называемый ***единицей решетки***. Наибольшая нижняя грань 0 для всех элементов полной решетки — это минимальный элемент решетки или ***нуль решетки***.

Нуль и единица решетки единственны, поскольку наименьшая верхняя грани и наибольшая нижняя грань единственны.

В решетках $(A, \wedge, \vee)$ выполняются следующие свойства операций $\vee$, $\wedge$ для любых $a, b \in A$:

1. Операции $\wedge$, $\vee$ коммутативны:

$$a \wedge b = b \wedge a,\ a \vee b = b \vee a.$$

2. Операции $\wedge$, $\vee$ ассоциативны:

$$(a \wedge b) \wedge c = a \wedge (b \wedge c),\ (a \vee b) \vee c = a \vee (b \vee c).$$

3. Операции $\wedge$, $\vee$ идемпотентны:

$$a \wedge a = a,\ a \vee a = a.$$

4. Выполняются законы поглощения:

$$a \wedge (a \vee b) = a,\ a \vee (a \wedge b) = a.$$

Выполнение этих свойств достаточно, чтобы структура $(A, \wedge, \vee)$ являлась решеткой. Если в решетке присутствуют нуль решетки и единица решетки, то для них выполняются следующие свойства:

Для любого $a \in A$ верно $a \vee 0 = a$, $a \wedge 1 = a$, $a \wedge 0 = 0$, $a \vee 1 = 1$.

Решетка с нулем 0 и единицей 1 называется ***решеткой с дополнением***, если для каждого ее элемента a есть обратный элемент $\neg a$ по отношению к обеим операциям $\wedge$ и $\vee$: для любого $a \in A$ справедливо $a \vee (\neg a) = 1$, $a \wedge (\neg a) = 0$.

Операция нахождения обратного элемента удовлетворяет тождеству: $\neg(\neg a) = a$ для любого $a \in A$.

Решетка называется ***дистрибутивной***, если операции $\wedge$ и $\vee$ дистрибутивны друг относительно друга:

$$a \wedge (b \vee c) = (a \wedge b) \vee (a \wedge c),\ a \vee (b \wedge c) = (a \vee b) \wedge (a \vee c).$$

Определение

Дистрибутивная решетка с дополнением называется ***булевой решеткой*** или ***булевой алгеброй*** $(A, \wedge, \vee, \neg, 0, 1)$, где $\neg$ — унарная операция нахождения обратного элемента, 0, 1 — нуль и единица решетки.

Можно доказать, что в булевой алгебре выполняются следующие тождества, называемые законами де Моргана:

$$\neg(a \vee b) = \neg a \wedge \neg b,\ \neg(a \wedge b) = \neg a \vee \neg b \text{ для любых } a, b \in A.$$

Приведем еще несколько примеров решеток.

Пример. Любое полностью упорядоченное множество (см. п. 2.4), в котором для каждой пары элементов (a, b) существует максимальный из двух элемент $\max(a, b)$ и минимальный из двух элемент $\min(a, b)$, является решеткой, в которой для любых $a, b \in M$ верно $a \vee b = \max(a, b)$, $a \wedge b = \min(a, b)$. Такая решетка не является булевой, поскольку в ней нет обратных элементов.

Пример. Определим на множестве натуральных чисел N отношение частичного порядка следующим образом: $a \le b$, если a делит b. Тогда $a \vee b$ — наименьшее общее кратное a и b, $a \wedge b$ — наибольший общий делитель a, b. Например, $9 \vee 15 = 45$, $9 \wedge 15 = 3$, $5 \wedge 9 = 1$, $5 \vee 9 = 45$. Определенная таким образом структура $(N, \vee, \wedge)$ является решеткой, нуль которой есть число 1. Данная решетка не является булевой, она не имеет единицы решетки, то есть такого элемента, который был бы наименьшим общим кратным для всех пар чисел из N.

Пример. Система всех подмножеств $\{M_i\}$ любого конечного множества X частично упорядочена по включению, т.е. $M_i \le M_j$, если и только если $M_i \subseteq M_j$. Эта система является решеткой, элементами которой являются множества M_i, а операциями $\vee$, $\wedge$ — обычные теоретико-множественные операции объединения и пересечения: $\cup$, $\cap$. Эта решетка всегда полна. Единицей этой решетки является само множество X, нулем — пустое множество. Такая решетка является булевой алгеброй.

Заметим, что алгебраический формализм решеток применяется в теории графов и в программировании, в частности, в языках программирования и анализе типов данных.

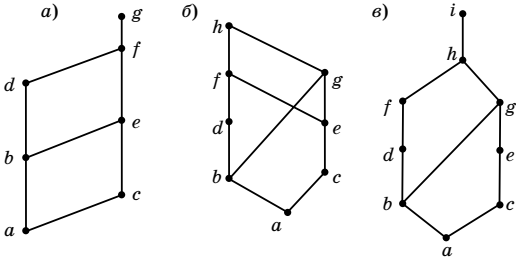
Вопросы

1. Дайте определение решетки. Определите наименьшую верхнюю и наибольшую нижнюю грань двух элементов частично упорядоченного множества.

2. Определите бинарные операции $\wedge$ и $\vee$.
3. Каким образом строение конечных частично упорядоченных множеств можно изображать с помощью диаграмм?
4. Приведите примеры диаграмм, изображающих структуры, являющиеся решетками, и структуры, не являющиеся решетками.
5. В каких случаях конечное упорядоченное множество не является решеткой?
6. Какие элементы называются нулем и единицей решетки?
7. Приведите примеры решеток.

Задания

1. Определите, являются ли решетками частично упорядоченные множества, изображенные следующими диаграммами:



2. Определите, являются ли решетками алгебраические структуры в виде следующих частично упорядоченных множеств:
   а) (A = {1, 3, 6, 9, 12}, отношение порядка — «делитель»). Отношение «делитель»обозначим «|» (например: 3 |6, 6 ∤ 9);
   б) (A = {1, 5, 25, 125}, отношение порядка −|);
   в) (Z — множество целых чисел, отношение порядка — ≤);
   г) (2^A, отношение порядка $\subseteq$), где $A = \{a, b, c\}$;
   д) (2^Z, $\subseteq$).
3. Докажите, что любое непустое конечное подмножество решетки имеет наименьшую верхнюю и наибольшую нижнюю грань (гранями могут быть элементы решетки, не принадлежащие этому подмножеству).
4. Какие из следующих пар элементов являются сравнимыми в частично упорядоченном множестве целых положительных чисел Z^+ по отношению «делитель» — (Z^+, |):
   а) 5, 15;
   б) 6, 9;
   в) 8, 16;
   г) 7, 7.

раздел 4

Булевы функции и преобразования

4.1. Булевы переменные и функции

Двоичные интерпретации, существенные и фиктивные переменные

Для представления информации в компьютерах используется двоичная система счисления. Таким образом, все операции, которые выполняет компьютер, производятся на множестве $\{0, 1\}$. Эти преобразования удобно формально представлять с помощью аппарата двоичной логики, который был разработан Джорджем Булем в середине XIX века. Эта алгебраическая структура является алгеброй и получила название булевой (п. 3.5). Булева алгебра используется при решении различных задач обработки информации, при работе с базами данных, в логическом программировании, при проектировании интеллектуальных систем, для конструирования и анализа работы компьютеров и других электронных устройств. В данной главе рассматриваются основные свойства булевых функций с аргументами из множества $\{0, 1\}$ и способы представления булевых функций в виде выражений булевой алгебры. Булева функция может иметь большое количество переменных и знаков операций, в то время, как может существовать другое, эквивалентное представление данной функции, имеющее меньшее количество переменных и операций. В разделах главы описываются методы получения выражений с минимальным количеством переменных и знаков операций.

Рассмотрим двухэлементное множество B, элементы которого будем обозначать через **0** и **1**: $B = \{0, 1\}$.

Определение
Переменные, которые могут принимать значения только из множества B, называются ***логическими*** или ***булевыми*** переменными. Сами значения 0 и 1 булевых переменных называются ***булевыми константами***.

В языках программирования для работы с такими переменными, как правило, вводится специальный логический (булевский) тип (например, в языках Pascal и Java — boolean, в C++ — bool). Переменная этого типа принимает два значения: true и false.

Определение
Функция вида $y = f(x_1, x_2, \ldots, x_n)$, аргументы x_i и значения y которой принадлежат множеству B, называется n-местной ***булевой функцией***. Такие функции также называют ***логическими*** или ***переключательными*** функциями.

Определение
Кортеж $(x_1, x_2, \ldots, x_n)$ конкретных значений булевых переменных называется ***двоичным словом*** (***n-словом***) или ***булевым набором*** длины n. Для булевой функции $y = f(x_1, x_2, \ldots, x_n)$ конкретное (индивидуальное) значение булевого набора $(x_1, x_2, \ldots, x_n)$ называется также ***интерпретацией булевой функции f***. Множество всех двоичных слов, обозначаемое через B^n, называется ***n-мерным булевым кубом*** и содержит 2^n элементов-слов: $|B^n| = 2^n$.

Действительно, для $n = 1$ есть всего $2^1 = 2$ слова — (0) и (1), а за счет увеличения длины слова на один символ количество слов увеличивается в два раза, так как дополнительный символ может принимать два значения — 0 или 1. Для наглядности изобразим в одной строке наборы булевых переменных возрастающей длины, во второй строке — количества различных наборов соответствующей длины:

$$(x_1),\ \ (x_1, x_2),\ \ (x_1, x_2, x_3),\ \ \ldots,\ \ (x_1, x_2, \ldots, x_n);$$
$$2,\qquad 2 \cdot 2,\qquad 2 \cdot 2 \cdot 2 = 2^3,\ \ \ldots,\ \ 2 \cdot 2 \cdot \ldots \cdot 2 = 2^n.$$

Покажем, что количество всех возможных булевых функций $y = f(x_1, x_2, \ldots, x_n)$ равно 2^{2^n}. Заметим, что в обозначениях п. 2.5 булева функция есть отображение f: $B^n \to B$, $B = \{0, 1\}$. Обозначим n-слово одним символом $u = (x_1, x_2, \ldots, x_n)$ и пронумеруем все различные слова: $\{u\} = \{u_1, u_2, \ldots, u_p\}$, где $p = 2^n$. Две булевы функции $f_1(u)$ и $f_2(u)$ совпадают, если $f_1(u_i) = f_2(u_i)$ для всех u_i, $i = 1, 2, \ldots, p = 2^n$.

Поскольку $f(u)$ может иметь два значения (0 или 1), то число различных функций $f(u)$ равно количеству различных булевых слов (наборов) длины p, то есть числу $2^p = 2^{2^n}$.

Функции нескольких независимых переменных можно рассматривать как функции от большего количества переменных. При этом значение функции не меняется при изменении значения этих «добавочных» переменных.

Определение

Переменная x_i в функции $f(x_1, \ldots, x_{i-1}, x_i, x_{i+1}, \ldots, x_n)$ называется ***несущественной*** (или ***фиктивной***), если

$$f(x_1, \ldots, x_{i-1}, 0, x_{i+1}, \ldots, x_n) = f(x_1, \ldots, x_{i-1}, 1, x_{i+1}, \ldots, x_n)$$

при любых значениях остальных переменных, т.е. если изменение значения x_i в любом наборе значений $x_1, \ldots, x_n$ не меняет значения функции.

В этом случае функция $f(x_1, \ldots, x_n)$ фактически зависит от $n-1$ переменной, т.е. представляет собой функцию $g(x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n)$. Говорят, что функция g получена из функции f удалением фиктивной переменной, а функция f получена из g введением фиктивной переменной, причем эти функции по определению равны.

Вопросы

1. Какие переменные называются булевыми или логическими переменными?
2. Дайте определение булевой функции.
3. Что представляет собой область определения и область значений булевой функции?
4. Поясните смысл понятия «интерпретация булевой функции».
5. Какие переменные называются несущественными или фиктивными?

Задания

1. Сколько интерпретаций имеет булева функция от трех переменных $f(x, y, z)$? Перечислите их.
2. Найдите количество булевых функций n переменных, принимающих значение 1 ровно на одном наборе.
3. Найдите количество булевых функций n переменных, принимающих на противоположных наборах одинаковые значения. Противоположными наборами называются такие наборы, которые в одинаковых позициях содержат противоположные элементы. Например: (0, 0) – (1, 1); (1, 0) – (0, 1) и т.д.

4.2. Способы задания булевых функций

Таблица истинности, двухэлементная булева алгебра, алгебра логики, суперпозиция булевых функций, приоритет операций, эквивалентность формул булевой алгебры

Булевы функции могут быть заданы следующими способами:

1. С помощью таблицы истинности (значениями на каждой из интерпретаций).

2. Порядковым номером, который имеет эта функция.

3. Аналитически (в виде формулы).

Рассмотрим каждый из указанных способов подробнее.

4.2.1. Таблицы истинности

Таблицы, в которых каждой интерпретации (то есть набору аргументов) функции поставлено в соответствие ее значение, называются ***таблицами истинности булевой функции***.

В таблице истинности каждой переменной и значению самой функции соответствует по одному столбцу, а каждой интерпретации — по одной строке. Количество строк в таблице соответствует количеству различных интерпретаций функции. Булевы функции $\varphi(x)$, которые зависят от одной переменной, приведены в таблице 4.1.

Таблица 4.1. Булевы функции одной переменной

x	φ_0	φ_1	φ_2	φ_3
0	0	0	1	1
1	0	1	0	1

Каждой функции в соответствии с принимаемыми ей значениями можно присвоить следующие названия:

$\varphi_0 \equiv 0$ — функция константа 0,

$\varphi_1 = x$ — функция повторения аргумента,

$\varphi_2 = \bar{x}$ — функция инверсии или отрицания аргумента,

$\varphi_3 \equiv 1$ — функция константа 1.

Всевозможные булевы функции двух переменных $f(x, y)$ представлены в таблице 4.2, их количество равно $2^{2^2} = 16$.

Таблица 4.2. Булевы функции двух переменных

x	y	f_0	f_1	f_2	f_3	f_4	f_5	f_6	f_7	f_8	f_9	f_{10}	f_{11}	f_{12}	f_{13}	f_{14}	f_{15}
0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
0	1	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
1	0	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
1	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1

Большинство из шестнадцати булевых функций $f(x, y)$ часто применяются на практике. Поскольку данные функции используются как в математике, так и в программировании, они могут иметь различное обозначение. Обозначения, названия и прочтение булевых функций от двух переменных представлены в таблице 4.3.

Таблица 4.3. Булевы функции двух переменных

Функция	Обозначение	Название	Другие обозначения	Прочтение
1	2	3	4	5
$f_0(x, y)$	0	константа 0		константа 0
$f_1(x, y)$	$x \wedge y = xy$	конъюнкция (логическое «и»)	·, &, &&, *, AND, И, ×, min	x и y
$f_2(x, y)$	$\overline{x \to y}$	отрицание импликации	>	x и не y
$f_3(x, y)$	x	повторение первого аргумента		как x
$f_4(x, y)$	$\overline{x \leftarrow y}$	отрицание обратной импликации	<	не x и y
$f_5(x, y)$	y	повторение второго аргумента		как y
$f_6(x, y)$	$x \oplus y$	исключающее «или» (сумма по модулю 2)	≠, <>, > <, !=, XOR	x не как y
$f_7(x, y)$	$x \vee y$	дизъюнкция (логическое «или»)	OR, ИЛИ, +, max	x или y
$f_8(x, y)$	$x \downarrow y$	отрицание дизъюнкции (стрелка Пирса)	$\overline{x \vee y}$	не x и не y
$f_9(x, y)$	$x \sim y$	эквивалентность	⇔, ≡, Eqv, =	x как y
$f_{10}(x, y)$	$\bar{y}$	отрицание второго аргумента	$\neg y$	не y

Продолжение таблицы 4.3

1	2	3	4	5
$f_{11}(x, y)$	$x \leftarrow y$	обратная импликация	$\subset$	x, если y (x или не y)
$f_{12}(x, y)$	$\bar{x}$	отрицание первого аргумента	$\neg x$	не x
$f_{13}(x, y)$	$x \rightarrow y$	импликация	$\supset$, $\Rightarrow$, Imp	если x, то y (не x или y)
$f_{14}(x, y)$	$x \mid y$	отрицание конъюнкции (штрих Шеффера)	$\overline{x \wedge y}$	не x или не y
$f_{15}(x, y)$	1	константа 1		константа 1

Обозначения Not, And, Or, Xor, Imp, Eqv используются в языке программирования Basic; обозначения !, &, != используются в языке C; обозначения $\neg$, $\wedge$, $\vee$ используются в системе Mathcad. Для краткости в примерах и выкладках мы будем опускать знак конъюкции и писать xy вместо $x \wedge y$.

4.2.2. Номера булевых функций и интерпретаций

Каждой функции присваивается порядковый номер в виде натурального числа, двоичный код которого представляет собой столбец значений функции в таблице истинности. Младшим разрядом считается самая нижняя строка (значение функции на интерпретации (1, 1, ..., 1)), а старшим — самая верхняя (значение функции на интерпретации (0, 0, ..., 0)). Указанный порядковый номер функции, как двоичный, так и десятичный, полностью определяет булеву функцию.

Каждой интерпретации булевой функции также присваивается свой номер — значение двоичного кода, который представляет собой интерпретация. Интерпретации, записанной в верхней строке таблицы истинности, присваивается номер 0, затем следует интерпретация номер 1 и т.д. В самой нижней строке расположена интерпретация с номером $2^n - 1$, где n — количество переменных, от которых зависит булева функция.

Пример. Найдем порядковый номер функции $f(x, y)$, принимающей следующие значения: $f(0, 0) = 1$, $f(0, 1) = 1$, $f(1, 0) = 0$, $f(1, 1) = 1$. Построим таблицу истинности для этой функции (таблица 4.4).

Таблица 4.4. Таблица истинности f(x, y)

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

Двоичный код, соответствующий значениям этой функции, — 1101. Переведем двоичное число 1101_2 в десятичную систему счисления. Для этого каждому разряду двоичного числа присвоим весовой коэффициент, кратный соответствующей степени числа 2, начиная с нижней строки: 2^0, 2^1, 2^2 и т.д. Умножив весовой коэффициент на соответствующую двоичную цифру и сложив полученные значения, найдем порядковый номер функции.

$$1101_2 = 1 \times 2^3 + 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 = 8 + 4 + 0 + 1 = 13_{10}.$$

Таким образом, десятичный номер данной функции — 13, то есть рассматриваемая функция соответствует импликации $f_{13}(x, y) = x \to y$ (см. $f_{13}(x, y)$ в таблице 4.3). Таким образом, функцию $f_{13}(x, y)$ можно задать с помощью двоичного кода, соответствующего ее двоичному номеру: $f_{13}(x, y)$ соответствует двоичному числу $(1101)_2$.

Пример. Построим таблицу истинности для бинарной функции с порядковым номером 14. Для этого найдем двоичное число, которое соответствует десятичному числу 14. Представив это число как сумму степеней числа 2, получим:

$$14_{10} = 8 + 4 + 2 = 1 \times 2^3 + 1 \times 2^2 + 1 \times 2^1 + 0 \times 2^0 = 1110_2.$$

Таким образом, $f_{14}(x, y)$ соответствует двоичному числу $(1110)_2$.

Построим искомую таблицу истинности, разместив полученное число в столбце значения функции таким образом, чтобы младший разряд оказался в нижней строке:

Таблица 4.5. Таблица истинности $f_{14}(x, y)$

x	y	$f_{14}(x, y)$
0	0	1
0	1	1
1	0	1
1	1	0

4.2.3. Булевы алгебры: общая, двухэлементная и логическая

В п. 3.5 введено понятие ***булевой алгебры*** $(A, \wedge, \vee, \neg, 0, 1)$ как ***дистрибутивной решетки с дополнением***, где $\wedge$, $\vee$ — бинарные операции, $\neg$ — унарная операция на множестве-носителе A, которые обладают определенными свойствами. Анализируя алгебраическую форму всех этих свойств (см. п. 3.5) и заменяя обозначения операций $\wedge$, $\vee$, $\neg$ на $\bullet$, $+$, $^-$ соответственно, мы выделяем следующий набор ***независимых свойств***, которые можно считать ***аксиомами*** или ***независимыми законами*** булевой алгебры:

1. Коммутативность: $a + b = b + a$, $a \bullet b = b \bullet a$.
2. Ассоциативность: $a + (b + c) = (a + b) + c$,
   $a \bullet (b \bullet c) = (a \bullet b) \bullet c$.
3. Дистрибутивность: $a + (b \bullet c) = (a + b) \bullet (a + c)$,
   $a \bullet (b + c) = (a \bullet b) + (a \bullet c)$.
4. Законы для нуля, единицы и отрицания:
   $$a + 0 = a, \quad a + \overline{a} = 1, \quad a \bullet 1 = 1, \quad a \bullet \overline{a} = 0.$$

Все остальные соотношения (некоторые из которых также называются «законами» булевой алгебры) являются следствием указанных выше. В частности, из 1–4 следуют:

5. Законы идемпотентности: $a + a = a \bullet a = a$.
6. Другие свойства единицы и нуля: $a + 1 = 1$, $a \bullet 0 = 0$.
7. Законы поглощения: $a \bullet (a + b) = a + (a \bullet b) = a$.
8. Закон инволюции (двойного отрицания): $a = \overline{\overline{a}}$.
9. Ззаконы де Моргана (двойственности): $\overline{a + b} = \overline{a} \bullet \overline{b}$;
   $\overline{a \bullet b} = \overline{a} + \overline{b}$.

Таким образом, определение булевой алгебры из п. 3.5 выглядит так (после «алгебраической расшифровки» участвующих в нем понятий):

> **Определение**
> ***Булева алгебра*** — это алгебраическая структура $(A, \bullet, +, ^-, 0, 1)$ с бинарными операциями $\bullet$, $+$: $A^2 \to A$, унарной операцией «$^-$»: $A \to A$ и выделенными элементами $0, 1$ в носителе A, которые удовлетворяют свойствам 1–4.

Обозначение операций $\wedge$, $\vee$ вместо $\bullet$, $+$ в специальных булевых алгебрах заимствовано из логики, в которой Дж. Буль впервые выделил алгебраическую структуру со свойствами 1–4. Носитель этой структуры $B = \{0, 1\}$ состоит из двух элементов.

Определение
Алгебраическая структура $(B, \wedge, \vee, \bar{\ })$, $B = \{0, 1\}$, где операция $\wedge$ есть ***конъюнкция*** $f_1(x, y)$, $\vee$ есть ***дизъюнкция*** $f_7(x, y)$ (см. таблицы 4.2, 4.3), «$\bar{\ }$» есть ***отрицание*** или ***инверсия*** $\varphi_2(x)$ (см. таблицу 4.1), называется ***двухэлементной булевой алгеброй***.

Выполнение законов булевой алгебры 1–4 в структуре $(B, \wedge, \vee, \bar{\ })$, а также законов 5–9 следует из определения операций конъюнкции, дизъюнкции и отрицания через таблицы истинности 4.1, 4.2 (см. п. 4.4).

Определение
Алгеброй логики называется двухэлементная булева алгебра $(B, \wedge, \vee, \bar{\ }, \to, \sim)$, $B = \{0, 1\}$, в которой множество операций дополнено двумя бинарными операциями: ***импликацией*** $\to (f_{13}(x, y))$ и ***эквивалентностью*** $\sim (f_9(x, y))$ (см. таблицы 4.2, 4.3).

4.2.4. Булевы формулы и приоритет операций

Булевы функции могут быть заданы аналитически, т.е. формулами.

Определение
Формула — это выражение, содержащее булевы функции и их суперпозиции.

Определение
Суперпозицией называется прием получения новых функций путем подстановки значений одних функций вместо значений аргументов других функций. Точнее, ***суперпозицией*** булевой функции f_0 и функций $f_1, \ldots, f_n$ называется функция $f(x_1, \ldots, x_m) = f_0(g_1(x_1, \ldots, x_m), \ldots, g_k(x_1, \ldots, x_m))$, где каждая из функций $g_i(x_1, \ldots, x_m)$ совпадает с одной из функций $f_1, \ldots, f_n$, $i \in \{1, \ldots, k\}$. При этом некоторые из функций f_i, а значит и g_i могут тождественно совпадать с одной из переменных x_t, $t \in \{1, \ldots, m\}$.

Пример. Рассмотрим формулу, задающую функцию $f(x, y, z)$ следующим образом:

$$f(x, y, z) = (x \wedge \bar{y}) \vee z.$$

Эта формула содержит функции: $g(x_1)$ — отрицание, $s(x_1, x_2)$ — конъюнкция и $l(x_1, x_2)$ — дизъюнкция. Данную формулу можно

представить в виде суперпозиции указанных функций следующим образом:

$$f(x, y, z) = l(s(x, g(y)), z).$$

В формуле $(x \wedge \overline{y}) \vee z$ использована инфиксная запись, когда знаки функций (операций) расположены между операндами. Для записи выражений в инфиксной форме необходимо определить порядок выполнения операций, что производится с помощью скобок или с помощью задания приоритета операций.

Если в формуле отсутствуют скобки, то операции выполняются в следующей последовательности: отрицание, конъюнкция, дизъюнкция, импликация и эквивалентность:

$$\overline{\ }, \wedge, \vee, \rightarrow, \sim.$$

Пример. Предыдущую формулу $f(x, y, z) = (x \wedge \overline{y}) \vee z$ с учетом приоритета операций можно записать без скобок:

$$(x \wedge \overline{y}) \vee z = x \wedge \overline{y} \vee z.$$

Обратно с учетом приоритета операций расставим скобки:

$$x \sim z \rightarrow \overline{x} \vee z = x \sim (z \rightarrow (\overline{x} \vee z)).$$

Необходимо отметить, что знак операции отрицания может располагаться не только над отдельными переменными, но и над формулами или их частями. В этом случае операции выполняются так, как если бы выражение, находящееся под знаком отрицания, было заключено в скобки.

Пример. В формуле $x \rightarrow \overline{z \wedge y \vee x} \vee z$ скобки можно расставить следующим образом:

$$x \rightarrow (((\overline{z \wedge y}) \vee x) \vee z).$$

В отличие от табличного задания, представление функции формулой не единственно. Например, функцию штрих Шеффера можно представить с помощью основных операций булевой алгебры формулами:

$$f_{14} = \overline{x}_1 \vee \overline{x}_2 \quad \text{или} \quad f_{14} = \overline{x_1 x_2},$$

а функцию стрелка Пирса следующим образом:

$$f_8 = \overline{x}_1 \overline{x}_2 \quad \text{или} \quad f_8 = \overline{x_1 \vee x_2}.$$

Определение

Формулы, представляющие одну и ту же функцию, называются ***эквивалентными*** или ***равносильными***.

Эквивалентность формул обозначается знаком равенства. Один из способов установить эквивалентность формул состоит

в следующем: для каждой формулы строится таблица истинности, а затем полученные таблицы сравниваются, т.е. фактически для каждого набора переменных проверяется, равны ли на нем значения функций. Существуют и другие способы проверки эквивалентности формул, о которых будет сказано позже.

4.2.5. Переход от формулы к таблице истинности функции

Построим таблицу истинности для функции, заданной следующей формулой:

$$f(x, y, z, t) = x \vee \overline{y}(z \vee \overline{xy}) \to t.$$

Функция зависит от четырех переменных и, следовательно, для нее имеется $2^4 = 16$ интерпретаций. Для построения таблицы истинности необходимо определить значения функции на всех интерпретациях:

$$f(0, 0, 0, 0) = 0 \vee \overline{0}(0 \vee \overline{00}) \to 0 = 0 \vee 1(0 \vee \overline{0}) \to 0 =$$
$$= 0 \vee 1(0 \vee 1) \to 0 = 0 \vee 1(1) \to 0 = 0 \vee 1 \to 0 = 1 \to 0 = 0;$$

$$f(0, 0, 0, 1) = 0 \vee \overline{0}(0 \vee \overline{00}) \to 1 = 0 \vee 1(0 \vee \overline{0}) \to 1 =$$
$$= 0 \vee 1(0 \vee 1) \to 1 = 0 \vee 1(1) \to 1 = 0 \vee 1 \to 1 = 1 \to 1 = 1;$$

$$f(0, 0, 1, 0) = 0 \vee \overline{0}(1 \vee \overline{00}) \to 0 = 0 \vee 1(1 \vee \overline{0}) \to 0 =$$
$$= 0 \vee 1(1 \vee 1) \to 0 = 0 \vee 1(1) \to 0 = 0 \vee 1 \to 0 = 1 \to 0 = 0;$$

. . .

$$f(1, 1, 1, 1) = 1 \vee \overline{1}(1 \vee \overline{11}) \to 1 = 1 \vee 0(1 \vee \overline{1}) \to 1 =$$
$$= 1 \vee 0(1 \vee 0) \to 1 = 1 \vee 0(1) \to 1 = 1 \vee 0 \to 1 = 1 \to 1 = 1.$$

Разместим в таблице истинности интерпретации в порядке возрастания соответствующих им двоичных номеров и запишем полученное значение функции на каждой интерпретации. Получим таблицу истинности функции $f(x, y, z, t) = x \vee \overline{y}\,(z \vee \overline{xy}) \to t$ (таблица 4.6).

Таблица 4.6. Таблица истинности f(x, y, z, t)

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

Продолжение таблицы 4.6

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

Вопросы

1. Перечислите способы задания булевых функций.
2. Каким образом строится таблица истинности булевой функции?
3. Перечислите основные булевы функции от двух переменных.
4. Каким образом определяется номер булевой функции? Номер интерпретации?
5. Дайте определение суперпозиции булевых функций.
6. Какой приоритет определен для операций алгебры логики? Для какой цели служит приоритет операций?
7. Какие формулы называются эквивалентными?
8. Каким образом осуществляется переход от формулы к таблице истинности функции?

Задания

1. Опустите максимально возможное число скобок в формуле с учетом приоритета выполнения операций:
   а) $(((x \sim y) \sim ((((x \wedge z) \wedge t) \vee (\overline{x})) \to y)) \vee y)$;
   б) $((x \to z) \to (((y \sim (\overline{z})) \wedge t)) \vee ((\overline{t} \wedge x) \sim y))$;
   в) $(y \wedge (\overline{z}) \vee (((\overline{x}) \to z)) \vee ((\overline{t} \wedge y)) \vee ((\overline{y}) \sim (t \vee x)))$.
2. Определите интерпретации, на которых выполняются соотношения:
   а) $x \wedge (x \to y) \to (x \to z) = 0$;
   б) $(x \to y) \wedge (t \to x) \vee \overline{t} = 1$;
   в) $(\overline{x} \downarrow y) \wedge (x \sim y) = 1$.
3. Постройте таблицы истинности следующих функций и определите их порядковый номер:
   а) $f(x, y) = (x \to y) \wedge (y \to x)$;
   б) $f(x, y, z) = (x \wedge y) \oplus (x \wedge z) \oplus (y \wedge z)$;
   в) $f(x, y, z) = (x \wedge y) \vee (x \wedge z) \vee (y \wedge z)$;
   г) $f(x, y, z) = (x \wedge \overline{y}) \vee (x \wedge \overline{z}) \vee (y \wedge \overline{z})$;

д) $f(x, y) = x \mid (x \mid y)$;
е) $f(x, y) = (x \mid y) \mid (x \mid y)$.

4. Проверьте с помощью таблиц истинности, справедливы ли следующие соотношения:
а) $x \vee (y \sim z) = (x \vee y) \sim (x \vee z)$;
б) $x \to (y \sim z) = (x \to y) \sim (x \to z)$;
в) $x \wedge (y \sim z) = (x \wedge y) \sim (x \wedge z)$;
г) $x \to (y \vee z) = (x \to y) \vee (x \to z)$;
д) $x \to (y \wedge z) = (x \to y) \wedge (x \to z)$;
е) $x \oplus (y \wedge z) = (x \oplus y) \wedge (x \oplus z)$;
ж) $x \to (y \to z) = (x \to y) \to (x \to z)$.

4.3. Двойственность

Двойственные и самодвойственные булевы функции, принцип двойственности

В основе булевой алгебры, как и некоторых других алгебраических структурах лежит принцип двойственности.

Определение
Функция $f^*(x_1, \ldots, x_n)$ называется ***двойственной*** к функции $f(x_1, \ldots, x_n)$, если

$$f^*(x_1, \ldots, x_n) = \overline{f}(\overline{x}_1, \ldots, \overline{x}_n). \qquad (4.1)$$

Докажем, что $(f^*)^* = f$, для этого по формуле 4.1 найдем функцию, двойственную f^* и, используя закон двойного отрицания $f = \overline{\overline{f}}$ (подробно будет рассмотрен ниже в п. 4.4), получим:

$$(f^*(x_1, \ldots, x_n))^* = (\overline{f}(\overline{x}_1, \ldots, \overline{x}_n))^* = (\overline{\overline{f}}(\overline{\overline{x}}_1, \ldots, \overline{\overline{x}}_n)) = f(x_1, \ldots, x_n).$$

Таким образом, отношение двойственности между функциями симметрично. Из определения двойственности следует, что для любой функции двойственная ей определяется однозначно.

Определение
Функция, двойственная сама себе, т.е. $f = f^*$, называется ***самодвойственной***.

Совместим таблицы истинности булевой функции $f(x, y)$ и двойственной функции $f^*(x, y)$, учитывая в столбце значений f^* равенства $\overline{f}(\overline{0}, \overline{0}) = \overline{f}(1, 1)$, $\overline{f}(\overline{1}, \overline{0}) = \overline{f}(0, 1)$ и т.д. В столбцах f и $f*$ совмещенной таблицы 4.7 каждое значение функции $f(x, y)$ равно отрицанию симметричного ему значения функции $f^*(x, y)$

относительно горизонтальной линии середины таблицы (в таблице 4.7 симметричные значения соединены стрелками).

Таблица 4.7. Сравнение двойственных функций

x	y	$f(x, y)$	$f^*(x, y)$
0	0	$f(0, 0)$	$\overline{f}(1, 1)$
0	1	$f(0, 1)$	$\overline{f}(1, 0)$
1	0	$f(1, 0)$	$\overline{f}(0, 1)$
1	1	$f(1, 1)$	$\overline{f}(0, 0)$

Таким образом, чтобы построить таблицу истинности функции, двойственной данной, необходимо построить таблицу истинности заданной функции, каждое значение булевой функции заменить на противоположное и записать полученный столбец в обратной последовательности.

Пример. Найти функцию, двойственную функции $f(x, y, z)$, если известно, что $f(x, y, z) = 1$ только на интерпретациях (001), (011), (111).

Для столбца значений f генерируем набор противоположных (инверсных) значений (10101110). Записав его в обратной последовательности, получим таким образом столбец значения двойственной функции f^* (таблица 4.8).

Таблица 4.8. Таблица истинности двойственных функций

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

Рассмотрим таблицу истинности самодвойственной функции (таблица 4.9).

Таблица 4.9. Самодвойственная функция

x	y	$f(x, y) = f^*(x, y)$
0	0	$f(0, 0) = \overline{f}(1, 1)$
0	1	$f(0, 1) = \overline{f}(1, 0)$
1	0	$f(1, 0) = \overline{f}(0, 1)$
1	1	$f(1, 1) = \overline{f}(0, 0)$

Проведем линию симметрии посередине таблицы. Как видно из таблицы 4.9, каждое значение функции равно отрицанию симметричного ему значения. Таким образом, по таблице истинности функции всегда можно определить, является данная функция самодвойственной или нет.

Пример. Функции $f(x, y, z)$ и $g(x, y, z)$ заданы таблицами истинности (таблица 4.10). Определить, являются ли данные функции самодвойственными.

Таблица 4.10. Функции $f(x, y, z)$ и $g(x, y, z)$

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

Из таблицы 4.10 видно, что каждое значение функции $f(x, y, z)$ является отрицанием симметричного ему значения. Следовательно, функция $f(x, y, z)$ является самодвойственной. Для функции $g(x, y, z)$ имеются значения функции, не равные отрицанию симметричных им значений, например: $g(0, 0, 0) = 0$, а симметричное ему значение $g(1, 1, 1) = 0$. Следовательно, функция $g(x, y, z)$ не является самодвойственной.

Значения самодвойственной функции полностью определяются ее значениями на половине интерпретаций. Всего имеется 2^n интерпретаций функции n переменных. Половина данного количества будет составлять $2^n/2 = 2^{n-1}$. Таким образом, количество самодвойственных функций от n переменных равняется $2^{2^{n-1}}$.

Пусть функция F заданна как суперпозиция функций f_0 и функций $f_1, \ldots, f_n$: $F = f_0(f_1, \ldots, f_n)$. Функцию F^*, двойственную F, можно получить, заменив в формуле F функции $f_0; f_1, \ldots, f_n$ на двойственные им $f_0^*; f_1^*, \ldots, f_n^*$.

Действительно, если внутренние функции f_i, $i = \overline{1, n}$ зависят от m аргументов $x_1, \ldots, x_m$, то, по определению, для результирующей функции $F = F(x_1, \ldots, x_m)$ двойственная F^* имеет вид:

$$F^* = \overline{F}(\overline{x}_1, \ldots, \overline{x}_m) = \overline{f}_0(f_1(\overline{x}_1, \ldots, \overline{x}_m), \ldots, f_n(\overline{x}_1, \ldots, \overline{x}_m)),$$

причем

$$f_i(\overline{x}_1, \ldots, \overline{x}_m) = \overline{\overline{f}}_i(\overline{x}_1, \ldots, \overline{x}_m) = \overline{f}_i^*(x_1, \ldots, x_m).$$

С другой стороны,

$$f_0^*(f_1^*, \ldots, f_n^*) = \overline{f}_0(\overline{f}_1^*, \ldots, \overline{f}_n^*) = \overline{f}_0(f_1(\overline{x}_1, \ldots, \overline{x}_m), \ldots, f_n(\overline{x}_1, \ldots, \overline{x}_m)).$$

Таким образом,

$$F^* = f_0^*(f_1^*, \ldots, f_n^*).$$

Методом математической индукции это правило перехода к двойственной функции распространяется на «многоступенчатые» суперпозиции булевых функций, когда «внутренние» функции $f_1, \ldots, f_n$, в свою очередь, являются суперпозициями других функций и т.д.

Укажем функции, двойственные к «элементарным» функциям логики $\wedge$, $\vee$, $\overline{\ }$, константа 0, константа 1:

$$f(x, y) = x \wedge y; \qquad f^*(x, y) = \overline{\overline{x} \wedge \overline{y}} = x \vee y;$$

$$f(x) = \overline{x}; \qquad f^*(x) = \overline{\overline{\overline{x}}} = \overline{x} = f(x);$$

$$f(x) = 0; \qquad f^*(x) = \overline{0} = 1 = \overline{f(x)}.$$

Поскольку отношение двойственности симметрично, можно сказать, что конъюнкция двойственна дизъюнкции, дизъюнкция двойственна конъюнкции, функция «0» двойственна функции «1», и наоборот, а отрицание — самодвойственная функция.

Отсюда следует так называемый ***принцип двойственности***, указывающий правило получения двойственных формул в булевой алгебре: «Для того, чтобы получить двойственную формулу булевой алгебры необходимо заменить в ней все конъюнкции на дизъюнкции, дизъюнкции на конъюнкции, 0 на 1, 1 на 0 и использовать скобки, где необходимо, чтобы порядок выполнения операций остался прежним».

Пример. Найти функцию, двойственную функции $f = x \vee \vee \overline{y}z \vee 0$.

Воспользуемся приведенным выше правилом получения двойственных функций:

$$f^* = (x \vee (\bar{y}z) \vee 0)^* = x \wedge (\bar{y} \vee z) \wedge 1.$$

Следствие. *Если функции равны, то и двойственные им функции также равны*:

$$\text{если } f_1 = f_2, \quad \text{то } f_1^* = f_2^*.$$

Вопросы

1. Дайте определение двойственной функции.
2. Какие функции называются самодвойственными?
3. Каким образом формируется таблица истинности двойственной функции?
4. Определите, используя таблицу истинности булевой функции, является ли она самодвойственной или нет.
5. Сформулируйте принцип двойственности.
6. Каким образом можно аналитическим путем получить из заданной формулой функции двойственную ей?

Задания

1. Определить количество наборов, на которых самодвойственная функция n переменных равна 1.
2. Найти двойственные формулы к следующим функциям:
   а) $(x \wedge (y \vee z)) \vee \bar{x} \wedge \bar{y}$;
   б) $x\bar{y} \vee yz \vee xz$;
   в) $x\bar{y} \vee x \vee y \vee zt$.
3. Определить, являются ли следующие функции самодвойственными:
   а) $f(x, y) = (\bar{x} \vee y) \wedge (\bar{y} \vee x)$;
   б) $f(x, y, z) = (x \wedge y) \vee (x \wedge z) \vee (y \wedge z)$;
   в) $f(x, y, z) = (\bar{x} \wedge \bar{y}) \vee (x \wedge z) \vee (\bar{y} \wedge \bar{z})$;
   г) $f(x, y, z) = (x \wedge \bar{y}) \vee (x \wedge \bar{z}) \vee (y \wedge \bar{z})$;
   д) $f(x, y) = x \vee (\overline{x \wedge y})$.

4.4. Законы булевой алгебры

В этой главе будет проверено выполнение законов булевой алгебры 1–4, а также 5–9 из п. 4.2.3 для двухэлементной алгебраической структуры $(B, \wedge, \vee, \bar{\ })$, где $\wedge$, $\vee$, $\bar{\ }$ есть операции конъюнкции f_1, дизъюнкции f_7 и отрицания φ_2, заданные таблицами истинности (таблицы 4.1, 4.2). В дальнейшм, если не оговорено противное, мы будем называть двухэлементную булеву алгебру $(B, \wedge, \vee, \bar{\ })$ просто «булевой алгеброй».

1. ***Коммутативность конъюнкции и дизъюнкции***

$$x \vee y = y \vee x; \quad x \wedge y = y \wedge x.$$

Совместим таблицы истинности для обеих частей первого равенства (таблица 4.11).

Таблица 4.11. Таблицы истинности

x	y	$x \vee y$	$y \vee x$
0	0	0	0
0	1	1	1
1	0	1	1
1	1	1	1

Столбцы $x \vee y$ и $y \vee x$ в таблицах истинности содержат одинаковые значения, что доказывает коммутативность операции дизъюнкции. Найдем тождество, двойственное данному, для чего заменим все функции на двойственные им:

$$(x \vee y)^* = (y \vee x)^*, \quad x \wedge y = y \wedge x.$$

2. ***Ассоциативность конъюнкции и дизъюнкции***

$$x \vee (y \vee z) = (x \vee y) \vee z; \quad x \wedge (y \wedge z) = (x \wedge y) \wedge z.$$

Составим таблицы истинности для левой и правой частей второго равенства (таблица 4.12).

Таблица 4.12. Доказательство ассоциативности конъюнкции

x	y	z	$y \wedge z$	$x \wedge (y \wedge z)$	$x \wedge y$	$(x \wedge y) \wedge z$
0	0	0	0	0	0	0
0	0	1	0	0	0	0
0	1	0	0	0	0	0
0	1	1	1	0	0	0
1	0	0	0	0	0	0
1	0	1	0	0	0	0
1	1	0	0	0	1	0
1	1	1	1	1	1	1

Столбцы, соответствующие левой и правой частям второго равенства, содержат одинаковые значения, что доказывает справедливость тождества $x \wedge (y \wedge z) = (x \wedge y) \wedge z$. Тождество, выражающее ассоциативность дизъюнкции, двойственно доказанному, так как

может быть получено из него путем замены всех функций на двойственные им, и, следовательно, оно тоже верно.

3. ***Дистрибутивность конъюнкции и дизъюнкции относительно друг друга***

$$x \wedge (y \vee z) = (x \wedge y) \vee (x \wedge z);$$

$$x \vee (y \wedge z) = (x \vee y) \wedge (x \vee z).$$

Докажем дистрибутивность конъюнкции относительно дизъюнкции, используя таблицу истинности (таблица 4.13).

Таблица 4.13. Дистрибутивность конъюнкции относительно дизъюнкции

x	y	z	$y \vee z$	$x \wedge (y \vee z)$	$x \wedge y$	$x \wedge z$	$(x \wedge y) \vee (x \wedge z)$
0	0	0	0	0	0	0	0
0	0	1	1	0	0	0	0
0	1	0	1	0	0	0	0
0	1	1	1	0	0	0	0
1	0	0	0	0	0	0	0
1	0	1	1	1	0	1	1
1	1	0	1	1	1	0	1
1	1	1	1	1	1	1	1

Столбцы, соответствующие левой и правой частям первого равенства, содержат одинаковые значения, что и доказывает справедливость равенства $x \wedge (y \vee z) = (x \wedge y) \vee (x \wedge z)$. Двойственное тождество выражает дистрибутивность дизъюнкции относительно конъюнкции.

4. ***Идемпотентность конъюнкции и дизъюнкции***

$$x \vee x = x; \quad x \wedge x = x.$$

Построим таблицы истинности унарных функций из левых частей равенств (таблица 4.14).

Таблица 4.14. Закон идемпотентности

x	$x \vee x$	$x \wedge x$
0	0	0
1	1	1

В таблице 4.14 значения всех столбцов одинаковы и совпадают со значением переменной x, что и доказывает оба тождества. Данные тождества так же, как и рассмотренные выше, являются двойственными друг другу.

5. ***Закон исключенного третьего***

$$x \vee \bar{x} = 1.$$

Докажем данный закон, используя таблицу истинности (таблица 4.15).

Таблица 4.15. Закон исключенного третьего

x	$\bar{x}$	$x \vee \bar{x}$
0	1	1
1	0	1

Столбец таблицы истинности, представляющий левую часть доказываемого тождества, равен константе единицы, что и требовалось доказать.

6. ***Закон противоречия***

$$x \wedge \bar{x} = 0.$$

Данный закон является двойственным к доказанному выше закону исключенного третьего.

7. ***Тождества с константами***

$$x \vee 0 = x; \quad x \wedge 1 = x; \quad x \vee 1 = 1; \quad x \wedge 0 = 0.$$

Построим таблицы истинности для левых частей тождеств (таблица 4.16).

Таблица 4.16. Таблицы истинности тождеств

x	$x \vee 0$	$x \wedge 1$	$x \vee 1$	$x \wedge 0$
0	0	0	1	0
1	1	1	1	0

Полученная таблица доказывает справедливость данных тождеств.

8. ***Законы элиминации***

$$x \wedge (x \vee y) = x; \quad x \vee (x \wedge y) = x.$$

Докажем данный закон аналитически, используя тождества с константами и дистрибутивный закон:

$$x \wedge (x \vee y) = (x \vee 0) \wedge (x \vee y) = x \vee (0 \wedge y) = x \vee 0 = x;$$

$$x \vee (x \wedge y) = (x \wedge 1) \vee (x \wedge y) = x \wedge (1 \vee y) = x \wedge 1 = x.$$

9. ***Закон двойного отрицания***

$$\bar{\bar{x}} = x.$$

Построим соответствующую таблицу истинности (таблица 4.17).

Таблица 4.17. Закон двойного отрицания

x	$\bar{x}$	$\bar{\bar{x}}$
0	1	0
1	0	1

Полученная таблица истинности доказывает справедливость тождества.

Следствие. *Если к некоторой части A формулы F булевой алгебры операция отрицания применена более чем один раз, то можно удалить любое четное число данных операций и значение формулы F не изменится.*

10. ***Законы де Моргана***

$$\overline{x \vee y} = \bar{x} \wedge \bar{y}; \quad \overline{x \wedge y} = \bar{x} \vee \bar{y}.$$

Докажем первое из приведенных тождеств с помощью таблицы истинности (таблица 4.18).

Таблица 4.18. Доказательство закона де Моргана

x	y	$x \vee y$	$\overline{x \vee y}$	$\bar{x}$	$\bar{y}$	$\bar{x} \wedge \bar{y}$
0	0	0	1	1	1	1
0	1	1	0	1	0	0
1	0	1	0	0	1	0
1	1	1	0	0	0	0

Построенная таблица истинности доказывает справедливость тождества. Второй закон де Моргана является двойственным первому и, значит, также является верным.

Вопросы

1. Перечислите основные законы булевой алгебры.
2. Какими способами можно доказать законы булевой алгебры?
3. Каким образом принцип двойственности применим к законам булевой алгебры?
4. Сформулируйте и запишите тождества для каждого из десяти законов булевой алгебры.

Задания

1. Упростите с помощью законов логики Буля нижеприведенные выражения. Затем с помощью таблиц истинности сравните полученные выражение с исходными:
   а) $(x \vee (\bar{t} \wedge y)) \wedge ((\bar{x} \wedge (\bar{y} \vee t)) \vee z)) \vee \bar{z} \vee (x \vee (y \wedge \bar{t}))$;
   б) $((x \vee z) \wedge (x \vee t)) \wedge (((z \vee (z \wedge y)) \wedge \bar{z}) \vee \bar{x})$;
   в) $(\bar{y} \vee t) \wedge ((\bar{x} \wedge z) \vee (x \wedge z) \vee (\bar{t} \wedge \bar{z}) \vee (x \wedge \bar{z})) \wedge (y \vee t)$;

г) $(x \vee \overline{z}) \wedge (\overline{x} \vee \overline{y}) \wedge (\overline{y} \vee z) \wedge (\overline{x} \vee y) \wedge (y \vee z)$;
д) $(x \wedge z) \vee ((y \vee \overline{t}) \wedge (\overline{x} \vee \overline{t}) \wedge (t \vee y) \wedge (\overline{x} \vee t) \vee (x \wedge \overline{z}))$;
е) $((\overline{y} \vee \overline{z}) \wedge (x \vee y)) \vee (t \wedge \overline{z}) \vee (((\overline{y} \wedge \overline{x}) \vee z) \wedge (x \vee y))$;
ж)$(x \wedge \overline{z}) \vee (\overline{x} \wedge \overline{y}) \vee (y \wedge z) \vee (\overline{x} \wedge y) \vee (z \wedge \overline{y})$;
з) $((x \vee (z \vee (y \wedge z))) \wedge (\overline{z \wedge t}) \wedge (z \wedge \overline{t})) \wedge (z \vee (\overline{t} \wedge \overline{z}) \vee t)$;
и) $((x \vee \overline{x}) \wedge (\overline{y} \vee \overline{t}) \wedge (\overline{y} \vee \overline{z}) \wedge (\overline{z} \vee t)) \vee ((\overline{y} \vee z) \wedge (z \vee t))$;
к) $(x \vee \overline{z}) \wedge ((\overline{x} \wedge t) \vee (y \wedge t) \vee (\overline{x} \wedge \overline{t}) \vee (y \wedge \overline{t})) \wedge (x \vee z)$.

4.5. Дизъюнктивные и конъюктивные разложения булевых функций

Теоремы разложения, элементарные конъюнкция и дизъюнкция, конституенты нуля и единицы, нормальные формы

Среди множества эквивалентных формул, представляющих выбранную булеву функцию f, выделяется одна формула, которая называется ***совершенной нормальной формой*** функции f. Она имеет регламентированную логическую структуру и однозначно определяется по функции f (п. 4.6), а ее построение основано на рекуррентном применении теорем о специальных разложениях булевой функции по переменным, которые приводятся в этом разделе.

Для упрощения математических выкладок введем двоичный параметр σ и обозначение x^σ следующим образом:

$$x, \sigma \in B = \{0, 1\},$$

$$x^\sigma = \begin{cases} \overline{x}, & \text{если } \sigma = 0, \\ x, & \text{если } \sigma = 1. \end{cases}$$

Можем сделать вывод, что

$$x^\sigma = \begin{cases} 1, & x = \sigma, \\ 0, & x \neq \sigma. \end{cases}$$

Непосредственно проверяется, что бинарная функция $f(x, \sigma) = x^\sigma$ представляется формулой:

$$x^\sigma = x\,\sigma \vee \overline{x}\,\overline{\sigma}. \qquad (4.2)$$

Теорема 1. *О дизъюнктивном разложении булевой функции $f(x_1, x_2, \ldots, x_n)$ по k переменным*

Любую булеву функцию $f(x_1, x_2, \ldots, x_n)$ можно представить в следующей форме:

$$f(x_1, \ldots, x_k, x_{k+1}, \ldots, x_n) =$$

$$= \bigvee_{(\sigma_1, \sigma_2, \ldots, \sigma_k)} x_1^{\sigma_1} \wedge x_2^{\sigma_2} \wedge \ldots \wedge x_k^{\sigma_k} \wedge f(\sigma_1, \sigma_2, \ldots, \sigma_k, x_{k+1}, \ldots, x_n). \quad (4.3)$$

Запись $\bigvee_{(\sigma_1, \sigma_2, \ldots, \sigma_k)}$ означает многократную дизъюнкцию, которая берется по всем возможным наборам значений $(\sigma_1, \sigma_2, \ldots, \sigma_k)$ при любом k $(1 \le k \le n)$.

□ *Доказательство.* Необходимо проверить, что на произвольной интерпретации $(a_1, a_2, \ldots a_n)$ левая и правая части равенства (4.3) принимают одинаковые значения. Подставив значения $a_1, a_2, \ldots a_n$ вместо $x_1, x_2, \ldots, x_n$, получим:

$$f(a_1, a_2, \ldots a_n) =$$

$$= \bigvee_{(\sigma_1, \sigma_2, \ldots, \sigma_k)} a_1^{\sigma_1} \wedge a_2^{\sigma_2} \wedge \ldots \wedge a_k^{\sigma_k} \wedge f(\sigma_1, \sigma_2, \ldots, \sigma_k, a_{k+1}, \ldots, a_n).$$

По формуле (4.2) можем записать, что

$$a_i^{\sigma_i} = \begin{cases} 1, & a_i = \sigma_i \\ 0, & a_i \ne \sigma_i \end{cases}$$

Конъюнкция $a_1^{\sigma_1} \wedge a_2^{\sigma_2} \wedge \ldots \wedge a_k^{\sigma_k}$ равна нулю, если хотя бы один элемент конъюнкции $a_i^{\sigma_i}$ равен нулю, что произойдет, если $a_i \ne \sigma_i$. Если же $(a_1, a_2, \ldots a_k)$ совпадает с $(\sigma_1, \sigma_2, \ldots, \sigma_k)$, то $a_1^{\sigma_1} \wedge a_2^{\sigma_2} \wedge \ldots \wedge \wedge a_k^{\sigma_k} = 1$.

Отсюда следует, что

$$a_1^{\sigma_1} \wedge a_2^{\sigma_2} \wedge \ldots \wedge a_k^{\sigma_k} \wedge f(\sigma_1, \sigma_2, \ldots, \sigma_k, a_{k+1}, \ldots, a_n) =$$

$$= 0 \wedge f(\sigma_1, \sigma_2, \ldots, \sigma_k, a_{k+1}, \ldots, a_n) = 0$$

при $(a_1, a_2, \ldots a_k) \ne (\sigma_1, \sigma_2, \ldots, \sigma_k)$.

Если $(a_1, \ldots a_k) = (\sigma_1, \ldots, \sigma_k)$, то

$$a_1^{\sigma_1} \wedge a_2^{\sigma_2} \wedge \ldots \wedge a_k^{\sigma_k} \wedge f(\sigma_1, \sigma_2, \ldots, \sigma_k, a_{k+1}, \ldots, a_n) =$$

$$= 1 \wedge f(\sigma_1, \sigma_2, \ldots, \sigma_k, a_{k+1}, \ldots, a_n) = f(a_1, a_2, \ldots a_n).$$

Исходя из этого, запишем значение правой части тождества (4.3):

$$\bigvee_{(\sigma_1, \sigma_2, \ldots, \sigma_k)} a_1^{\sigma_1} \wedge a_2^{\sigma_2} \wedge \ldots \wedge a_k^{\sigma_k} \wedge f(\sigma_1, \sigma_2, \ldots, \sigma_k, a_{k+1}, \ldots, a_n) =$$

(распишем многократную дизъюнкцию $\bigvee$ на наборах $(\sigma_1, \sigma_2, \ldots, \sigma_k)$)

$$= 0 \vee \ldots \vee 0 \vee f(a_1, a_2, \ldots a_n) \vee 0 \vee \ldots \vee 0 = f(a_1, a_2, \ldots a_n).$$

Итак, значение правой части тождества (4.3) совпадает со значением левой части на произвольной интерпретации. Теорема доказана. ■

Пример. Запишем дизъюнктивное разложение функции $f(x, y, z, t) = (\overline{x \wedge y \vee \bar{z}}) \wedge t$ по переменным x, z.

Решение. Воспользуемся теоремой о разложении:

$$f(x, y, z, t) = \bigvee_{(\sigma_1, \sigma_2)} z_1^{\sigma_1} \wedge z_2^{\sigma_2} \wedge f(\sigma_1, y, \sigma_2, t) =$$

$$= \bar{x} \wedge \bar{z} \wedge f(0, y, 0, t) \vee \bar{x} \wedge z \wedge f(0, y, 1, t) \vee$$

$$\vee x \wedge \bar{z}z \wedge f(1, y, 0, t) \vee x \wedge z \wedge f(1, y, 1, t).$$

Вычислим:

$$f(0, y, 0, t) = (\overline{0 \wedge y \vee \bar{0}}) \wedge t = 0,$$

$$f(0, y, 1, t) = (\overline{0 \wedge y \vee \bar{1}}) \wedge t = t,$$

$$f(1, y, 0, t) = (\overline{1 \wedge y \vee \bar{0}}) \wedge t = 0,$$

$$f(1, y, 1, t) = (\overline{1 \wedge y \vee \bar{1}}) \wedge t = \bar{y} \wedge t.$$

Подставим полученные значения $f(0, y, 0, t)$, $f(0, y, 1, t)$, $f(1, y, 0, t)$, $f(1, y, 1, t)$ в формулу дизъюнктивного разложения по переменным x, z:

$$f(x, y, z, t) = \bar{x} \wedge \bar{z} \wedge 0 \vee \bar{x} \wedge z \wedge t \vee x \wedge \bar{z} \wedge 0 \vee x \wedge z \wedge (\bar{y} \wedge t) =$$

$$= \bar{x} \wedge z \wedge t \vee x \wedge z \wedge \bar{y} \wedge t.$$

Рассмотрим следствия из теоремы 1.

Следствие 1. *Дизъюнктивное разложение булевой функции $f(x_1, x_2, ..., x_n)$ по одной переменной.*

Любую булеву функцию $f(x_1, x_2, ..., x_n)$ можно представить в следующей форме:

$$f(x_1, x_2, ..., x_n) = \bigvee_{\sigma_i} x_i^{\sigma_i} \wedge f(x_1, x_2, ..., x_{i-1}, \sigma_i, x_{i+1}, ..., x_n). \quad (4.4)$$

Запись означает, что дизъюнкция берется по всем значениям σ_i, то есть 0 и 1. Запись $f(x_1, x_2, ..., x_{i-1}, \sigma_i, x_{i+1}, ..., x_n)$ обозначает значение функции на наборе $x_1, x_2, ..., x_n$, где вместо значения переменной x_i подставлено σ_i.

Пример. Рассмотрим функцию $f(x, y, z, t) = (\overline{xy \vee \bar{z}})t$. Осуществить дизъюнктивное разложение этой функции по переменной x.

Решение. Воспользуемся следствием теоремы о разложении:

$$f(x, y, z, t) = \bar{x} \wedge f(0, y, z, t) \vee x \wedge f(1, y, z, t).$$

Вычислим значение функции $f(x, y, z, t)$ при $x = 0$ и $x = 1$:

$$f(0, y, z, t) = (\overline{0 \wedge y \vee \bar{z}})t = zt;$$

$$f(1, y, z, t) = (\overline{1 \wedge y \vee \bar{z}})t = \bar{y} \wedge z \wedge t.$$

Подставим полученные значения $f(0, y, z, t)$ и $f(1, y, z, t)$ в формулу дизъюнктивного разложения:

$$f(x, y, z, t) = \bar{x} \wedge f(0, y, z, t) \vee x \wedge f(1, y, z, t) = \bar{x}zt \vee x\bar{y}zt.$$

Следствие 2. *Дизъюнктивное разложение булевой функции* $f(x_1, x_2, ..., x_n)$ *по всем n переменным.*

Любую булеву функцию $f(x_1, x_2, ..., x_n) \neq 0$ можно представить в следующей форме:

$$f(x_1, x_2, ..., x_n) = \bigvee_{\substack{(\sigma_1, \sigma_2, ..., \sigma_n) \\ f(\sigma_1, \sigma_2, ..., \sigma_n) = 1}} x_1^{\sigma_1} \wedge x_2^{\sigma_2} \wedge ... \wedge x_n^{\sigma_n}. \qquad (4.5)$$

Запись $\bigvee_{\substack{(\sigma_1, \sigma_2, ..., \sigma_n) \\ f(\sigma_1, \sigma_2, ..., \sigma_n) = 1}}$ означает, что дизъюнкция берется по всем наборам значений $(\sigma_1, \sigma_2, ..., \sigma_n)$, на которых $f(\sigma_1, \sigma_2, ..., \sigma_n) = 1$.

Действительно, запишем соотношение (4.3) для случая $k = n$:

$$f(x_1, x_2, ..., x_n) = \bigvee_{(\sigma_1, \sigma_2, ..., \sigma_n)} x_1^{\sigma_1} \wedge x_2^{\sigma_2} \wedge ... \wedge x_n^{\sigma_n} \wedge f(\sigma_1, \sigma_2, ..., \sigma_n).$$

Заметим следующее:

если $f(\sigma_1, \sigma_2, ..., \sigma_n) = 0$,

то $x_1^{\sigma_1} \wedge x_2^{\sigma_2} \wedge ... \wedge x_n^{\sigma_n} \wedge f(\sigma_1, \sigma_2, ..., \sigma_n) = 0$;

если $f(\sigma_1, \sigma_2, ..., \sigma_n) = 1$,

то $x_1^{\sigma_1} \wedge x_2^{\sigma_2} \wedge ... \wedge x_n^{\sigma_n} \wedge f(\sigma_1, \sigma_2, ..., \sigma_n) = x_1^{\sigma_1} \wedge x_2^{\sigma_2} \wedge ... \wedge x_n^{\sigma_n}$.

Поэтому дизъюнктивное разложение по всем переменным содержит только конъюнкции, соответствующие наборам $(\sigma_1, \sigma_2, ..., \sigma_n)$, для которых $f(\sigma_1, \sigma_2, ..., \sigma_n) = 1$.

Пример. Рассмотрим функцию $f(x, y, z) = xy \vee \bar{z}$. Осуществить дизъюнктивное разложение этой функции по всем переменным.

Решение. Определим значение функции на каждой из интерпретаций:

$$f(0, 0, 0) = 0 \wedge 0 \vee \bar{0} = 0 \vee 1 = 1,$$

$$f(0, 0, 1) = 0 \wedge 0 \vee \bar{1} = 0 \vee 0 = 0,$$

$$f(0, 1, 0) = 0 \wedge 1 \vee \bar{0} = 0 \vee 1 = 1,$$

$$f(0, 1, 1) = 0 \wedge 1 \vee \bar{1} = 0 \vee 0 = 0,$$

$$f(1, 0, 0) = 1 \wedge 0 \vee \overline{0} = 0 \vee 1 = 1,$$
$$f(1, 0, 1) = 1 \wedge 0 \vee \overline{1} = 0 \vee 0 = 0,$$
$$f(1, 1, 0) = 1 \wedge 1 \vee \overline{0} = 1 \vee 1 = 1,$$
$$f(1, 1, 1) = 1 \wedge 1 \vee \overline{1} = 1 \vee 0 = 1.$$

Используя формулу (4.5), получим:

$$f(x, y, z) = x^0 y^0 z^0 \vee x^0 y^1 z^0 \vee x^1 y^0 z^0 \vee x^1 y^1 z^0 \vee x^1 y^1 z^1 =$$
$$= \overline{x}\,\overline{y}\,\overline{z} \vee \overline{x}\,y\,\overline{z} \vee x\,\overline{y}\,\overline{z} \vee x\,y\,\overline{z} \vee x\,y\,z.$$

Определение
Элементарной конъюнкцией называется конъюнкция любого числа булевых переменных, взятых с отрицанием или без него, в которой каждая переменная встречается не более одного раза. Элементарной конъюнкцией, содержащей ноль переменных, будем считать константу 1.

Пример. Элементарными конъюнкциями для функции от одной переменной могут быть y, $\overline{z}$, для функции от двух переменных — $\overline{x} \wedge y$, $x \wedge \overline{z}$, для функции от трех переменных — $x \wedge \overline{y} \wedge z$, $x \wedge \overline{y} \wedge \overline{z}$, $x \wedge y \wedge z$ и другие.

Определение
Дизъюнктивной нормальной формой (ДНФ) называется формула, представленная в виде дизъюнкции элементарных конъюнкций.

Определение
Элементарная конъюнкция $x_1^{\sigma_1} \wedge x_2^{\sigma_2} \wedge \ldots \wedge x_n^{\sigma_n}$ называется ***конституентой единицы (минтермом)*** функции $f(x_1, x_2, \ldots, x_n)$, если $f(\sigma_1, \sigma_2, \ldots, \sigma_n) = 1$, то есть интерпретация, обращающая в единицу данную элементарную конъюнкцию, обращает в единицу и функцию f.

Конституента единицы обладает следующими свойствами:

1. Конституента единицы равна единице только на соответствующей ей интерпретации.

2. Значение конституенты единицы однозначно определяется номером соответствующей интерпретации.

3. Конъюнкция любого числа различных конституент единицы функции равна нулю.

Пример. Элементарная конъюнкция $x \wedge \overline{y}$ является конституентой единицы функции двух переменных $f(x, y)$ на интерпретации

(1, 0), так как $x \wedge \bar{y} = x^1 \wedge y^0$ и $x \wedge \bar{y} = 1$. Элементарная конъюнкция $x \wedge y \wedge z$ является конституентой единицы функции трех переменных $f(x, y, z)$ на интерпретации **(1, 1, 1)**, так как $x \wedge y \wedge z = x^1 \wedge y^1 \wedge z^1$ и $x \wedge y \wedge z = 1$.

Определение
Совершенной дизъюнктивной нормальной формой (СДНФ) булевой функции называется формула, представленная в виде дизъюнкции конституент единицы данной функции.

СДНФ функции является результатом дизъюнктивного разложения функции по всем переменным и соответствует формуле **(4.5)**. Как видно из определения, СДНФ функции содержит только $\wedge, \vee, \bar{}$, следовательно, применив к СДНФ принцип двойственности, можно получить двойственное представление, которое называется ***конъюнктивным разложением***.

Теорема 2. *О конъюнктивном разложении булевой функции $f(x_1, x_2, ..., x_n)$ по k переменным*

Любую булеву функцию $f(x_1, x_2, ..., x_n)$ можно представить в следующей форме:

$$f(x_1, \ldots, x_k, x_{k+1}, ..., x_n) =$$

$$= \bigwedge_{(\sigma_1,\sigma_2,...,\sigma_k)} x_1^{\bar{\sigma}_1} \vee x_2^{\bar{\sigma}_2} \vee ... \vee x_k^{\bar{\sigma}_k} \vee f(\sigma_1, \sigma_2, ..., \sigma_k, x_{k+1}, ..., x_n). \qquad (4.6)$$

Запись $\bigwedge_{(\sigma_1,\sigma_2,...,\sigma_k)}$ означает, что конъюнкция берется по всем возможным наборам значений $(\sigma_1, \sigma_2, ..., \sigma_k)$.

□ *Доказательство.* Докажем теорему, используя принцип двойственности и тождество **(4.3)**. Запишем дизъюнктивное разложение функции $f^*(x_1, x_2, ..., x_n)$, двойственной функции $f(x_1, x_2, ..., x_n)$.

$$f^*(x_1, x_2, ..., x_n) =$$

$$= \bigvee_{(\sigma_1, \sigma_2, ..., \sigma_k)} x_1^{\sigma_1} \wedge x_2^{\sigma_2} \wedge ... \wedge x_k^{\sigma_k} \wedge f^*(\sigma_1, \sigma_2, ..., \sigma_k, x_{k+1}, ..., x_n).$$

Тождество, двойственное к последнему, имеет вид:

$$f^{**}(x_1, x_2, ..., x_n) =$$

$$= \bigwedge_{(\sigma_1, \sigma_2, ..., \sigma_k)} (x_1^{\sigma_1})^* \vee (x_2^{\sigma_2})^* \vee ... \vee (x_k^{\sigma_k})^* \vee f^{**}(\sigma_1, \sigma_2, ..., \sigma_k, x_{k+1}, ..., x_n).$$

В левой части равенства получаем $f(x_1, x_2, ..., x_n)$. В правой части перейдем к двойственным выражениям для $x_i^{\sigma_i}$. Для этого запишем формулу (4.2) и найдем двойственную ей путем замены всех входящих в нее функций на двойственные:

$$(x_i^{\sigma_i})^* = (x_i\sigma_i \vee \overline{x}_i\overline{\sigma}_i)^* = (x_i \vee \sigma_i)(\overline{x}_i \vee \overline{\sigma}_i) =$$

$$= x_i\overline{\sigma}_i \vee \overline{x}_i\sigma_i = x_i(\overline{\sigma}_i) \vee \overline{x}_i(\overline{\overline{\sigma}}_i) = x_i^{\overline{\sigma}_i}.$$

Таким образом, равенство с f^{**} принимает форму (4.6), что и требовалось доказать. ■

Пример. Рассмотрим конъюнктивное разложение функции $f(x, y, z, t) = (xy \vee \overline{z})t$ по переменным x, t.

Решение. По формуле (4.6) имеем:

$$f(x, y, z, t) = (x^{\overline{0}} \vee t^{\overline{0}} \vee f(0, y, z, 0))(x^{\overline{0}} \vee t^{\overline{1}} \vee$$

$$\vee f(0, y, z, 1))(x^{\overline{1}} \vee t^{\overline{0}} \vee f(1, y, z, 0))(x^{\overline{1}} \vee t^{\overline{1}} \vee f(1, y, z,1)) =$$

$$= (x \vee t \vee f(0, y, z, 0))(x \vee \overline{t} \vee f(0, y, z, 1))(\overline{x} \vee t \vee$$

$$\vee f(1, y, z, 0))(\overline{x} \vee \overline{t} \vee f(1, y, z, 1)).$$

Используя формулы

$$f(0, y, z, 0) = (0 \wedge y \vee \overline{z}) \wedge 0 = 0,$$

$$f(0, y, z, 1) = (0 \wedge y \vee \overline{z}) \wedge 1 = \overline{z},$$

$$f(1, y, z, 0) = (1 \wedge y \vee \overline{z}) \wedge 0 = 0,$$

$$f(1, y, z, 1) = (1 \wedge y \vee \overline{z}) \wedge 1 = y \vee \overline{z},$$

которые индуцирует функция f при специальных значениях переменных x, t, получим искомое конъюктивное разложение:

$$f(x, y, z, t) = (x \vee t \vee 0)(x \vee \overline{t} \vee \overline{z})(\overline{x} \vee t \vee 0)(\overline{x} \vee \overline{t} \vee (y \vee \overline{z})) =$$

$$= (x \vee t)(x \vee \overline{t} \vee \overline{z})(\overline{x} \vee t)(\overline{x} \vee \overline{t} \vee y \vee \overline{z}).$$

Следствие 3. *Конъюнктивное разложение булевой функции $f(x_1, x_2, ..., x_n)$ по одной переменной.*

Любую булеву функцию $f(x_1, x_2, ..., x_n)$ можно представить в следующей форме:

$$f(x_1, x_2, ..., x_n) = \bigwedge_{\sigma_i} x_i^{\overline{\sigma}_i} \vee f(x_1, x_2, ... x_{i-1}, \sigma_i, x_{i+1}, ..., x_n). \quad (4.7)$$

Запись $\bigwedge_{\sigma_i}$ означает, что конъюнкция берется по двум возможным значениям σ_i, то есть 0 и 1. Индекс i является фиксированным.

Пример. Получить конъюнктивное разложение функции $f(x, y, z) = xy \vee \overline{z}$ по переменной x.

Решение. Воспользуемся предыдущим следствием:

$$f(x, y, z) = (x^{\overline{0}} \vee f(0, y, z)) \wedge (x^{\overline{1}} \vee f(1, y, z)) =$$

$$= (x \vee f(0, y, z)) \wedge (\overline{x} \vee f(1, y, z)) =$$

$$= (x \vee 0 \cdot y \vee \overline{z})(\overline{x} \vee 1 \cdot y \vee \overline{z}) = (x \vee \overline{z})(\overline{x} \vee y \vee \overline{z}).$$

Следствие 4. *Конъюнктивное разложение булевой функции* $f(x_1, x_2, \ldots, x_n)$ *по всем переменным.*

Любую булеву функцию $f(x_1, x_2, \ldots, x_n) \neq 1$ можно представить в следующей форме:

$$f(x_1, x_2, \ldots, x_n) = \bigwedge_{\substack{(\sigma_1, \sigma_2, \ldots, \sigma_n) \\ f(\sigma_1, \sigma_2, \ldots, \sigma_n) = 0}} x_1^{\overline{\sigma}_1} \vee x_2^{\overline{\sigma}_2} \vee \ldots \vee x_n^{\overline{\sigma}_n}. \qquad (4.8)$$

В правой части конъюнкция берется по всем наборам значений $(\sigma_1, \sigma_2, \ldots, \sigma_k)$, на которых $f(x_1, x_2, \ldots, x_n) = 0$.

Рассмотрим, каким образом формула (4.8) получена из формулы (4.6). При $k = n$

$$f(x_1, x_2, \ldots, x_n) = \bigwedge_{(\sigma_1, \sigma_2, \ldots, \sigma_n)} x_1^{\overline{\sigma}_1} \vee x_2^{\overline{\sigma}_2} \vee \ldots \vee x_n^{\overline{\sigma}_n} \vee f(\sigma_1, \sigma_2, \ldots, \sigma_n).$$

Если $f(\sigma_1, \sigma_2, \ldots, \sigma_n) = 0$, то $x_1^{\overline{\sigma}_1} \vee x_2^{\overline{\sigma}_2} \vee \ldots \vee x_n^{\overline{\sigma}_n} \vee f(\sigma_1, \sigma_2, \ldots, \sigma_n) = x_1^{\overline{\sigma}_1} \vee x_2^{\overline{\sigma}_2} \vee \ldots \vee x_n^{\overline{\sigma}_n}$,

если $f(\sigma_1, \sigma_2, \ldots, \sigma_n) = 1$, то $x_1^{\overline{\sigma}_1} \vee x_2^{\overline{\sigma}_2} \vee \ldots \vee x_n^{\overline{\sigma}_n} \vee f(\sigma_1, \sigma_2, \ldots, \sigma_n) = 1$.

Поэтому конъюнктивное разложение по всем переменным содержит только дизъюнкции, соответствующие наборам $(\sigma_1, \sigma_2, \ldots, \sigma_n)$, для которых $f(\sigma_1, \sigma_2, \ldots, \sigma_n) = 0$.

Пример. Получить конъюнктивное разложение функции $f(x, y, z) = xy \vee \overline{z}$ по всем переменным.

Решение. Определим значение функции на каждой из интерпретаций:

$$f(0, 0, 0) = 0 \wedge 0 \vee \overline{0} = 0 \vee 1 = 1,$$

$$f(0, 0, 1) = 0 \wedge 0 \vee \overline{1} = 0 \vee 0 = 0,$$

$$f(0, 1, 0) = 0 \wedge 1 \vee \overline{0} = 0 \vee 1 = 1,$$

$$f(0, 1, 1) = 0 \wedge 1 \vee \overline{1} = 0 \vee 0 = 0,$$

$$f(1, 0, 0) = 1 \wedge 0 \vee \overline{0} = 0 \vee 1 = 1,$$

$$f(1, 0, 1) = 1 \wedge 0 \vee \overline{1} = 0 \vee 0 = 0,$$
$$f(1, 1, 0) = 1 \wedge 1 \vee \overline{0} = 1 \vee 1 = 1,$$
$$f(1, 1, 1) = 1 \wedge 1 \vee \overline{1} = 1 \vee 0 = 1.$$

По формуле (4.8) получаем

$$f(x, y, z) = (x^1 \vee y^1 \vee z^0)(x^1 \vee y^0 \vee z^0)(x^0 \vee y^1 \vee z^0) =$$
$$= (x \vee y \vee \overline{z})(x \vee \overline{y} \vee \overline{z})(\overline{x} \vee y \vee \overline{z}).$$

Определение

Элементарной дизъюнкцией называется дизъюнкция любого числа булевых переменных, взятых с отрицанием или без него, в которой каждая переменная встречается не более одного раза. Элементарной дизъюнкцией, содержащей ноль переменных, будем считать константу 0.

Пример. Элементарными дизъюнкциями от одной, двух и более переменных являются: y, $\overline{z}$, $\overline{x} \vee y$, $x \vee \overline{z}$, $\overline{x} \vee y \vee z$, $x \vee \overline{y} \vee \overline{z}$, $x \vee y \vee z$ и т.п.

Определение

Конъюнктивной нормальной формой (***КНФ***) называется формула, представленная в виде конъюнкции элементарных дизъюнкций.

Определение

Элементарная дизъюнкция $x_1^{\overline{\sigma}_1} \vee x_2^{\overline{\sigma}_2} \vee \ldots \vee x_n^{\overline{\sigma}_n}$ называется ***константой нуля*** (***макстермом***) функции $f(x_1, x_2, \ldots, x_n)$, если $f(\sigma_1, \sigma_2, \ldots, \sigma_n) = 0$, то есть интерпретация, обращающая в нуль данную элементарную дизъюнкцию, обращает в нуль и функцию f.

Конституента нуля обладает следующими свойствами:

1. Конституента нуля равна нулю только на соответствующей ей интерпретации.
2. Конституента нуля однозначно определяется номером соответствующей ей интерпретации.
3. Дизъюнкция любого числа различных конституент нуля функции равна единице.

Пример. Элементарная дизъюнкция $x \vee \overline{y}$ является конституентой нуля функции двух переменных $f(x, y)$ на интерпре-

тации (0, 1), так как $x \vee \bar{y} = x^1 \vee y^0 = x^{\bar{0}} \vee y^{\bar{1}}$, следовательно, на интерпретации $(x, y) = (0, 1)$ выполнено равенство $x \vee \bar{y} = 0$. Элементарная дизъюнкция $x \vee y \vee z$ является конституентой нуля функции трех переменных $f(x, y, z)$ на интерпретации (0, 0, 0). Действительно, $x \vee y \vee z = x^1 \vee y^1 \vee z^1 = x^{\bar{0}} \vee y^{\bar{0}} \vee z^{\bar{0}}$, следовательно, дизъюнкция $x \vee y \vee z$ равна нулю на интерпретации $(0, 0, 0) = (x, y, z)$.

> **Определение**
> ***Совершенной конъюнктивной нормальной формой (СКНФ)*** функции называется формула, представленная в виде конъюнкции конституент нуля данной функции.

СКНФ функции является результатом конъюнктивного разложения функции по всем переменным и соответствует формуле (4.8). Из анализа формул (4.5) и (4.8), соответствующих СДНФ и СКНФ функции, можно сделать следующие выводы:

1. Для каждой булевой функции $f(x_1, x_2, ..., x_n)$, не являющейся константой нуль, существует представление в виде СДНФ.
2. Для каждой булевой функции $f(x_1, x_2, ..., x_n)$, не являющейся константой единица, существует представление в виде СКНФ.
3. Две различные булевы функции не могут иметь одинаковые СДНФ или СКНФ.
4. Для каждой булевой функции $f(x_1, x_2, ..., x_n)$ существует представление в виде формулы булевой алгебры, содержащей только операции дизъюнкции, конъюнкции и отрицания.

Вопросы

1. Запишите формулы дизъюнктивного разложения булевых функций от n переменных по k переменным ($k < n$) по всем n переменным, по одной переменной.
2. Запишите формулы конъюнктивного разложения булевых функций от n переменных по k переменным ($k < n$) по всем n переменным, по одной переменной.
3. Дайте определения следующих понятий: элементарная конъюнкция, элементарная дизъюнкция, конституента единицы, конституента нуля. Каким образом эти понятия связаны с дизъюнктивным и конъюнктивным разложением булевых функций?

4. Какими свойствами обладают конституенты единицы и конституенты нуля?
5. Сформулируйте определения понятий нормальных и совершенных нормальных форм булевых функций. Объясните их связь с дизъюнктивным и конъюнктивным разложением булевых функций.

Задания

1. Выпишите конституенты единицы булевой функции $f(x_1, x_2, x_3, x_4)$ из следующего списка элементарных конъюнкций:
   а) $x_1\bar{x}_2\bar{x}_3$; б) $x_1\bar{x}_2\bar{x}_1$; в) $x_4x_2\bar{x}_3x_1$; г) x_1.
2. Выпишите конституенты нуля булевой функции $f(x_1, x_2, x_3, x_4)$ из следующего списка элементарных дизъюнкций:
   а) $x_1 \vee \bar{x}_3 \vee x_5$; б) $x_4 \vee x_2 \vee \bar{x}_3$; в) $\bar{x}_1 \vee \bar{x}_2 \vee \bar{x}_3 \vee \bar{x}_4$; г) $\bar{x}_1$.
3. Найти дизъюнктивное разложение следующих функций по переменным x, z:
   а) $(yx \vee x\bar{z})(x \vee \bar{y}z(z \vee \bar{x}y))$;
   б) $((x \vee (z \vee yz)))(z \vee \bar{x}\,\bar{z} \vee y)$;
   в) $((x \vee \bar{y})(\bar{y} \vee \bar{z}\,)(\bar{z} \vee x)) \vee ((\bar{y} \vee z))$;
   г) $(x \vee \bar{z})(\bar{x}t \vee yt \vee \bar{x}t \vee y\bar{t})(x \vee z)$;
   д) $(xy \vee x\bar{y}) \vee ((\bar{x} \vee y)(z \vee \bar{t}\,)(\bar{x} \vee \bar{y})(t \vee z))$.
4. Найти конъюнктивное разложение следующих функций по переменным x, z:
   а) $(xz \vee y)(x\bar{y} \vee \bar{x}\,\bar{y} \vee \bar{z})(x \vee \bar{y})$;
   б) $((y \vee z)(t \vee y\bar{z})) \vee \bar{t}\,\bar{x} \vee ((z \vee y)(\bar{t} \vee \bar{z}))$;
   в) $yt \vee ((z \vee \bar{t})(x \vee z)(\bar{t} \vee \bar{z})(x \vee \bar{z})) \vee \bar{y}t$;
   г) $(\bar{z} \vee t)(t \vee x) \vee (\bar{z} \vee \bar{x})(\bar{z} \vee \bar{t})(\bar{t} \vee z)$;
   д) $x\bar{t} \vee ((\bar{z}\,\bar{y}) \vee t)(z \vee y)) \vee ((\bar{t} \vee \bar{z})(z \vee y))$;
   е) $(((t \vee \bar{t}\,z)\bar{t}) \vee \bar{y})((y \vee t)(y \vee x))$;
   ж)$((z \vee \bar{x})(\bar{x} \vee \bar{t})(x \vee z)(\bar{y} \vee x)) \vee y\bar{t} \vee yt$;
   з) $(t \vee \bar{x}t \vee x)(y \vee (t \vee tz))\bar{z}\bar{x}t$;
   и) $\bar{z}\,\bar{y} \vee tz \vee \bar{y}z \vee t\bar{z} \vee y\bar{t}$.

4.6. Нормальные формы представления булевых функций

Алгоритмы перехода от таблиц истинности булевых функций к СДНФ/СКНФ и наоборот, алгоритмы перехода от произвольной формулы к СКНФ и СДНФ

Для каждой интерпретации функции имеются единственные соответствующие ей конституента единицы и конституента нуля. Поэтому различных конституент единицы и нуля для функции n переменных $f(x_1, x_2, \ldots, x_n)$ имеется столько же, сколько

и интерпретаций этой функции — 2^n. Подсчитаем, сколько существует различных СДНФ и СКНФ для булевых функций n переменных. СДНФ функции n переменных можно представить в виде:

$$f(x_1, x_2, ..., x_n) = f_0 \wedge M_0 \vee f_1 \wedge M_1 \vee ... \vee f_{2^n-1} \wedge M_{2^n-1},$$

где f_i — значение функции $f(x_1, x_2, ..., x_n)$ на i-ой интерпретации;

M_i — конституента единицы, соответствующая i-ой интерпретации.

Поскольку $M_0, M_1, ..., M_{2^n-1}$ одинаковы для всех функций n переменных, то вид СДНФ зависит от набора 2^n булевых констант $(f_0, f_1, ..., f_{2^n-1})$. Из этого следует, что количество различных СДНФ равно количеству упорядоченных наборов 2^n булевых констант и равно 2^{2^n} (п. 4.1). Количество различных булевых функций от n переменных также составлет 2^{2^n} (п. 4.1). Следовательно, для каждой булевой функции существует единственная СДНФ. Аналогично доказывается, что различных СКНФ функции от n переменных существует 2^{2^n} и, следовательно, каждой булевой функции соответствует единственная СКНФ.

Пример. Запишем конституенты единицы и нуля, соответствующие интерпретациям функций трех переменных (таблица 4.19).

Таблица 4.19. Конституенты нуля и единицы функций трех переменных

Номер интерпретации	Интерпретация			Конституента единицы	Конституента нуля
	x_1	x_2	x_3		
0	0	0	0	$\bar{x}\bar{y}\bar{z}$	$x \vee y \vee z$
1	0	0	1	$\bar{x}\bar{y}z$	$x \vee y \vee \bar{z}$
3	0	1	1	$\bar{x}yz$	$x \vee \bar{y} \vee \bar{z}$
4	1	0	0	$x\bar{y}\bar{z}$	$\bar{x} \vee y \vee z$
5	1	0	1	$x\bar{y}z$	$\bar{x} \vee y \vee \bar{z}$
6	1	1	0	$xy\bar{z}$	$\bar{x} \vee \bar{y} \vee z$
7	1	1	1	xyz	$\bar{x} \vee \bar{y} \vee \bar{z}$

Алгоритм перехода от таблицы истинности булевой функции к СДНФ

1. Выделить все интерпретации $(\sigma_1, \sigma_2, ..., \sigma_n)$, на которых значение функции равно единице.

2. Записать конституенты единицы вида $x_1^{\sigma_1} \wedge x_2^{\sigma_2} \wedge \ldots \wedge x_n^{\sigma_n}$, соответствующие отмеченным интерпретациям.

3. Получить СДНФ функции посредством соединения операцией дизъюнкции записанных конституент единицы.

Алгоритм перехода от таблицы истинности булевой функции к СКНФ

1. Выделить все интерпретации (σ_1, σ_2, ..., σ_n), на которых значение функции равно нулю.

2. Записать конституенты нуля вида $x_1^{\bar{\sigma}_1} \vee x_2^{\bar{\sigma}_2} \vee \ldots \vee x_n^{\bar{\sigma}_n}$, соответствующие выделенным интерпретациям.

3. Записав конъюнкцию конституент нуля, получить СКНФ функции.

Пример. Получить СДНФ для функций $f_{13}(x, y)$ и $f_8(x, y)$ (таблицы 4.20, 4.21).

Решение. Построим СДНФ заданных функций, используя вышеописанный алгоритм. На первом шаге построим таблицу истинности булевой функции $f_{13}(x, y)$ (таблица 4.20).

Таблица 4.20. Функция $f_{13}(x, y)$

x	y	$f_{13}(x, y)$
0	0	1
0	1	1
1	0	0
1	1	1

Для получения СДНФ отметим интерпретации, на которых $f_{13}(x, y) = 1$, запишем соответствующие конституенты единицы, объединив их знаком дизъюнкции (см. формулу (4.5)):

$$f_{13}(x, y) = x^0 y^0 \vee x^0 y^1 \vee x^1 y^1 = \bar{x}\,\bar{y} \vee \bar{x}y \vee xy.$$

Перейдем к СДНФ функции $f_8(x, y)$ с таблицей истинности (таблица 4.21).

Таблица 4.21. Функция $f_8(x, y)$

x	y	$f_8(x, y)$
0	0	1
0	1	0
1	0	0
1	1	0

Данная функция равна единице только на одной интерпретации, которая соответствует конституенте единицы, следовательно:

$$f_8(x, y) = x^0 y^0 = \overline{x}\,\overline{y}.$$

Пример. Получить СКНФ для функций $f_{13}(x, y)$ и $f_8(x, y)$.

Решение. Функция f_{13} равна нулю на единственном наборе (1, 0) (см. таблицу 4.20). Записав соответствующую этому набору конституенту нуля, получим СКНФ функции f_{13} согласно алгоритму:

$$f_{13}(x, y) = x^{\overline{1}} \vee y^{\overline{0}} = x^0 \vee y^1 = \overline{x} \vee y.$$

Для получения СКНФ функции $f_8(x, y)$ выпишем все три конституенты нуля (см. таблицу 4.21), объединив их знаком конъюнкции:

$$f_8(x, y) = (x^{\overline{0}} \vee y^{\overline{1}})\,(x^{\overline{1}} \vee y^{\overline{0}})\,(x^{\overline{1}} \vee y^{\overline{1}}) = (x \vee \overline{y})\,(\overline{x} \vee y)\,(\overline{x} \vee \overline{y}).$$

Количество интерпретаций, на которых функция равна единице, равно количеству конституент единицы в СДНФ этой функции. Аналогично количество интерпретаций, на которых функция равна нулю, равно количеству конституент нуля в СКНФ этой функции. Поэтому для функций, равных нулю на более чем половине интерпретаций, выгоднее строить СДНФ, чем СКНФ. Действительно, в данном случае СДНФ будет содержать меньше символов переменных и знаков операций, чем СКНФ. И наоборот, для функций, равных единице на большинстве интерпретаций, выгоднее строить СКНФ.

Получение таблицы истинности функции, заданной СДНФ или СКНФ, представляет собой процедуру, обратную рассмотренной выше.

Пример. Для функции $f(x, y, z) = x\,y\,\overline{z} \vee \overline{x}\,y\,z \vee \overline{x}\,\overline{y}\,\overline{z}$, заданной СДНФ, построить ее таблицу истинности.

Решение. Данная функция содержит три конституенты единицы — $x \wedge y \wedge \overline{z}$, $\overline{x} \wedge y \wedge z$, $\overline{x} \wedge \overline{y} \wedge \overline{z}$, которым соответствуют интерпретации (1, 1, 0), (0, 1, 1), (0, 0, 0). На данных интерпретациях функция равна единице, на остальных — нулю. Запишем указанные значения функции в столбец $f(x, y, z)$ (таблица 4.22). Столбцы x, y, z, $f(x, y, z)$ образуют таблицу истинности функции f.

Пример. Для функции $g(x, y, z) = (x \vee y \vee \overline{z})(\overline{x} \vee y \vee z)(\overline{x} \vee \overline{y} \vee \overline{z})$, заданной СКНФ, построить ее таблицу истинности.

Решение. Воспользуемся вышеописанным алгритмом. Конституентам нуля $x \vee y \vee \overline{z}$, $\overline{x} \vee y \vee z$, $\overline{x} \vee \overline{y} \vee \overline{z}$ данной функции

соответствуют интерпретации (0, 0, 1), (1, 0, 0), (1, 1, 1). На перечисленных интерпретациях функция равна нулю, на остальных — единице. Записываем данные значения функции в столбец $g(x, y, z)$ (таблица 4.22). Столбцы x, y, z, $g(x, y, z)$ образуют таблицу истинности функции g.

Таблица 4.22. Таблица истинности $f(x, y, z)$ и $g(x, y, z)$

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

Алгоритм перехода от произвольной формулы алгебры логики к СДНФ

1. Исключить константы, используя законы действий с константами.

2. Опустить знаки отрицания непосредственно на переменные, используя законы де Моргана.

3. Используя дистрибутивный закон, раскрыть скобки. К полученным элементарным конъюнкциям применить законы идемпотентности и противоречия, упростить их и привести подобные. Результатом выполнения указанных действий является получение ДНФ булевой функции.

4. Построить конституенты единицы функции введением в каждую элементарную конъюнкцию недостающих переменных, используя закон исключенного третьего.

5. С помощью дистрибутивного закона раскрыть скобки и привести подобные, используя закон идемпотентности. Полученная формула соответствует СДНФ функции.

Алгоритм перехода от произвольной формулы алгебры логики к СКНФ

1. Исключить константы, используя законы действий с константами.

2. Опустить знаки отрицания непосредственно на переменные, используя законы де Моргана.

3. Посредством использования дистрибутивного закона, привести функцию к виду конъюнкции элементарных дизъюнкций. К полученным элементарным дизъюнкциям применить законы идемпотентности и исключенного третьего, упростить их и привести подобные. Результатом выполнения указанных действий является получение КНФ булевой функции.

4. Построить конституенты нуля функции введением в каждую элементарную дизъюнкцию недостающих переменных, используя закон противоречия.

5. С помощью дистрибутивного закона привести функцию к виду конъюнкции конституент нуля и упростить формулу, используя закон идемпотентности. Полученная формула является СКНФ функции.

Пример. Построить СДНФ функции

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

Решение. Воспользуемся предпоследним алгоритмом. Опускаем отрицания на переменные, используя законы де Моргана:

$$xy \vee \overline{(x(\bar{y} \vee z) \vee yz)} = xy \vee \overline{(x(\bar{y} \vee z))}\,\overline{(yz)} =$$
$$= xy \vee (\bar{x} \vee \overline{(\bar{y} \vee z)})(\bar{y} \vee \bar{z}) = xy \vee (\bar{x} \vee (y\bar{z}))(\bar{y} \vee \bar{z}).$$

Построим ДНФ, используя дистрибутивный закон, законы идемпотентности и противоречия:

$$xy \vee (\bar{x} \vee (y\bar{z}))(\bar{y} \vee \bar{z}) = xy \vee (\bar{x}\,\bar{y} \vee y\bar{z}\,\bar{y} \vee \bar{x}\,\bar{z} \vee y\bar{z}\,\bar{z}) =$$
$$= xy \vee \bar{x}\,\bar{y} \vee 0 \vee \bar{x}\,\bar{z} \vee y\bar{z} = xy \vee \bar{x}\,\bar{y} \vee \bar{x}\,\bar{z} \vee y\bar{z}.$$

Данная функция зависит от трех переменных, поэтому в элементарные конъюнкции необходимо ввести недостающие переменные, используя закон исключенного третьего:

$$xy \vee \bar{x}\,\bar{y} \vee \bar{x}\,\bar{z} \vee y\bar{z} = xy(z \vee \bar{z}) \vee \bar{x}\,\bar{y}(z \vee \bar{z}) \vee \bar{x}(y \vee \bar{y})\bar{z} \vee (x \vee \bar{x})y\bar{z}.$$

Используя дистрибутивный закон, раскроем скобки и приведем подобные для получения СДНФ:

$$xyz \vee xy\bar{z} \vee \bar{x}\,\bar{y}z \vee \bar{x}\,\bar{y}\,\bar{z} \vee \bar{x}y\bar{z} \vee \bar{x}\,\bar{y}\,\bar{z} \vee xy\bar{z} \vee \bar{x}y\bar{z} =$$
$$= xyz \vee xy\bar{z} \vee \bar{x}\,\bar{y}z \vee \bar{x}\,\bar{y}\,\bar{z} \vee \bar{x}y\bar{z}.$$

Получена СДНФ заданной функции:

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

Пример. Построить СКНФ функции

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

Решение. Воспользуемся последним алгоритмом. Как в предыдущем примере, опустим отрицания непосредственно на переменные и, используя законы де Моргана, получим:

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

Построим КНФ, используя дистрибутивный закон, законы идемпотентности и исключенного третьего:

$$xy \vee (\bar{x} \vee (y\bar{z}))(\bar{y} \vee \bar{z}) = xy \vee (\bar{x} \vee y)(\bar{x} \vee \bar{z})(\bar{y} \vee \bar{z}) =$$
$$= (x \vee (\bar{x} \vee y)(\bar{x} \vee \bar{z})(\bar{y} \vee \bar{z}))(y \vee (\bar{x} \vee y)(\bar{x} \vee \bar{z})(\bar{y} \vee \bar{z})) =$$
$$= (x \vee \bar{x} \vee y)(x \vee \bar{x} \vee \bar{z})(x \vee \bar{y} \vee \bar{z})(y \vee \bar{x} \vee y)(y \vee \bar{x} \vee \bar{z})(y \vee \bar{y} \vee \bar{z}) =$$
$$= 1 \cdot 1 \cdot (x \vee \bar{y} \vee \bar{z})(y \vee \bar{x})(y \vee \bar{x} \vee \bar{z}) \cdot 1 = (x \vee \bar{y} \vee \bar{z})(y \vee \bar{x})(y \vee \bar{x} \vee \bar{z}).$$

Данная функция зависит от трех переменных, следовательно, в элементарную дизъюнкцию $(y \vee \bar{x})$ необходимо ввести недостающую переменную z, используя закон противоречия. После чего, используя дистрибутивный закон, следует привести функцию к виду конъюнкции конституент нуля:

$$(x \vee \bar{y} \vee \bar{z})(y \vee \bar{x})(y \vee \bar{x} \vee \bar{z}) = (x \vee \bar{y} \vee \bar{z})(\bar{x} \vee y \vee z\bar{z})(\bar{x} \vee y \vee \bar{z}) =$$
$$= (x \vee \bar{y} \vee \bar{z})(\bar{x} \vee y \vee z)(\bar{x} \vee y \vee \bar{z})(\bar{x} \vee y \vee \bar{z}).$$

После приведения подобных с помощью закона идемпотентности получаем СКНФ:

$$f(x, y, z) = (x \vee \bar{y} \vee \bar{z})(\bar{x} \vee y \vee z)(\bar{x} \vee y \vee \bar{z}).$$

Вопросы

1. Сколько существует конституент единицы и нуля для функции n переменных?
2. Каким образом для заданной интерпретации булевой функции строятся соответствующие им конституенты единицы, нуля?
3. Опишите алгоритмы перехода от таблицы истинности булевой функции к СДНФ и СКНФ.
4. Объясните алгоритм перехода от СДНФ булевой функции к таблице истинности.
5. Опишите алгоритм перехода от СКНФ булевой функции к таблице истинности.
6. Дайте сравнительную характеристику алгоритмов перехода от произвольной формулы булевой функции к СДНФ и СКНФ.

Задания

1. Найти ДНФ функции, заданной формулой $(x(\overline{x} \to y)) \to y$.
2. Получить КНФ формулы $(\overline{x \vee y})(x \to y)$.
3. Построить таблицы истинности для функций, заданных СКНФ:
   а) $f_1(x, y, z) = (\overline{x} \vee y \vee \overline{z})(x \vee \overline{y} \vee z)(x \vee y \vee \overline{z})$;
   б) $f_2(x, y, z) = (\overline{x} \vee y \vee z)(x \vee \overline{y} \vee \overline{z})(\overline{x} \vee \overline{y} \vee z)$;
   в) $f_3(x, y, z) = (\overline{x} \vee \overline{y} \vee \overline{z})(x \vee y \vee z)(\overline{x} \vee y \vee \overline{z})$.
4. Построить таблицы истинности для функций, заданных СДНФ:
   а) $f(x, y, z) = x\,\overline{y}\,\overline{z} \vee \overline{x}\,y\,\overline{z} \vee x\,\overline{y}\,z$;
   б) $f(x, y, z) = x\,y\,z \vee x\,\overline{y}\,\overline{z} \vee \overline{x}\,\overline{y}\,z$;
   в) $f(x, y, z) = x\,y\,\overline{z} \vee \overline{x}\,y\,z \vee x\,\overline{y}\,z$;
   г) $f(x, y, z) = \overline{x}\,y\,z \vee x\,\overline{y}\,\overline{z} \vee \overline{x}\,\overline{y}\,z$.
5. Найдите СДНФ следующих функций:
   а) $f_1(x, y, z) = (1, 1, 0, 1, 0, 0, 0, 0)$, здесь функция f_1 задана столбцом значений из таблицы истинности;
   б) $f_2(x, y, z) = x \wedge y \wedge z$;
   в) $f_3(x, y, z) = x \oplus y \oplus z$;
   г) $f_4(x, y, z) = x \vee yz$;
   д) $f_5(x, y, z) = (x \wedge y) \to z$.
6. Получите СКНФ следующих функций:
   а) $f_1(x, y, z) = (1, 1, 0, 1, 0, 0, 0, 0)$, здесь функция f_1 задана столбцом значений из таблицы истинности;
   б) $f_2(x, y, z) = x \vee y \vee z$;
   в) $f_3(x, y, z) = x \oplus y \oplus z$;
   г) $f_4(x, y, z) = x \vee \overline{y}z \vee \overline{z}$;
   д) $f_{201}(x, y, z)$.
7. Докажите справедливость следующих тождеств, используя законы булевой алгебры:
   а) $((x \mid y) \mid (x \sim y)) \mid ((z \oplus t) \to (t \leftarrow z)) = ((y \to z) \to (x \leftarrow z)) \downarrow \downarrow ((x \mid t) \mid (t \to y))$;
   б) $((x \wedge \overline{z}) \downarrow (y \leftarrow z)) \wedge ((x \mid t) \leftarrow (y \wedge t)) = ((x \mid y) \mid (x \oplus y)) \to \to ((z \oplus t) \wedge (t \to z))$;
   в) $((x \downarrow y) \vee (x \oplus y)) \leftarrow ((z \leftarrow t) \downarrow (z \sim t)) = ((z \to x) \wedge (z \to y)) \to \to ((x \downarrow t) \vee (y \downarrow t))$;
   г) $((x \sim y) \leftarrow (x \downarrow y)) \downarrow ((z \sim t) \downarrow (z \leftarrow t)) = ((z \leftarrow x) \downarrow \downarrow (z \leftarrow y)) \mid ((x \downarrow t) \downarrow (y \downarrow t))$;
   д) $((x \wedge y) \vee (x \oplus y)) \leftarrow ((t \leftarrow z) \downarrow (t \sim z)) = ((x \to z) \wedge (y \to z)) \to \to ((x \mid t) \mid (y \mid t))$;
   е) $((x \vee y) \leftarrow (x \oplus y)) \vee ((z \leftarrow t) \downarrow (z \sim t)) = ((z \leftarrow x) \downarrow \downarrow (z \leftarrow y)) \wedge ((x \vee t) \leftarrow (y \downarrow t))$;
   ж) $((t \to y) \to (\overline{z} \leftarrow y)) \downarrow ((z \vee x) \mid (t \to x)) = ((\overline{z} \mid t) \mid (z \oplus t)) \mid ((x \sim y) \to \to (\overline{x} \leftarrow y))$.

4.7. Алгебра Жегалкина. Линейные функции

Структура и тождества алгебры Жегалкина, представление дизъюнкции и отрицания полиномом Жегалкина, линейность булевых функций

Определение

Алгебра $(B, \wedge, \oplus, 0, 1)$, образованная множеством $B = \{0, 1\}$ вместе с операциями $\wedge$ (конъюнкции), $\oplus$ (XOR — от eXclusive OR, сумма по модулю 2) и константами 0, 1, называется ***алгеброй Жегалкина***.

Пример. Формула $(x \oplus y \oplus z) \wedge (x \oplus z \oplus 1) \oplus x \wedge y \oplus 1$, где x, y, z — булевы переменные, является примером формулы алгебры Жегалкина, т.к. она содержит операции конъюнкции и XOR.

В алгебре Жегалкина операция конъюнкции полностью идентична умножению, а операция XOR представляет сложение по модулю для конечных множеств.

4.7.1. Тождества алгебры Жегалкина

Напомним свойства операции конъюнкции (см. п. 4.4):

1) $x \wedge (y \wedge z) = (x \wedge y) \wedge z$ — ассоциативность.

2) $x \wedge y = y \wedge x$ — коммутативность.

3) $x \wedge x = x$ — идемпотентность.

4) $x \wedge 0 = 0$, $x \wedge 1 = x$ — действия с константами.

Свойства операции XOR (сложение по модулю 2):

5) $x \oplus (y \oplus z) = (x \oplus y) \oplus z$ — ассоциативность операции XOR.

Для доказательства строим таблицу истинности (таблица 4.23).

Таблица 4.23. Доказательство ассоциативности операции XOR

x	y	z	$y \oplus z$	$x \oplus (y \oplus z)$	$x \oplus y$	$(x \oplus y) \oplus z$
0	0	0	0	0	0	0
0	0	1	1	1	0	1
0	1	0	1	1	1	1
0	1	1	0	0	1	0
1	0	0	0	1	1	1
1	0	1	1	0	1	0
1	1	0	1	0	0	0
1	1	1	0	1	0	1

Левая и правая части тождества имеют одинаковые таблицы истинности, таким образом, тождество 5 доказано.

6) $x \oplus y = y \oplus x$ — коммутативность операции XOR следует из таблицы истинности (таблица 4.24).

Таблица 4.24. Доказательство тождества 6

x	y	$x \oplus y$	$y \oplus x$
0	0	0	0
0	1	1	1
1	0	1	1
1	1	0	0

7) $x \oplus x = 0$ — закон приведения подобных слагаемых.

Как видно из таблицы истинности для операции XOR (табл. 4.25), если операнды одинаковы, то функция равна нулю, а если операнды различны, то значение функции — единица.

8) $x \oplus 0 = x$ — операция с константой 0 (таблица 4.25).

Таблица 4.25. Доказательство тождеств 7 и 8

x	$x \oplus x$	$x \oplus 0$
0	0	0
1	0	1

9) $x(y \oplus z) = xy \oplus xz$ — дистрибутивность $\wedge$ относительно $\oplus$ следует из таблицы истинности (таблица 4.26).

Таблица 4.26. Дистрибутивности $\wedge$ относительно $\oplus$

x	y	z	$y \oplus z$	$x(y \oplus z)$	xy	xz	$xy \oplus xz$
0	0	0	0	0	0	0	0
0	0	1	1	0	0	0	0
0	1	0	1	0	0	0	0
0	1	1	0	0	0	0	0
1	0	0	0	0	0	0	0
1	0	1	1	1	0	1	1
1	1	0	1	1	1	0	1
1	1	1	0	0	1	1	0

Операция XOR (сумма по модулю 2) играет важную роль в программировании, при обработке и кодировании информации. Она имеет важное свойство — наличие обратного элемента x' для каждого $x \in \{0, 1\}$ (операции дизъюнкции и конъюнкции таким свойством не обладают). В данной алгебре каждый элемент является

обратным к самому себе: $x' = x$. Это позволяет решать уравнения путем прибавления к обеим частям одинаковых элементов. Например, уравнение вида $x \oplus a = b$ решается так:

$x \oplus a \oplus a = b \oplus a$;

$x \oplus 0 = b \oplus a$;

$x = b \oplus a$.

Возможность решения подобных уравнений, отсутствующая в булевой алгебре, обусловила широкое применение операции XOR, в частности, при кодировании информации.

Алгебра Жегалкина является кольцом с единицей 1 (см. п. 3.4). Алгебра Жегалкина не является решеткой, поскольку в ней не выполняются все необходимые для этого свойства, например, идемпотентность обеих операций (п. 3.5).

Любая булева функция может быть представлена через операции конъюнкции, дизъюнкции, отрицания (п. 4.5). Для доказательства представимости любой булевой функции формулой алгебры Жегалкина достаточно выразить дизъюнкцию и отрицание через конъюнкцию и XOR — операции алгебры Жегалкина.

Представление отрицания в алгебре Жегалкина формулой $\overline{x} = x \oplus 0$ следует из таблицы истинности (таблица 4.27):

Таблица 4.27. Операция отрицания

x	$\overline{x}$	$x \oplus 1$
0	1	1
1	0	0

Представление дизъюнкции в алгебре Жегалкина реализуется формулой $x \vee y = xy \oplus x \oplus y$.

Докажем данную формулу аналитически:

$$x \vee y = \overline{\overline{x \vee y}} = \overline{\overline{x} \wedge \overline{y}} = \overline{(x \oplus y)(y \oplus 1)} = (x \oplus 1)(y \oplus 1) \oplus 1 =$$
$$= xy \oplus y \oplus x \oplus 1 \oplus 1 = xy \oplus y \oplus x.$$

Таким образом, любая логическая функция может быть представлена формулой в алгебре Жегалкина.

4.7.2. Полином Жегалкина

Среди всех эквивалентных представлений функции в алгебре Жегалкина выделяется особый вид формул, называемый ***полином Жегалкина***.

Определение

Полиномом Жегалкина называется конечная сумма по модулю 2 попарно различных элементарных конъюнкций над множеством переменных $\{x_1, x_2, ..., x_n\}$. Количество переменных, входящих в элементарную конъюнкцию, называется ***рангом элементарной конъюнкции***.

Количество попарно различных элементарных конъюнкций в полиноме называется ***длиной полинома***.

Представление в виде полинома существует и единственно для каждой булевой функции.

Для построения полинома Жегалкина функции, заданной некоторой формулой алгебры Жегалкина, необходимо раскрыть все скобки в данной формуле по закону дистрибутивности и произвести все возможные упрощения с использованием законов действия с константами, идемпотентности и приведения подобных слагаемых.

Пример. Представить полиномами Жегалкина логические функции импликацию ($\rightarrow$) и эквивалентность ($\sim$).

Решение. Вначале запишем СДНФ данных функций, затем выразим операции дизъюнкции и отрицания через операции конъюнкции и XOR. Получив формулу алгебры Жегалкина, пользуясь описанным выше правилом, получим полином Жегалкина для каждой из данных функций:

$$x \rightarrow y = \bar{x} \vee y = (x \oplus 1) \vee y = (x \oplus 1)y \oplus (x \oplus 1) \oplus y =$$
$$= xy \oplus y \oplus x \oplus 1 \oplus y = xy \oplus x \oplus 1;$$

$$x \sim y = x\,y \vee \bar{x}\,\bar{y} = x\,y\,\bar{x}\,\bar{y} \oplus x\,y \oplus \bar{x}\,\bar{y} = x\,y \oplus \bar{x}\,\bar{y} =$$
$$= xy \oplus (x \oplus 1)(y \oplus 1) = xy \oplus xy \oplus x \oplus y \oplus 1 = x \oplus y \oplus 1.$$

По виду полинома Жегалкина определяется такое важное свойство булевых функций, как линейность.

Определение

Булева функция называется ***линейной***, если ее полином Жегалкина не содержит конъюнкций переменных.

Являются ли линейными операции булевой алгебры? Отрицание — линейная функция, поскольку ее полином Жегалкина $x \oplus 1$ не содержит конъюнкций переменных. Дизъюнкция — нелинейная функция, поскольку ее полином Жегалкина $x \oplus y \oplus xy$ содержит конъюнкцию переменных x и y.

Пример. Определить, линейны ли функции импликации ($\rightarrow$) и эквивалентности (~).

Решение. Проанализируем структуру формул, выведенных в предыдущем примере. Из него видно, что импликация ($\rightarrow$) является нелинейной функцией, а эквивалентность (~) — функция линейная.

Пример. Исследовать на линейность функцию $f(x, y, z) = (x \vee y) \rightarrow \bar{z}$.

Решение. Построим полином Жегалкина функции $f(x, y, z)$, используя следующие тождества: $x \rightarrow y = \bar{x} \vee y$, $x \vee y = xy \oplus x \oplus y$, $\bar{x} = x \oplus 1$:

$$f(x, y, z) = (x \vee y) \rightarrow \bar{z} = \overline{(x \vee y)} \vee \bar{z} = \overline{(x \vee y)}\bar{z} \oplus \overline{(x \vee y)} \oplus \bar{z} =$$
$$= (xy \oplus x \oplus y \oplus 1)(z \oplus 1) \oplus (xy \oplus x \oplus y \oplus 1) \oplus z \oplus 1 =$$
$$= xyz \oplus xz \oplus yz \oplus 1 \wedge z \oplus xy \wedge 1 \oplus x \wedge 1 \oplus y \wedge 1 \oplus 1 \wedge 1 \oplus$$
$$\oplus xy \oplus x \oplus y \oplus 1 \oplus z \oplus 1 = xyz \oplus xz \oplus yz \oplus z \oplus xy \oplus x \oplus$$
$$\oplus y \oplus 1 \oplus xy \oplus x \oplus y \oplus 1 \oplus z \oplus 1 = xyz \oplus xz \oplus yz \oplus 1.$$

Функция $f(x, y, z) = (x \vee y) \rightarrow \bar{z}$ не является линейной, поскольку ее полином Жегалкина содержит конъюнкции переменных.

Теорема 1

Для любой булевой функции существует единственный полином Жегалкина.

□ *Доказательство.* Докажем существование полинома. Для любой функции существует представление в виде формул булевой алгебры (п. 4.5). От формул булевой алгебры всегда можно перейти к формулам алгебры Жегалкина, а затем к полиному Жегалкина, используя тождества представления операций дизъюнкции и отрицания в алгебре Жегалкина, а также правило построения полинома. Таким образом, для любой булевой функции существует полином Жегалкина.

Подсчитаем количество различных полиномов Жегалкина от n переменных. Вначале найдем количество различных элементарных конъюнкций без отрицаний от n переменных. В общем случае такую конъюнкцию можно записать в следующем виде:

$$(x_1 \vee \sigma_1) \wedge (x_2 \vee \sigma_2) \wedge \ldots \wedge (x_n \vee \sigma_n),$$

где $\sigma_1, \sigma_2, \ldots, \sigma_n$ — булевы константы, причем $\sigma_i = 0$, если x_i присутствует в конъюнкции и $\sigma_i = 1$, если x_i не присутствует в конъюнкции. Набор $(\sigma_1, \sigma_2, \ldots, \sigma_n)$ полностью определяет элементарную конъюнкцию, следовательно, количество таких наборов $(\sigma_1, \sigma_2, \ldots, \sigma_n)$ равно количеству элементарных конъюнкций без отрицаний от n

переменных. Количество двоичных наборов (кодов) длины n равно 2^n, что было доказано в п. 4.1. Поэтому количество элементарных конъюнкций от n переменных без отрицаний равно 2^n. Полином Жегалкина, по определению, является суммой по модулю 2 таких элементарных конъюнкций. Перенумеруем все возможные элементарные конъюнкции без отрицаний от n переменных и обозначим их K_1, K_2, ..., K_{2^n}. Запишем полином Жегалкина в виде:

$$K_1 \wedge \sigma_1 \oplus K_2 \wedge \sigma_2 \oplus \ldots \oplus K_{2^n} \wedge \sigma_{2^n},$$

где σ_1, σ_2,..., σ_{2^n} — булевы константы, причем $\sigma_i = 1$, если конъюнкция K_i присутствует в полиноме и $\sigma_i = 0$, если конъюнкция K_i не присутствует в полиноме. Набор (σ_1, σ_2, ..., σ_{2^n}) полностью определяет вид полинома, значит количество таких наборов длины 2^n равно количеству полиномов Жегалкина от n переменных, то есть равно 2^{2^n} (п. 4.1). Количество различных булевых функций от n переменных также равно 2^{2^n} (п. 4.1). Так как одному полиному не могут соответствовать две различные функции, каждой булевой функции соответствует единственный полином Жегалкина. Теорема доказана. ■

На данной теореме основан метод неопределенных коэффициентов, который можно использовать для нахождения полинома Жегалкина заданной функции. Рассмотрим этот метод на примере.

Пример. Построить полином Жегалкина для функции $f_{13}(x, y)$ — импликации, используя метод неопределенных коэффициентов.

Решение. Запишем полином для данной функции в виде суммы по модулю 2 всех возможных элементарных конъюнкций для x, y с неопределенными коэффициентами:

$$f_{13}(x, y) = x \to y = a_1xy \oplus a_2x \oplus a_3y \oplus a_4,$$

где коэффициенты a_1, a_2, a_3, a_4 принимают значения из множества $\{0, 1\}$ и определяют присутствие или отсутствие элементарной конъюнкции в полиноме. Ищем последовательно значения коэффициентов, подставляя значения переменных и функции на различных интерпретациях:

$$f_{13}(0, 0) = 0 \to 0 = 1,$$

$$1 = a_1 \wedge 0 \wedge 0 \oplus a_2 \wedge 0 \oplus a_3 \wedge 0 \oplus a_4 = a_4.$$

Итак, $a_4 = 1$.

Далее

$$f_{13}(0, 1) = 0 \to 1 = 1,$$

$$1 = a_1 \wedge 0 \wedge 1 \oplus a_2 \wedge 0 \oplus a_3 \wedge 1 \oplus 1 = a_3 \oplus 1,$$

откуда следует, что $a_3 = 0$.

На следующей интерпретации

$$f_{13}(1, 0) = 1 \to 0 = 0,$$
$$0 = a_1 \wedge 1 \wedge 0 \oplus a_2 \wedge 1 \oplus a_3 \wedge 0 \oplus 1 = a_2 \oplus 1,$$

откуда следует, что $a_2 = 1$.

Наконец,

$$f_{13}(1, 1) = 1 \to 1 = 1,$$
$$1 = a_1 \wedge 1 \wedge 1 \oplus 1 \wedge 1 \oplus 0 \wedge 1 \oplus 1 = a_1 \oplus 1 \oplus 1 = a_1.$$

Подставив полученные значения коэффициентов a_1, a_2, a_3, a_4, получаем полином Жегалкина для функции f_{13}:

$$x \to y = a_1xy \oplus a_2x \oplus a_3y \oplus a_4 =$$
$$= 1 \wedge xy \oplus 1 \wedge x \oplus 0 \wedge y \oplus 1 = xy \oplus x \oplus 1.$$

Вопросы

1. Запишите структуру алгебры Жегалкина.
2. Перечислите основные законы алгебры Жегалкина.
3. Запишите тождества, позволяющие выразить основные операции булевой алгебры в алгебре Жегалкина.
4. Дайте определение понятию полином Жегалкина.
5. Какие булевы функции называются линейными?
6. Сколько различных полиномов Жегалкина можно построить для произвольной булевой функции?
7. Сформулируйте правило построения полинома Жегалкина.

Задания

1. Представить в виде полинома Жегалкина следующие логические функции:
   а) $(xz \vee y)(x\bar{y} \vee \bar{x}\,\bar{y} \vee \bar{z})(x \vee \bar{y})$;
   б) $((y \vee z)(t \vee y\bar{z})) \vee t\,\bar{x} \vee ((z \vee y)(\bar{t} \vee \bar{z}))$;
   в) $yt \vee ((z \vee \bar{t})(x \vee z)(\bar{t} \vee \bar{z})(x \vee \bar{z})) \vee \bar{y}t$;
   г) $(\bar{z} \vee t)(t \vee x) \vee (\bar{z} \vee \bar{x})(\bar{z} \vee \bar{t})(\bar{t} \vee z)$;
   д) $x\bar{t} \vee ((\bar{z}\,\bar{y}) \vee t)(z \vee y)) \vee ((\bar{t} \vee \bar{z})(z \vee y))$;
   е) $(((t \vee \bar{t}\,z)\bar{t}) \vee \bar{y})((y \vee t)(y \vee x))$;
   ж) $((z \vee \bar{x})(\bar{x} \vee \bar{t})(x \vee z)(\bar{y} \vee x)) \vee y\bar{t} \vee yt$;
   з) $(t \vee \bar{x}\,\bar{t} \vee x)(y \vee (t \vee tz))\bar{z}\bar{x}t$;
   и) $\bar{z}\,\bar{y} \vee tz \vee \bar{y}z \vee t\bar{z} \vee y\bar{t}$.
2. Исследовать на линейность следующие булевы функции:
   а) $((y \vee z) \leftarrow (y \oplus z)) \vee ((x \leftarrow t) \downarrow (x \sim t))$;
   б) $((z \to t) \mid (z \oplus t)) \mid ((x \sim y) \to (x \wedge y))$;
   в) $((z \mid t) \mid (\bar{z} \sim t)) \to ((x \oplus y) \wedge (y \to x))$;
   г) $((t \leftarrow x) \vee (t \sim x)) \leftarrow ((z \leftarrow y) \downarrow (\bar{z} \oplus y))$;
   д) $((z \to t) \leftarrow (z \sim t)) \vee ((x \wedge y) \downarrow (x \oplus y))$;
   е) $((z \vee t) \mid (z \sim t)) \mid ((\bar{x} \oplus \bar{y}) \to (x \leftarrow y))$;
   ж) $((y \mid z) \mid (y \sim z)) \to ((x \oplus t) \wedge (x \to t))$;
   з) $((y \downarrow \bar{t}) \vee (\bar{y} \oplus t)) \leftarrow ((x \leftarrow z) \downarrow (x \sim z))$;
   и) $(x \oplus \bar{y}) \leftarrow (y \wedge x)) \downarrow ((\bar{z} \sim \bar{t}) \downarrow (t \leftarrow z))$;

к) $((x \downarrow y) \vee (\overline{x} \sim y)) \leftarrow ((z \leftarrow t) \downarrow (z \sim t))$;
л) $((z \rightarrow x) \leftarrow (x \oplus \overline{z})) \wedge ((t \leftarrow y) \downarrow (y \sim t))$;
м) $((z \mid \overline{y}) \mid (z \sim \overline{y})) \mid ((\overline{x} \oplus \overline{t}) \rightarrow (\overline{x} \leftarrow \overline{t}))$;
н) $((z \downarrow \overline{y}) \vee (z \oplus \overline{y})) \leftarrow ((\overline{t} \leftarrow \overline{x}) \downarrow (\overline{t} \sim \overline{x}))$.

3. С помощью метода неопределенных коэффициентов построить полином Жегалкина для следующих функций:
а) $(yx \vee x\overline{z})(x \vee \overline{y}z(z \vee \overline{x}y))$;
б) $((x \vee (z \vee yz)))(z \vee \overline{x}\,\overline{z} \vee y)$;
в) $((x \vee \overline{y})(\overline{y} \vee \overline{z})(\overline{z} \vee x)) \vee ((\overline{y} \vee z))$;
г) $(x \vee z)(\overline{x}t \vee yt \vee \overline{xt} \vee y\overline{t})(x \vee z)$;
д) $(xy \vee x\overline{y}) \vee ((\overline{x} \vee y)(z \vee \overline{t})(\overline{x} \vee \overline{y})(t \vee z))$.

4.8. Полнота и замкнутость

Замкнутые классы булевых функций, функционально полная система булевых функций, теорема об одновременной полноте

Любая логическая функция может быть представлена с помощью операций булевой алгебры (п. 4.5) или алгебры Жегалкина (п. 4.7). Естественно возникают вопросы: «Существуют ли другие множества операций, с помощью которых можно определить любую булеву функцию?», «Какими свойствами они обладают?». Введем обозначение: P_2 — множество всех возможных булевых функций, зависящих от любого числа переменных.

Определение
Замыканием множества Σ ***булевых функций*** называется множество $[\Sigma]$, состоящее из функций, которые можно получить суперпозицией функций из Σ. Если $\Sigma = [\Sigma]$, то множество булевых функций Σ называется ***замкнутым классом***. Другими словами можно сказать, что множество Σ называется замкнутым классом, если любая суперпозиция функций из Σ также принадлежит Σ.

Определение
Система булевых функций $\Sigma = \{f_1, f_2, ..., f_n\}$ называется ***функционально полной***, если $[\Sigma] = P_2$.

С другой стороны, если существуют булевы функции, не принадлежащие $[\Sigma]$, то система Σ не является функционально полной.

Теорема 1. *Об одновременной полноте*

Пусть даны две системы булевых функций Σ_1 *и* Σ_2, *причем система* Σ_1 *— функционально полная. Если каждая функция из*

системы Σ_1 представлена в виде суперпозиции функций из системы Σ_2, то система функций Σ_2 также является функционально полной.

□ *Доказательство.* Пусть $f \in P_2$— некоторая булева функция. Представим ее в виде суперпозиии функций из Σ_1. Выразим каждую функцию полученной формулы через функции системы Σ_2, что по условию теоремы возможно. Полученная формула представляет функцию f и содержит только функции из Σ_2. Следовательно, система Σ_2 является функционально полной, что и требовалось доказать. ■

Пример. Доказать, что системы $\{\vee, \bar{}\}$, $\{\wedge, \bar{}\}$ – функционально полные, если известно, что система $\{\wedge, \vee, \bar{}\}$ является такой.

Решение. Выразим функцию $\wedge$ через функции системы $\{\vee, \bar{}\}$:

$$x \wedge y = \overline{\overline{x \wedge y}} = \bar{\bar{x} \vee \bar{y}}.$$

Следовательно, все функции полной системы $\{\wedge, \vee, \bar{}\}$ можно представить с помощью функций системы $\{\vee, \bar{}\}$. Таким образом, система $\{\vee, \bar{}\}$ является функционально полной.

Аналогично доказывается полнота системы функций $\{\wedge, \bar{}\}$, поскольку функцию $\vee$ можно выразить через функции системы $\{\wedge, \bar{}\}$ следующим образом:

$$x \vee y = \overline{\overline{x \vee y}} = \bar{\bar{x} \wedge \bar{y}}.$$

Утверждение доказано.

Вопросы

1. Как определяется замыкание множества булевых функций?
2. Определите понятие замкнутого класса булевых функций.
3. Какая система булевых функций называется функционально полной?
4. Перечислите основные замкнутые классы булевых функций.
5. Сформулируйте теорему о полноте двух систем булевых функций.

Задания

1. Доказать полноту следующих систем функций путем приведения их к известным полным классам:
   а) $\{x \to y, \bar{x}\}$;
   б) $\{x \downarrow y\}$;
   в) $\{x \to y, x \oplus y\}$;
   г) $\{x \vee y, x \oplus y, 1\}$;
   д) $\{x \sim y, 1, x \vee y\}$.
2. Проверить на полноту следующие классы функций:
   а) $\{x \wedge y, x \oplus y \oplus 1\}$;
   б) $\{0, 1, (x \wedge y) \vee z\}$;

в) $\{0, 1, x \sim y\}$;
г) $\{x \wedge y \oplus x, x \sim y, 0\}$;
д) $\{x \vee y, x \wedge y \oplus x \wedge z\}$;
е) $\{\overline{x}, (x \wedge y) \vee (x \wedge z) \vee (y \wedge z)\}$;
ж)$\{x \oplus y, x \vee y \vee \overline{z}\}$.

4.9. Функции, сохраняющие ноль и единицу. Монотонные функции

Отношение порядка для интерпретаций, признаки монотонности функции

Определение
Булева функция $f(x_1, x_2, \ldots, x_n)$ называется ***функцией сохраняющей 0***, если на нулевом наборе она равна 0: $f(0, 0, \ldots, 0) = 0$.

Определение
Булева функция $f(x_1, x_2, \ldots, x_n)$ называется ***функцией сохраняющей 1***, если на единичном наборе она равна 1: $f(1, 1, \ldots, 1) = 1$.

Функции $x \wedge y$ и $x \vee y$ сохраняют 0, поскольку $0 \wedge 0 = 0$ и $0 \vee 0 = 0$. Кроме того, данные функции также сохраняют 1, поскольку $1 \wedge 1 = 1$ и $1 \vee 1 = 1$. Функция $\overline{x}$ не сохраняет 0 и не сохраняет 1, так как $\overline{0} = 1$, $\overline{1} = 0$.

Пример. Определить, сохраняет ли 0 и 1 функция $f(x, y, z) = x \vee \overline{y}\,\overline{z}$.

Решение. Проверим значения данной функции на нулевом и единичном наборах.

$$f(0, 0, 0) = 0 \vee \overline{0} \wedge \overline{0} = 0 \vee 1 \wedge 1 = 0 \vee 1 = 1,$$
$$f(1, 1, 1) = 1 \vee \overline{1} \wedge \overline{1} = 1 \vee 0 \wedge 0 = 1 \vee 0 = 1.$$

Следовательно, данная функция сохраняет 1 и не сохраняет 0.

Рассмотрим важный класс булевых функций — монотонные булевы функции. Для этого введем ***отношение порядка***, которое будем обозначать символом $\leq$, ***для наборов булевых констант*** (интерпретаций) $\alpha = (a_1, a_2, \ldots, a_n)$, $\beta = (b_1, b_2, \ldots, b_n)$ следующим образом:

$$\alpha \leq \beta, \quad \text{если } a_i \leq b_i \text{ для всех } i = 1, \ldots, n.$$

Если хотя бы для одной пары (a_i, b_i) отношение $a_i \le b_i$ не выполняется, то соответствующие им наборы α и β в отношении порядка не участвуют, т.е. являются несравнимыми.

Пример. Определить отношение порядка для интерпретаций функции одной и двух переменных.

Решение. Для функции одной переменной $f(x)$ имеем два набора переменных: (0) и (1). Отношение порядка устанавливается следующим образом:

$$(0) \le (0),\ (0) \le (1),\ (1) \le (1).$$

Здесь все пары интерпретаций сравнимы.

Для функции двух переменных $f(x, y)$ имеем четыре интерпретации (0, 0), (0, 1), (1, 0) и (1, 1), для которых отношение порядка устанавливается следующим образом:

$$(0, 0) \le (0, 0);\ (0, 0) \le (0, 1);\ (0, 0) \le (1, 0);$$
$$(0, 0) \le (1, 1);\ (0, 1) \le (0, 1);\ (0, 1) \le (1, 1);$$
$$(1, 0) \le (1, 0);\ (1, 0) \le (1, 1);\ (1, 1) \le (1, 1).$$

Несравнимыми наборами являются, например, (0, 1) и (1, 0).

Определение

Булева функция f называется ***монотонной***, если для любых пар наборов значений переменных $(a_1, \ldots, a_n)$ и $(b_1, \ldots, b_n)$, для которых выполняется отношение $(a_1, \ldots, a_n) \le (b_1, \ldots, b_n)$, верно и неравенство $f(a_1, \ldots, a_n) \le f(b_1, \ldots, b_n)$.

Для проверки функции на монотонность необходимо исследовать выполнение неравенства $f(a_1, \ldots, a_n) \le f(b_1, \ldots, b_n)$ для всех пар наборов $(a_1, \ldots, a_n) \le (b_1, \ldots, b_n)$, кроме случая $(a_1, \ldots, a_n) = (b_1, \ldots, b_n)$, в котором значения функций совпадают.

Пример. Исследовать на монотонность функции $f(x, y) = x \wedge y$, $g(x, y) = x \oplus y$.

Решение. Для функции $f(x, y)$ запишем все наборы значений переменных, для которых выполняется отношение порядка, определим значение функции на данных наборах и сравним их:

$(0, 0) \le (0, 1)$, $f(0, 0) = 0, f(0, 1) = 0$, $f(0, 0) \le f(0, 1)$.
$(0, 0) \le (1, 0)$, $f(0, 0) = 0, f(1, 0) = 0$, $f(0, 0) \le f(1, 0)$.
$(0, 0) \le (1, 1)$, $f(0, 0) = 0, f(1, 1) = 1$, $f(0, 0) \le f(1, 1)$.
$(0, 1) \le (1, 1)$, $f(0, 1) = 0, f(1, 1) = 1$, $f(0, 1) \le f(1, 1)$.
$(1, 0) \le (1, 1)$, $f(1, 0) = 0, f(1, 1) = 1$, $f(1, 0) \le f(1, 1)$.

Вывод: функция $f(x, y) = x \wedge y$ является монотонной.

Аналогично проведем исследование функции $g(x, y)$.

$(0, 0) \le (0, 1), \quad g(0, 0) = 0, \quad g(0, 1) = 1, \quad g(0, 0) \le g(0, 1).$

$(0, 0) \le (1, 0), \quad g(0, 0) = 0, \quad g(1, 0) = 1, \quad g(0, 0) \le g(1, 0).$

$(0, 0) \le (1, 1), \quad g(0, 0) = 0, \quad g(1, 1) = 0, \quad g(0, 0) \le g(1, 1).$

$(0, 1) \le (1, 1), \quad g(0, 1) = 1, \quad g(1, 1) = 0, \quad g(0, 1) \not\le g(1, 1).$

Вывод: функция $g(x, y) = x \oplus y$ не является монотонной.

Теорема 1

Булева функция, отличная от констант 0 и 1, является монотонной, если и только если она допускает представление формулой булевой алгебры без отрицаний.

□ *Доказательство.* Пусть некоторая формула булевой алгебры без отрицаний представляет некоторую функцию f от n переменных. Преобразуем данную формулу в ДНФ, раскрыв скобки с помощью дистрибутивного закона. Полученная ДНФ также не будет содержать отрицаний. Предположим, что функция f немонотонна, тогда на какой-то интерпретации $\alpha \le \beta$ значение функции $f(\alpha) \not\le f(\beta)$, то есть $f(\alpha) = 1$, $f(\beta) = 0$. Равенство $f(\alpha) = 1$ означает, что ДНФ функции f содержит элементарную конъюнкцию, которая равна 1 на наборе α. Так как данная конъюнкция не содержит отрицаний, ее можно обозначить через $x_1 \wedge x_2 \wedge \ldots \wedge x_k$. Запишем набор значений α как $(a_1, a_2, \ldots, a_k, a_{k+1}, \ldots, a_n)$. Поскольку $x_1 \wedge x_2 \wedge \ldots \wedge x_k = 1$ на наборе α, то $a_1 \wedge a_2 \wedge \ldots \wedge a_k = 1$, и, следовательно, $a_1 = a_2 = \ldots = a_k = 1$. Исследуем теперь набор $\beta = (b_1, b_2, \ldots, b_k, b_{k+1}, \ldots, b_n)$. Изначально было сделано предположение, что $\alpha \le \beta$, следовательно, $b_1 = b_2 = \ldots = b_k = 1$. Тогда конъюнкция $x_1 \wedge x_2 \wedge \ldots \wedge x_k$ при подстановке значений из набора β равна единице, откуда следует, что $f(\beta) = 1$. Однако в начале доказательства было сделано предположение $f(\beta) = 0$. Значит, предположение ложно и функция f монотонна.

Пусть теперь f — монотонная функция от n переменных. Представим ее в ДНФ. Предположим, что данная ДНФ содержит элементарную конъюнкцию с отрицанием. Обозначим ее $\bar{x}_1 \wedge K$, где K — элементарная конъюнкция переменных $x_2, \ldots, x_k$. Рассмотрим некоторый набор $\alpha_1 = (0, a_2, \ldots, a_k, a_{k+1}, \ldots, a_n)$, на котором $\bar{x}_1 \wedge K = 1$, а значит, и $f(\alpha_1) = 1$. Рассмотрим набор $\alpha_2 = (1, a_2, \ldots, a_k, a_{k+1}, \ldots, a_n)$. Поскольку $\alpha_1 \le \alpha_2$ и функция f монотонна, то $f(\alpha_2) = 1$. Последнее утверждение будет справедливо только тогда, когда ДНФ

функции f содержит элементарную конъюнкцию $x_1 \wedge K$, которую набор α_2 обращает в единицу. Таким образом, ДНФ функции f можно записать в следующем виде: $\bar{x}_1 \wedge K \vee x_1 \wedge K$. Применив дистрибутивный закон и закон исключенного третьего, получим:

$$\bar{x}_1 \wedge K \vee x_1 \wedge K = (\bar{x}_1 \vee x_1) \wedge K = K.$$

Таким образом, переменная с отрицанием всегда может быть удалена из ДНФ монотонной функции. Теорема доказана. ■

Пример. Определить, является ли функция $f(x, y, z, t) = (\bar{x} \vee \bar{y}) \to (z \vee t)$ монотонной.

Решение. Выразим $f(x, y, z, t)$ через элементарные функции булевой алгебры:

$$(\bar{x} \vee \bar{y}) \to (z \vee t) = \overline{(\bar{x} \vee \bar{y})} \vee (z \vee t) = xy \vee z \vee t.$$

Полученная формула булевой алгебры не содержит отрицаний, следовательно, функция $f(x, y, z, t)$ является монотонной.

Вопросы

1. Дайте определения булевых функций, сохраняющих ноль и единицу.
2. Как по таблице истинности функции определить, сохраняет ли она ноль (единицу)?
3. Перечислите функции двух переменных, которые сохраняют ноль (единицу).
4. В чем заключается монотонность булевых функций?
5. Как определяется отношение порядка для пар наборов булевых констант?
6. Если функция f представлена формулой булевой алгебры с отрицаниями, то верно ли, что f немонотонна? Обоснуйте ответ.
7. Можно ли по виду ДНФ булевой функции судить о ее монотонности?

Задания

1. Доказать монотонность следующих функций:
   а) $x \wedge (y \vee z)$; в) $x \vee y \vee z$;
   б) $x \vee (y \wedge z)$; г) $x \wedge y \wedge z$.
2. Исследовать следующие функции на монотонность:
   а) $yx \oplus y$; г) $x \to (y \to x)$;
   б) $xy \oplus y \oplus x$; д) $x \to (x \to y)$;
   в) $x \sim y$; е) $(x \wedge y) \vee (x \wedge z) \vee (y \wedge z)$.
3. Доказать, что функция, двойственная монотонной, сама является монотонной.

4.10. Теорема Поста о полноте

Классы Поста, критерий полноты, несократимость и слабая полнота системы функций

Классами Поста называются следующие пять множеств булевых функций:

T_0 — класс функций, сохраняющих ноль;
T_1 — класс функций, сохраняющих единицу;
S — класс самодвойственных функций;
M — класс монотонных функций;
L — класс линейных функций.

Булева функция $f = f(x_1, ..., x_n)$ может принадлежать одному или нескольким классам Поста. Более того, ***проектирующая функция*** $f(x_1, ..., x_i ..., x_n) = x_i$ принадлежит всем пяти классам (убедитесь в этом). Однако, существуют функции, не входящие ни в один класс, например, штрих Шеффера $f_{14}(x, y) = \overline{x \wedge y}$. Действительно, полином Жегалкина $\overline{x \wedge y} = xy \oplus 1$ здесь нелинеен, так что $f_{14} \notin L$. Тот факт, что функция f_{14} не содержится ни в одном из классов T_0, T_1, S, M, сразу усматривается из таблицы истинности для f_{14} (см. таблицу 4.2).

Теорема 1

Каждый из классов Поста замкнут.

Это означает, что любая суперпозиция функций из одного и того же класса Поста приводит к функции того же класса. Действительно, если $f, g_i \in T_0$ и

$$\varphi(x_1, ..., x_n) = f(g_1(x_1, ..., x_n), ..., g_m(x_1, ..., x_n)), \text{ то}$$
$$\varphi(0, ..., 0) = f(g_1(0, ..., 0), ..., g_m(0, ..., 0)) = f(0, ..., 0) = 0;$$

поэтому $\varphi \in T_0$. Аналогично проверяется, что суперпозиция булевых функций не нарушает свойства самодвойственности, монотонности, сохранения нуля, линейности.

Оказывается, для произвольной системы $\Sigma = \{f\}$ булевых функций полнота Σ означает, что ***дополнение каждого из классов Поста*** содержит хотя бы одну функцию f системы Σ. Соответствующий критерий полноты, принадлежащий Посту, мы приводим без доказательства.

Теорема 2. *Критерий полноты Поста*

Система Σ булевых функций полна, если и только если она содержит хотя бы одну функцию, не сохраняющую ноль, хотя бы

одну функция, не сохраняющую единицу, хотя бы одну несамодвойственную функцию, хотя бы одну немонотонную функцию и хотя бы одну нелинейную функцию.

Заметим, что одна и та же функция может представлять в функционально полной системе одно или несколько требуемых свойств. Поэтому минимальное число булевых функций в функционально полном наборе равно единице. Одна функция может обладать всеми пятью требуемыми свойствами.

> **Определение**
> Полная система булевых функций называется ***несократимой***, если из нее нельзя исключить ни одной булевой функции без потери свойства полноты.

Следствие. *Максимальное количество булевых функций в несократимом функционально полном наборе равно четырем.*

Пример. Определить с помощью теоремы Поста, является ли система $\{\bar{}, \to\}$ функционально полной?

Решение. Отрицание представляется полиномом Жегалкина: $\bar{x} = x \oplus 1$, следовательно, функция отрицания линейна.

По таблице истинности 4.28 определяем, что функция отрицания самодвойственна, не сохраняет 0, не сохраняет 1, немонотонна. Импликация является нелинейной функцией (пример п. 4.7).

Таблица 4.28. $f(x) = \bar{x}$

x	$\bar{x}$
0	1
1	0

Таблица 4.29. $f(x) = x \to y$

x	y	$x \to y$
0	0	1
0	1	1
1	0	0
1	1	1

По таблице истинности 4.29 определяем, что импликация несамодвойственна, не сохраняет 0, сохраняет 1, немонотонна. Построим таблицу 4.30, в которой отметим знаком «v» наличие соответствующих свойств у функции — отрицания и импликации, а знаком «-» — отсутствие.

В каждой строке таблицы присутствует знак «-». Следовательно, для каждого класса Поста в данной системе имеется хотя бы одна функция, которая этому классу Поста не принадлежит. По теореме Поста такая система булевых функций является функционально полной.

Таблица 4.30. Свойства исследуемого набора

Название свойства	$\bar{x}$	$x \to y$
Линейность	v	-
Монотонность	-	-
Сохранение 0	-	-
Сохранение 1	-	v
Самодвойственность	v	-

В таблице 4.31 приведены результаты исследования на принадлежность классам Поста каждой булевой функции двух переменных. Знак «v» обозначает, что функция принадлежит классу, а знак «-» — означает, что функция не принадлежит классу.

Таблица 4.31. Принадлежность булевых функций двух переменных классам Поста

x	y	f_0	f_1	f_2	f_3	f_4	f_5	f_6	f_7	f_8	f_9	f_{10}	f_{11}	f_{12}	f_{13}	f_{14}	f_{15}
		$f = 0$	$x \vee y$	$\overline{y \leftarrow x}$	x	$\overline{x \leftarrow y}$	y	$x \oplus y$	$y \wedge x$	$y \rightarrow x$	$x \sim y$	$\bar{y}$	$x \leftarrow y$	$\bar{x}$	$y \leftarrow x$	$x \mid y$	$f = 1$
0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
0	1	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
1	0	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
1	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1
L: Линейность		v	-	-	v	-	v	v	-	-	v	v	-	v	-	-	v
M: Монотонность		v	v	-	v	-	v	-	v	-	-	-	-	-	-	-	v
T_0: Сохранение 0		v	v	v	v	v	v	v	v	-	-	-	-	-	-	-	-
T_1: Сохранение 1		-	v	-	v	-	v	-	v	-	v	-	v	-	v	-	v
S: Самодвойственность		-	-	-	v	-	v	-	-	-	-	v	-	v	-	-	-

На практике часто предполагается, что постоянные булевы функции-константы 0 и 1 заданы заранее. В таком случае к набору функций Σ фактически добавляют функции-константы 0 и 1.

Определение

Система булевых функций Σ называют ***функционально полной в слабом смысле***, если любая логическая функция может быть представлена суперпозицией функций из множества, полученного путем добавления к множеству Σ констант 0 и 1.

Из теоремы Поста следует вывод, что для функциональной полноты системы функции Σ в слабом смысле достаточно, чтобы в Σ содержались нелинейная и немонотонная функции, поскольку остальные необходимые свойства присутствуют у констант (константа единицы не сохраняет 0, константа нуля не сохраняет 1, и обе эти функции несамодвойственны).

При синтезе логических схем каждой функции соответствует логический элемент определенного типа, а функциональная полнота означает возможность реализовать любую булеву функцию с помощью элементов из соответствующей «полной» системы типов. Константы 0 и 1 не требуют специальных элементов для реализации, поэтому часто их считают данными. В таком случае систему типов логических элементов исследуют на функциональную полноту в слабом смысле.

Вопросы

1. Сформулируйте теорему о замкнутых классах булевых функций.
2. Что представляет собой суперпозиция несамодвойственной функции и отрицания?
3. Какую функцию можно получить с помощью суперпозиции немонотонной функции и констант 0 и 1?
4. Какая функция может быть представлена посредством суперпозиции нелинейной функции, отрицания и констант 0 и 1?
5. Сформулируйте теорему Поста.
6. Какая полная система булевых функций называется несократимой?
7. Сформулируйте следствие из теоремы Поста.
8. Дайте определение функционально полной в слабом смысле системы булевых функций. Наличия каких функций достаточно для выполнения условия функционально полной в слабом смысле системы?

Задания

1. С помощью теоремы Поста проверить на полноту наборы булевых функций:
   а) $\{x \vee y, \neg x\}$;
   б) $\{x \to y, \neg x\}$;
   в) $\{x \wedge y \oplus x, 1\}$;
   г) $\{x \oplus y, x \sim (y \wedge z)\}$;
   д) $\{x \wedge y, x \oplus y, x \sim (x \vee y)\}$;
   е) $\{x \sim y, 1, x \vee y\}$;
   ж)$\{x \wedge y, x \oplus y \oplus 1\}$;
   з) $\{0, 1, (x \wedge y) \vee z\}$;

и) $\{0, 1, x \sim y\}$;
к) $\{x \wedge y \oplus x, x \sim y, 0\}$;
л) $\{x \vee y, x \wedge y \oplus x \wedge z\}$;
м) $\{\overline{x}, (x \wedge y) \vee (x \wedge z) \vee (y \wedge z)\}$;
н) $\{1, \overline{x}, x \oplus y\}$;
о) $\{x \oplus y, x \vee y \vee \overline{z}\}$.

2. Доказать функциональную полноту каждой из функций: штрих Шеффера или стрелка Пирса.

4.11. Минимизация булевых функций

Основные понятия. Метод карт Карно (диаграмм Вейча), частично определенные функции, метод Квайна — Мак-Класки, метод Порецкого — Блейка

4.11.1. Основные понятия

Поиск наиболее простой логической формулы представления булевой функции имеет большое значение при формировании запросов к базам данных, в логическом программировании, в интеллектуальных системах.

Задача минимизации состоит в отыскании простейшей, согласно выбранному ***критерию минимизации***, формулы. Критерии могут быть различными, например: количество переменных в формуле, количество знаков конъюнкции и дизъюнкции или комбинация подобных критериев.

Ниже рассматривается минимизация на множестве ДНФ и КНФ количества содержащихся в формуле символов переменных и операций. Такая задача называется ***канонической задачей минимизации***. Минимальные формы, получаемые в результате ее решения, называются минимальными ДНФ и КНФ.

Определение
Импликантой некоторой функции f называется функция g, такая, что на всех интерпретациях, на которых g равна единице, f тоже равна единице.

Минтермы функции являются ее импликантами, элементарные конъюнкции, входящие в состав ДНФ функции, также являются ее импликантами.

Определение
Множество S, состоящее из импликант функции f, называется ***покрытием*** (или ***полной системой импликант***) функции f, если каждое единичное значение функции f покрывается единицей хотя бы одной импликанты из множества S.

Набор импликант, составляющих ДНФ функции, является ее покрытием. Набор всех конституент единицы функции, входящих в ее СДНФ, является покрытием данной функции.

Всякую элементарную конъюнкцию A, входящую в состав элементарной конъюнкции B и содержащую меньше переменных, чем конъюнкция B, называют ***собственной частью*** конъюнкции B, и говорят, что конъюнкция A ***покрывает*** конъюнкцию B.

Определение
Простой импликантой функции f называется такая конъюнкция-импликанта, что никакая ее собственная часть не является импликантой данной функции.

Множество всех простых импликант функции составляет ***покрытие*** данной функции.

Определение
Дизъюнкция всех простых импликант функции называется ее ***сокращенной ДНФ***.

Определение
Дизъюнктивным ядром булевой функции f называется такое множество ее простых импликант, которое образует покрытие f, но после удаления любой импликанты теряет это свойство, то есть перестает быть полной системой импликант.

Определение
Тупиковой ДНФ называется ДНФ данной булевой функции, состоящая только из простых импликант.

В отличие от сокращенной ДНФ, тупиковая ДНФ может не содержать некоторые из простых импликант функции. Каждая булева функция имеет единственную сокращенную ДНФ и может иметь несколько тупиковых ДНФ. В каждую из тупиковых ДНФ входят все импликанты дизъюнктивного ядра данной функции.

Определение
Минимальной ДНФ (***МДНФ***) данной булевой функции называется одна из ее тупиковых ДНФ, которой соответствует наименьшее значение критерия минимизации ДНФ.

Для нахождения множества простых импликант функции, заданной СДНФ, используются следующие очевидные преобразования формул булевой алгебры, которые традиционно называются «операциями».

Операция ***неполного дизъюнктивного склеивания***:

$$Ax \vee A\bar{x} = A \vee Ax \vee A\bar{x}.$$

Операция ***дизъюнктивного поглощения***:

$$A \vee Ax = A.$$

Выполнение двух указанных операций последовательно представляет собой выполнение операции ***полного дизъюнктивного склеивания***:

$$Ax \vee A\bar{x} = A.$$

Здесь A — некоторая элементарная конъюнкция переменных, x — булева переменная.

Пример. Пусть имеется функция f, заданная СДНФ:

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

Выполним операции полного склеивания следующим образом:

$$f(x, y, z) = x\, y\, z \vee \bar{x}\, y\, z \vee \bar{x}\,\bar{y}\, z \vee \bar{x}\,\bar{y}\,\bar{z} =$$
$$= (x\, y\, z \vee \bar{x}\, y\, z) \vee (\bar{x}\, y\, z \vee \bar{x}\,\bar{y}\, z) \vee (\bar{x}\,\bar{y}\, z \vee \bar{x}\,\bar{y}\,\bar{z}) =$$
$$= y\, z \vee \bar{x}\, z \vee \bar{x}\,\bar{y}.$$

Выполним теперь операции склеивания другим способом:

$$f(x, y, z) = (x\, y\, z \vee \bar{x}\, y\, z) \vee (\bar{x}\,\bar{y}\, z \vee \bar{x}\,\bar{y}\,\bar{z}) = y\, z \vee \bar{x}\,\bar{y}.$$

В обоих случаях получены тупиковые ДНФ функции $f(x, y, z)$. Вторая тупиковая ДНФ проще первой, поскольку содержит меньшее количество символов переменных и знаков операций. Построим таблицу истинности функции $f(x, y, z)$ и ее трех импликант, входящих в состав ДНФ (таблица 4.32).

Таблица 4.32. Покрытие единичных значений функции

x	y	z	f	$y\,z$	$\bar{x}\,z$	$\bar{x}\,\bar{y}$
0	0	0	1	0	0	1
0	0	1	1	0	1	1
0	1	0	0	0	0	0
0	1	1	1	1	1	0
1	0	0	0	0	0	0
1	0	1	0	0	0	0
1	1	0	0	0	0	0
1	1	1	1	1	0	0

На тех интерпретациях, на которых импликанта равна единице, функция также равна единице. Таким образом, каждая импликанта

«покрывает» некоторые единичные значения функции (в таблице это показано стрелками). Для анализа различных представлений булевой функции через КНФ и получения минимальных КНФ естественно трансформировать изложенные выше понятия и операции склеивания по принципу двойственности. Двойственные понятия соответственно называются терминами: ***имплицента***, ***простая имплицента***, ***полная система имплицент***, ***сокращенная КНФ***, ***тупиковая КНФ***, ***минимальная КНФ***. Используемые для минимизации КНФ операции склеивания имеют следующий вид:

Операция ***неполного конъюнктивного склеивания***:

$$(A \vee x)(A \vee \bar{x}) = A(A \vee x)(A \vee \bar{x}).$$

Операция ***конъюнктивного поглощения***:

$$A(A \vee x) = A.$$

Операция ***полного конъюнктивного склеивания***:

$$(A \vee x)(A \vee \bar{x}) = A.$$

Здесь A — некоторая элементарная дизъюнкция, x — булева переменная.

4.11.2. Минимизация булевых функций методом карт Карно (диаграмм Вейча)

Целью минимизации является нахождение минимальной из тупиковых ДНФ (КНФ), то есть нахождение минимального покрытия данной функции. Для этого необходимо построить все возможные тупиковые ДНФ (КНФ), используя операции склеивания и поглощения для данной функции. Методика Карно и Вейча позволяет выполнить указанные операции графически.

Карта Карно для ДНФ (диаграмма Вейча — для КНФ) является аналогом таблицы истинности, представленной в специальной форме. Значения переменных располагаются в заголовках строк и столбцов карты. Каждой конституенте единицы функции соответствует одна ячейка таблицы. Ноль или единица в ячейке определяет значение функции на данной интерпретации. Значения переменных располагаются так, чтобы соседние (имеющие общую границу) строки и столбцы таблицы отличались значением только одной переменной. Поэтому запись интерпретаций двух переменных в порядке (0, 0), (0, 1), (1, 0), (1, 1) недопустима, поскольку соседние наборы (0, 1) и (1, 0) отличаются значениями обеих переменных. В картах Карно интерпретации двух переменных располагаются в следующей последовательности: (0, 0), (0, 1), (1, 1), (1, 0).

В этой последовательности первая и последняя интерпретации также отличаются значением только одной переменной, поэтому первая и последняя строки (столбцы) таблицы считаются соседними (противоположные границы таблицы считаются совпадающими). При таком расположении минтермы, к которым применима операция склеивания, располагаются в соседних ячейках карты, и склеивание производится графически посредством объединения ячеек в группы.

Карты Карно для функций двух переменных имеют вид таблицы 2×2, где столбцы соответствуют значениям первой переменной, а строки — значениям второй переменной. На рис. 4.1 изображена структура карты Карно, в ячейках указаны соответствующие минтермы в виде формул с абстрактными переменными.

$y \backslash x$	0	1
0	$\bar{x}\,\bar{y}$	$x\,\bar{y}$
1	$\bar{x}\,y$	$x\,y$

Рис. 4.1. Структура карты Карно для двух переменных

Структура карты Карно для функций трех переменных имеет вид таблицы 2×4, где столбцы соответствуют всевозможным наборам значений первых двух переменных, а строки — значениям 0 и 1 третьей переменной (рис. 4.2). В ячейках структуры указаны соответствующие минтермы с абстрактными переменными.

$z \backslash xy$	00	01	11	10
0	$\bar{x}\,\bar{y}\,\bar{z}$	$\bar{x}\,y\,\bar{z}$	$x\,y\,\bar{z}$	$x\,\bar{y}\,\bar{z}$
1	$\bar{x}\,\bar{y}\,z$	$\bar{x}\,y\,z$	$x\,y\,z$	$x\,\bar{y}\,z$

Рис. 4.2. Структура карты Карно для функции трех переменных

Для конкретной булевой функции карта Карно заполняется следующим образом. В ячейки, соответствующие интерпретациям, на которых функция равна единице, записывают единицы. Указанные ячейки соответствуют конституентам единицы, присутствующим в СДНФ функции. В остальные ячейки записывают нули.

Пример. Построить карту Карно для функции

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

Решение. Построение карты Карно для заданной функции: помещаем единицы в ячейки, соответствующие минтермам, которые

присутствуют в данной СДНФ, и нули — в оставшиеся ячейки (рис. 4.3).

z \ xy	00	01	11	10
0	1	0	0	0
1	1	1	1	0

Рис. 4.3. Карта Карно для функции
$f(x, y, z) = \bar{x}\,\bar{y}\,\bar{z} \vee \bar{x}\,y\,z \vee \bar{x}\,\bar{y}\,z \vee x\,y\,z$

К конституентам единицы, соответствующим любым двум соседним ячейкам, можно применить операцию склеивания, так как они отличаются только одной переменной. Заметим, что на карте Карно для функции трех переменных каждая ячейка имеет три соседние ячейки, на карте Карно для функции четырех переменных — четыре, на карте Карно для функции пяти переменных — пять и т.д. При этом для ячеек на карте Карно функции трех переменных, расположенных в крайнем правом или в крайнем левом столбцах, соседними являются ячейки, расположенные непосредственно слева (или справа), выше (или ниже) в соседней строке, и крайняя левая (или правая) в этой же строке. В качестве примера на рис. 4.4 отмечены ячейки, соседние с ячейками A и B.

z \ xy	00	01	11	10
0				
1		A		

z \ xy	00	01	11	10
0				
1				B

Рис. 4.4. Ячейки, соседние с ячейками A и B

Правило склеивания ячеек и записи минимальной ДНФ

1. Строится карта Карно, соответствующая данной функции.

2. Ячейки объединяются в группы, обозначающие операции склеивания. В объединении участвуют только соседние ячейки, в которых находятся единицы.

3. В группу можно объединить только количество ячеек, равное 2^n, $n = 1, 2, 3, \ldots$. При этом группа может иметь только прямоугольную или квадратную форму.

4. Задача склеивания заключается в нахождении набора максимальных групп ячеек. Максимальная группа — это группа, которая не входит целиком ни в одну другую группу и соответствует

простой импликанте функции. Количество групп в таком наборе должно быть минимальным, так как такая группа соответствует минимальной тупиковой ДНФ. Каждая единица карты Карно должна входить хотя бы в одну группу, что обеспечивает покрытие функции полученным *набором импликант*.

5. Каждая группа ячеек, полученная после склеивания, соответствует той импликанте функции, реальные переменные которой имеют одинаковое значение для всех ячеек группы. Переменные берутся без отрицания, если им соответствуют единичные значения, и с отрицанием — в противном случае.

6. Дизъюнкция всех полученных простых импликант представляет собой результат минимизации формулы и является *минимальной* ДНФ.

Поскольку при минимизации на множестве ДНФ в склеивании участвуют только ячейки, содержащие единицы, нули в картах Карно, как правило, не указывают и подразумевают, что пустые ячейки содержат нули.

Пример. Найти минимальную ДНФ функции из предыдущего примера.

Решение. Опуская нули на рис. 4.3, записываем исходную карту Карно в виде рис. 4.5.

z \ xy	00	01	11	10
0	1			
1	1	1	1	

Рис. 4.5. Карта Карно функции $f(x, y, z)$

Графически склеивание обозначается объединением соседних единиц таблицы в группы, результатом чего являются две импликанты (рис. 4.6):

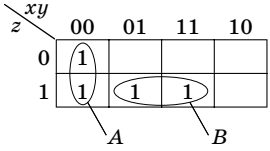


Рис. 4.6. Нахождение импликант функции $f(x, y, z)$

Импликанта $A = \bar{x}\,\bar{y}\,\bar{z} \vee \bar{x}\,\bar{y}\,z = \bar{x}\,\bar{y}\,(\bar{z} \vee z) = \bar{x}\,\bar{y}$.

Импликанта $B = \bar{x}\,y\,z \vee x\,y\,z = (\bar{x} \vee x)yz = yz$.

Таким образом, получим минимальную ДНФ:

$$f(x, y, z) = A \vee B = \bar{x}\,\bar{y} \vee yz.$$

Пример. Найти МДНФ для функции

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

Решение. Построим карту Карно для заданной функции (рис. 4.7).

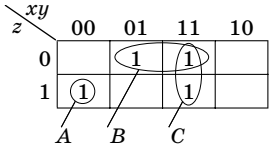


Рис. 4.7. Карта Карно для $f(x, y, z) = \bar{x}\,y\,\bar{z} \vee x\,y\,\bar{z} \vee \bar{x}\,\bar{y}\,z \vee x\,y\,z$

Минимальная ДНФ $f(x, y, z) = A \vee B \vee C = \bar{x}\,\bar{y}\,z \vee y\,\bar{z} \vee xy$.

Пример. Найти МДНФ следующей функции:

$$f(x, y, z) = x\,y\,z \vee x\,y\,\bar{z} \vee x\,\bar{y}\,z \vee x\,\bar{y}\,\bar{z} \vee \bar{x}\,y\,\bar{z} \vee \bar{x}\,\bar{y}\,z \vee \bar{x}\,\bar{y}\,\bar{z}.$$

Решение. Построим соответствующую карту Карно (рис. 4.8).

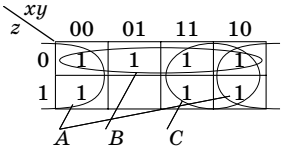


Рис. 4.8. Карта Карно для функции $f(x, y, z)$

Минимальная ДНФ $f(x, y, z) = A \vee B \vee C = \bar{y} \vee \bar{z} \vee x$.

Рассмотрим теперь применение карт Карно для минимизации функций, зависящих от четырех переменных. Карта Карно для функции четырех переменных имеет размер 4×4. Каждая ячейка имеет четыре соседних. Правила склеивания ячеек и записи результирующей формулы остаются прежними. Отличие состоит в том, что соседними необходимо считать не только крайний правый и крайний левый столбцы, но также крайнюю верхнюю и крайнюю нижнюю строки.

Пример. Построить минимальную ДНФ для функции

$$f(x, y, z, t) = x\,y\,z\,\bar{t} \vee x\,\bar{y}\,z\,t \vee \bar{x}\,y\,z\,t \vee$$
$$\vee\, \bar{x}\,\bar{y}\,z\,t \vee x\,\bar{y}\,\bar{z}\,t \vee \bar{x}\,\bar{y}\,\bar{z}\,t \vee x\,\bar{y}\,\bar{z}\,\bar{t} \vee \bar{x}\,\bar{y}\,\bar{z}\,\bar{t}.$$

Решение. Построим соответствующую карту Карно (рис. 4.9).

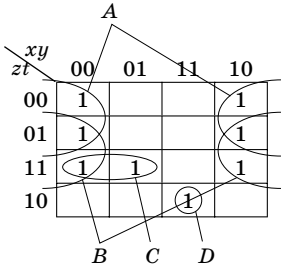


Рис. 4.9. Карта Карно для $f(x, y, z, t)$

Запишем минимальную ДНФ, объединив дизъюнкцией простые импликанты A, B, C, D соответственно:

$$f(x, y, z, t) = \overline{y}\,\overline{z} \vee \overline{y}\,t \vee \overline{x}\,z\,t \vee x\,y\,z\,\overline{t}.$$

Карта Карно для функции пяти переменных представляет в пространстве двухслойный параллелепипед, где каждый слой соответствует карте Карно от первых четырех переменных функции. Первый слой представляет собой на плоскости всевозможные интерпретации, при которых пятая переменная равна нулю, а второй — всевозможные интерпретации, при которых пятая переменная равна единице. Каждая ячейка на карте Карно для функции пяти переменных имеет пять соседних ячеек: четыре — на своем слое карты и пятая — на соседней, т.е. ячейку, которая совпадет с данной, если разместить слои карты один поверх другого. Объединение ячеек в группы на каждом слое карты осуществляется как и в случае четырех переменных. Кроме того, можно склеивать две одинаковые группы, находящиеся на разных слоях карты, расположение которых совпадет, если поместить один слой карты поверх другого.

Пример. Построить минимальную ДНФ для функции

$$f(x, y, z, t, w) =$$
$$= \overline{x}\,\overline{y}\,\overline{z}\,\overline{t}\,\overline{w} \vee \overline{x}\,y\,\overline{z}\,\overline{t}\,\overline{w} \vee x\,y\,\overline{z}\,\overline{t}\,\overline{w} \vee x\,\overline{y}\,\overline{z}\,\overline{t}\,\overline{w} \vee x\,y\,\overline{z}\,\overline{t}\,w \vee$$
$$\vee \overline{x}\,y\,\overline{z}\,\overline{t}\,w \vee \overline{x}\,\overline{y}\,z\,t\,\overline{w} \vee x\,y\,z\,t\,\overline{w} \vee x\,y\,z\,\overline{t}\,\overline{w} \vee x\,\overline{y}\,z\,\overline{t}\,\overline{w}.$$

Решение. Построим карту Карно для данной функции (рис. 4.10).

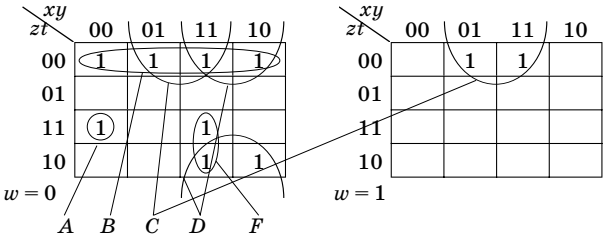


Рис. 4.10. Карта Карно для функции $f(x, y, z, t, w)$

Простые импликанты, входящие в минимальную ДНФ:

$$A = \overline{x}\,\overline{y}\,z\,t\,\overline{w}, \qquad B = \overline{z}\,\overline{t}\,\overline{w},$$
$$C = y\,\overline{z}\,\overline{t}, \qquad D = x\,\overline{t}\,\overline{w},$$
$$F = x\,y\,z\,\overline{w}.$$

Запишем минимальную ДНФ, объединив полученные импликанты знаками дизъюнкции:

$$f(x, y, z, t, w) = \overline{x}\,\overline{y}\,z\,t\,\overline{w} \vee \overline{z}\,\overline{t}\,\overline{w} \vee y\,\overline{z}\,\overline{t} \vee x\,\overline{t}\,\overline{w} \vee x\,y\,z\,\overline{w}.$$

Минимизация на множестве КНФ

Для минимизации булевой функции на множестве КНФ используются диаграммы Вейча. Их построение аналогично картам Карно. На карте помечаются ячейки, соответствующие интерпретациям, на которых функция равна нулю. После этого производится склеивание ячеек, содержащих нули и формирование минимальной КНФ. Склеивание ячеек производится по тем же правилам, что и при дизъюнктивной минимизации. Каждая группа ячеек, полученная в результате склеивания, соответствует дизъюнкции только тех переменных, которые имеют одинаковое значение для всех ячеек группы. Переменные берутся без отрицания, если им соответствует нулевое значение, и с отрицанием — в противном случае. Конъюнкция полученных элементарных дизъюнкций является результатом минимизации формулы.

Пример. Получить минимальную КНФ для функции, заданной СКНФ следующим образом:

$$f(x, y, z, t, w) =$$
$$= (x \vee y \vee z \vee t \vee w)\,(x \vee y \vee z \vee t \vee \overline{w})\,(x \vee y \vee \overline{z} \vee$$
$$\vee\, t \vee w)\,(x \vee y \vee \overline{z} \vee \overline{t} \vee w) \wedge (x \vee \overline{y} \vee \overline{z} \vee t \vee w)(x \vee \overline{y} \vee \overline{z} \vee$$
$$\vee\, \overline{t} \vee w)(\overline{x} \vee y \vee z \vee t \vee w)(\overline{x} \vee y \vee z \vee t \vee \overline{w})(\overline{x} \vee \overline{y} \vee \overline{z} \vee t \vee w) \wedge$$
$$\wedge\, (\overline{x} \vee y \vee \overline{z} \vee \overline{t} \vee \overline{w}).$$

Решение. Данная функция обращается в ноль на следующих наборах: (0, 0, 0, 0, 0), (0, 0, 0, 0, 1), (0, 0, 1, 0, 0), (0, 0, 1, 1, 0), (0, 1, 1, 0, 0), (0, 1, 1, 1, 0), (1, 0, 0, 0, 0), (1, 0, 0, 0, 1), (1, 1, 1, 0, 0), (1, 0, 1, 1, 1). Построим карту Карно для этой функции (рис. 4.11).

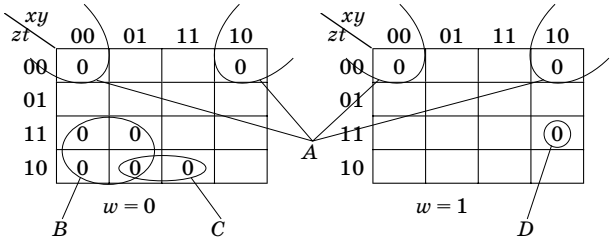


Рис. 4.11. Карта Карно для функции $f(x, y, z, t, w)$

Запишем минимальную КНФ, объединив знаками конъюнкции элементарные дизъюнкции:

$$f(x, y, z, t, w) = A \wedge B \wedge C \wedge D =$$
$$= (y \vee z \vee t)(x \vee \overline{z} \vee w)(\overline{y} \vee \overline{z} \vee t \vee w)(\overline{x} \vee y \vee \overline{z} \vee \overline{t} \vee \overline{w}).$$

Минимизация частично определенных функций

Если для решения задачи используются не все наборы входных данных, то допустимо любое значение функции на неиспользуемых интерпретациях и такая функция называется частично определенной. Другими словами, функция не определена на указанных интерпретациях. При минимизации такие функции доопределяются так, чтобы получить наиболее экономичную минимальную ДНФ (КНФ).

Пример. Функция $f(x, y, z, t)$ равна единице на наборах (0, 0, 1, 0), (0, 1, 1, 0), (1, 0, 1, 0), (1, 0, 0, 0) и не определена, если $xy = 1$. Необходимо построить минимальную ДНФ.

Решение. Составим карту Карно для заданной функции (рис. 4.12).

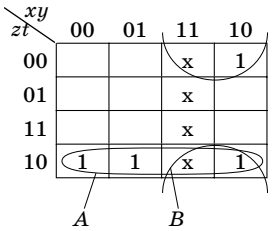


Рис. 4.12. Карта Карно для частично определенной функции

Минимальная ДНФ $f(x, y, z, t) = A \vee B = z\bar{t} \vee x\bar{t}$.

Метод карт Карно не является формальным и сложен в реализации. Кроме того, он не удобен в применении к функциям с количеством переменных более шести. Другой метод минимизации, который будет рассмотрен ниже, пригоден для программной реализации и может применяться при большом количестве переменных.

4.11.3. Минимизация функций методом Квайна — Мак-Класки

Метод минимизации Квайна — Мак-Класки также реализует переход от СДНФ к минимальной ДНФ с использованием операций склеивания и поглощения. Он был предложен В. Квайном, а затем усовершенствован Мак-Класки.

Алгоритм Квайна состоит из следующих шагов:

1. Записать СДНФ заданной функции.
2. Выполнить все возможные операции неполного дизъюнктивного склеивания. Результирующая формула является дизъюнкцией всех возможных импликант данной функции.
3. Выполнить все возможные операции дизъюнктивного поглощения. Результирующая формула является сокращенной ДНФ данной функции.
4. Составить импликантную таблицу и найти дизъюнктивное ядро.
5. Упростить импликантную таблицу посредством удаления строк, соответствующих импликантам дизъюнктивого ядра, и столбцов, соответствующих тем конституентам единицы, которые покрываются импликантами ядра.
6. Найти все тупиковые ДНФ данной функции.
7. Найти минимальную ДНФ.

Если при минимизации некоторой функции получается, что упрощенная импликантная таблица пуста, то тупиковая ДНФ этой функции является минимальной и состоит только из импликант дизъюнктивного ядра. Если же упрощенная импликантная таблица не пуста, то в каждой тупиковой ДНФ функции, кроме импликант ядра, присутствуют и некоторые простые импликанты, не входящие в дизъюнктивное ядро. Набор данных импликант определяет различия между тупиковыми ДНФ.

Пример. Найти методом Квайна минимальную ДНФ функции

$$f(x, y, z) = x\,y\,z \vee x\,\bar{y}\,\bar{z} \vee \bar{x}\,y\,z \vee \bar{x}\,\bar{y}\,z \vee \bar{x}\,\bar{y}\,\bar{z}.$$

Решение. Выполним все возможные операции дизъюнктивного склеивания и поглощения:

$$x\,y\,z \vee \bar{x}\,y\,z = y\,z,$$
$$x\,\bar{y}\,\bar{z} \vee \bar{x}\,\bar{y}\,\bar{z} = \bar{y}\,\bar{z},$$
$$\bar{x}\,y\,z \vee \bar{x}\,\bar{y}\,z = \bar{x}\,z,$$
$$\overline{x\,y}\,z \vee \bar{x}\,\bar{y}\,\bar{z} = \bar{x}\,\bar{y}.$$

Получим следующую формулу:

$$f(x, y, z) = x\,y\,z \vee x\,\bar{y}\,\bar{z} \vee \bar{x}\,y\,z \vee \overline{x\,y}\,z \vee \bar{x}\,\bar{y}\,\bar{z} =$$
$$= y\,z \vee \bar{y}\,\bar{z} \vee \bar{x}\,z \vee \bar{x}\,\bar{y}.$$

Произведя попарное сравнение всех элементарных конъюнкций, входящих в данную формулу, приходим к выводу, что получить другие элементарные конъюнкции с помощью операции склеивания в данной формуле невозможно. Таким образом, получена дизъюнкция всех возможных импликант данной функции:

$$f(x, y, z) = y\,z \vee \bar{y}\,\bar{z} \vee \bar{x}\,z \vee \bar{x}\,\bar{y}.$$

Полученная формула является сокращенной ДНФ данной функции.

Составим импликантную таблицу (таблица 4.33). Ее строки задаются простыми импликантами, а столбцы — конституентами единицы функции. Звездочкой отмечается каждая ячейка таблицы, для которой импликанта из строки является собственной частью конституеты из столбца.

Таблица 4.33. Импликантная таблица функции $f(x, y, z)$

	$x\,y\,z$	$x\,\bar{y}\,\bar{z}$	$\bar{x}\,y\,z$	$\bar{x}\,\bar{y}\,z$	$\bar{x}\,\bar{y}\,\bar{z}$
$y\,z$	*		*		
$\bar{y}\,\bar{z}$		*			*
$\bar{x}\,z$			*	*	
$\bar{x}\,\bar{y}$				*	*

Найдем дизъюнктивное ядро. В него входит каждая простая импликанта, которая является единственной в покрытии какой-либо конституенты единицы. В импликантной таблице по одному знаку «*» содержат столбцы, соответствующие конституентам единицы xyz и $x\bar{y}\,\bar{z}$, напротив простых импликант yz и $\bar{y}\,\bar{z}$. Эти простые импликанты составляют дизъюнктивное ядро. Составим упрощенную

импликантную таблицу. Для этого из импликантной таблицы вычеркиваем строки, соответствующие импликантам дизъюнктивного ядра, и столбцы, соответствующие конституентам единицы, которые покрыты импликантами ядра (таблица 4.34).

Таблица 4.34. Получение упрощенной импликантной таблицы

	$x\,y\,z$	$x\,\bar{y}\,\bar{z}$	$\bar{x}\,y\,z$	$\bar{x}\,\bar{y}\,z$	$\bar{x}\,\bar{y}\,\bar{z}$
$y\,z$	⊛		*		
$\bar{y}\,\bar{z}$		⊛			*
$\bar{x}\,z$			*	*	
$\bar{x}\,\bar{y}$				*	*

В данном случае импликанты ядра покрывают все конституенты единицы, кроме одной, поэтому упрощенная импликантная таблица имеет следующий вид (таблица 4.35):

Таблица 4.35. Упрощенная импликантная таблица

	$\bar{x}\,\bar{y}\,z$
$\bar{x}\,z$	*
$\bar{x}\,\bar{y}$	*

Из упрощенной импликантной таблицы находим, что тупиковые ДНФ данной функции, кроме дизъюнктивного ядра, включают или импликанту $\bar{x}\,z$, или $\bar{x}\,\bar{y}$. Таким образом, получаем две тупиковые ДНФ для данной функции:

$$\text{ДНФ1:}\quad f(x, y, z) = y\,z \vee \bar{y}\,\bar{z} \vee \bar{x}\,z;$$

$$\text{ДНФ2:}\quad f(x, y, z) = y\,z \vee \bar{y}\,\bar{z} \vee \bar{x}\,\bar{y}.$$

Указанные ДНФ содержат по 6 символов переменных, а также одинаковое количество знаков операций дизъюнкции (2) и конъюнкции (3), поэтому выберем в качестве минимальной ДНФ1, которая содержит меньше знаков операций отрицания.

В методе Квайна для функции большого числа переменных перечисление вариантов склеивания и множества тупиковых ДНФ представляет существенную сложность. Усовершенствование, введенное Мак-Класки, упрощает запись минимизируемой формулы. Согласно его предложению, конституенты единицы записываются в виде двоичного кода — номера данной конституенты. Указанный номер соответствует интерпретации, на которой конституента равна единице. Все остальные импликанты, получаемые в результате применения операций неполного склеивания, также записываются в виде двоичных кодов. В позициях кода, соответствующих

реальным переменным импликанты, помещается набор значений переменных, обращающий импликанту в единицу. В позициях фиктивных переменных импликанты ставится прочерк. Таким образом, операции склеивания и поглощения производятся не с самими импликантами, а с их двоичными кодами, что делает алгоритм простым в программной реализации.

Кроме того, множество кодов импликант делится на группы по количеству содержащихся в них единиц. Поскольку операции неполного склеивания применимы к импликантам, коды которых отличаются значением только в одной позиции, в склеивании могут участвовать только импликанты, принадлежащие к соседним группам, номера которых отличаются на единицу. Кроме того, выполнение операций склеивания производится пошагово. На первом шаге осуществляются все возможные склеивания конституент единицы, на втором шаге осуществляются склеивания на множестве импликант, полученных на первом шаге, и т.д. Деление на группы и шаги позволяет существенно сократить количество проверок пар импликант на возможность осуществления склеивания и делает алгоритм более эффективным.

При выполнении операций поглощения из формулы удаляются все импликанты, не являющиеся простыми. Результирующая формула представляет собой дизъюнкцию простых импликант функции — ее сокращенную ДНФ. Импликанты, участвовавшие хотя бы в одном склеивании, не являются простыми. Следовательно, вместо операций поглощения можно удалить из формулы все импликанты, участвовавшие хотя бы в одном склеивании, с одинаковым результатом. Поэтому при выполнении операций склеивания полученные импликанты помечаются. После выполнения всех возможных склеиваний помеченные импликанты удаляются. Это дает тот же результат, что и при выполнении операций поглощения.

Формальный метод нахождения множества всех тупиковых ДНФ функции был предложен С. Петриком. В этом методе каждой из простых импликант функции, не входящей в дизъюнктивное ядро, приписывается буквенное обозначение. Напомним, что поиск простых импликант производится в упрощенной импликантной таблице. С помощью данной таблицы записывается логическая формула покрытия конституент единицы простыми импликантами. Для этого для каждой конституенты единицы составляется дизъюнкция буквенных обозначений всех простых импликант, которые ее покрывают. Далее записывается формула, представляющая собой

конъюнкцию полученных элементарных дизъюнкций. В формуле раскрываются все скобки, после чего она приобретает вид дизъюнкции элементарных конъюнкций. Каждая элементарная конъюнкция в полученной формуле соответствует одной из тупиковых ДНФ функции. Для построения такой ДНФ необходимо записать импликанты, соответствующие буквенным обозначениям, которые входят в состав данной элементарной конъюнкции, и импликанты дизъюнктивного ядра функции.

Пример. Найти с помощью метода Квайна — Мак-Класки минимальную ДНФ функции $f(x, y, z, t)$, заданной следующей СДНФ:

$$f(x, y, z, t) = \bar{x}\,\bar{y}\,\bar{z}\,\bar{t} \vee \bar{x}\,\bar{y}\,\bar{z}\,t \vee \bar{x}\,\bar{y}\,z\,\bar{t} \vee \bar{x}\,\bar{y}\,z\,t \vee \bar{x}\,y\,\bar{z}\,t \vee$$
$$\vee \bar{x}\,y\,z\,t \vee x\,\bar{y}\,\bar{z}\,\bar{t} \vee x\,\bar{y}\,z\,\bar{t} \vee x\,y\,\bar{z}\,\bar{t} \vee x\,\bar{y}\,z\,t \vee x\,y\,\bar{z}\,t.$$

Решение. В таблице 4.36 запишем конституенты единицы данной функции в виде двоичных кодов во второй столбец. В первом столбце запишем десятичные номера интерпретаций.

Таблица 4.36. Двоичные коды конституент единицы функции

Десятичные номера интерпретаций	Двоичные коды конституент единицы	Конституенты единицы функции $f(x, y, z, t)$
0	0 0 0 0	$\bar{x}\,\bar{y}\,\bar{z}\,\bar{t}$
1	0 0 0 1	$\bar{x}\,\bar{y}\,\bar{z}\,t$
2	0 0 1 0	$\bar{x}\,\bar{y}\,z\,\bar{t}$
3	0 0 1 1	$\bar{x}\,\bar{y}\,z\,t$
5	0 1 0 1	$\bar{x}\,y\,\bar{z}\,t$
7	0 1 1 1	$\bar{x}\,y\,z\,t$
8	1 0 0 0	$x\,\bar{y}\,\bar{z}\,\bar{t}$
10	1 0 1 0	$x\,\bar{y}\,z\,\bar{t}$
11	1 0 1 1	$x\,\bar{y}\,z\,t$
12	1 1 0 0	$x\,y\,\bar{z}\,\bar{t}$
13	1 1 0 1	$x\,y\,\bar{z}\,t$

В соответствии с методом осуществим следующие шаги:

1. Сгруппируем двоичные коды импликант с одинаковым количеством единиц. Назовем число единиц m индексом группы. Упорядочим группы в порядке возрастания m.
2. Начиная с $m = 0$, произведем сравнение каждого двоичного кода в группе с индексом m с каждым кодом из группы с индексом $m + 1$. Если сравниваемые двоичные коды различны только в одном разряде, то в следующий столбец таблицы запишем

соответствующий им двоичный код с пустой меткой «-» на месте указанного разряда. Напротив каждого нового кода запишем номера кодов двух импликант, участвовавших в сравнении, и в следующем столбце эти импликанты пометим знаком «V», поскольку они не являются простыми импликантами. Все коды, которые остались непомеченными знаком «V», соответствуют простым импликантам, поэтому помечаем их знаком «X».

3. Если среди вновь полученных импликант имеются одинаковые, то из них для дальнейшего использования оставляем только одну.
4. Повторяем шаги 1–3 до тех пор, пока существует возможность получать новые коды импликант.

Результат выполнения описанных шагов приведен в таблице 4.37. Вначале заполняем три столбца нулевого цикла: m (десятичный индекс группы), двоичный код импликанты, номер импликанты. Импликанты делим на группы по значению m.

Таблица 4.37. Получение простых импликант функции $f(x, y, z, t)$ методом Квайна — Мак-Класки

ЦИКЛ 0				ЦИКЛ 1				ЦИКЛ 2			
m	Код импл.	№ импл.	Вид импл.	m	Код	№	Вид	m	Код	№	Вид
0	0000	0	V	0	000- 00-0 -000	0, 1 0, 2 0, 8	V V V	0	00-- ~~00--~~ -0-0 ~~-0-0~~	(0, 1), (2, 3) ~~(0, 2), (1, 3)~~ (0, 2), (8, 10) ~~(0, 8), (2, 10)~~	X X
1	0001 0010 1000	1 2 8	V V V	1	00-1 0-01 001- -010 10-0 1-00	1, 3 1, 5 2, 3 2, 10 8, 10 8, 12	V V V V V X	1	0--1 ~~0--1~~ -01- ~~-01-~~	(1, 3), (5, 7) ~~(1, 5), (3, 7)~~ (2, 3), (10, 11) ~~(2, 10), (3, 11)~~	X X
2	0011 0101 1010 1100	3 5 10 12	V V V V								
3	0111 1011 1101	7 11 13	V V V	2	0-11 -011 01-1 -101 101- 110-	3, 7 3, 11 5, 7 5, 13 10, 11 12, 13	V V V X V X				

Затем проводим сравнение первой импликанты из группы $m = 0$ (импликанта 0000) с первой импликантой из группы $m = 1$ (импликанта 0001). Они отличаются только в одном разряде, поэтому

производим их склеивание и получаем новую импликанту с кодом 000-. Записываем данный код в столбец кода импликанты цикла 1, а напротив импликант 0000 и 0001 ставим метки «V», поскольку они не являются простыми. Их номера указываем рядом с кодом 000-. Затем сравниваем импликанту 0000 со второй импликантой из группы $m = 1$ — импликантой 0010 и т.д.

Когда сравнение всех кодов импликант столбца «цикл 0» завершено, анализируем столбец вида импликанты. Напротив всех импликант ставим обозначение «V», так как в цикле 0 каждая импликанта допускает склеивание и не является простой. Далее разделяем на группы $m = 0$, $m = 1$ и $m = 2$ коды импликант цикла 1. Затем производим попарное сравнение этих импликант и получаем столбец кода импликант «цикл 2». Коды, содержащие знаки «-», могут образовывать новые импликанты, только если они содержат знаки «-» в одних и тех же разрядах. По окончании сравнения выделяем простые импликанты в столбце «вид» «цикла 1», которые не имеют метки «V», и помечаем их меткой «X». В столбце «код» «цикла 2» имеются одинаковые импликанты, поэтому произвольно вычеркиваем одну из одинаковых строк, чтобы каждая импликанта встречалась только один раз. Сравнивая коды импликант в«цикле 2», приходим к выводу, что методом склеивания из них невозможно получить новые импликанты. Все коды цикла 2 соответствуют *простым импликантам*, следовательно, построение таблицы завершено.

Для нахождения тупиковых ДНФ построим импликантную таблицу, в строках которой располагаем двоичные коды, соответствующие простым импликантам, а в столбцах — коды, соответствующие конституентам единицы. Если двоичный код строки является частью кода столбца (позиции со знаком «-»не сравниваются), то в соответствующую ячейку таблицы записываем знак «*» (таблица 4.38).

Таблица 4.38. Импликантная таблица функции $f(x, y, z, t)$

	0000	0001	0010	0011	0101	0111	1000	1010	1011	1100	1101
1-00							*			*	
-101					*						*
110-										*	*
00--	*	*	*	*							
-0-0	*		*				*	*			
0--1		*		*	*	*					
-01-			*	*				*	*		

Отметим столбцы таблицы, которые содержат по одному знаку «*». Соответствующие им простые импликанты являются *дизъюнктивными ядрами*. Вычеркиваем строки таблицы, соответствующие ядрам, и столбцы, покрытые ядрами, как показано в таблице 4.39.

Таблица 4.39. Получение упрощенной импликантной таблицы

	0000	0001	0010	0011	0101	0111	1000	1010	1011	1100	1101
1-00							*			*	
-101					*						*
110-										*	*
00--	*	*	*	*							
-0-0	*		*				*	*			
0--1		*		*	*	⊛					
-01-			*	*				*	⊛		

Получаем упрощенную импликантную таблицу (таблицу 4.40).

Таблица 4.40. Упрощенная импликантная таблица

		0000	1000	1100	1101
A	1-00		*	*	
B	-101				*
C	110-			*	*
D	00--	*			
E	-0-0	*	*		

Теперь найдем все тупиковые ДНФ функции $f(x, y, z, t)$ и выберем из них минимальную. Согласно методу Петрика, каждой из импликант таблицы приписываем буквенное обозначение и записываем формулу покрытия конституент единицы простыми импликантами. Конституента единицы с кодом 0000 может быть покрыта импликантой D или импликантой E, то есть дизъюнкции: $D \vee E$. Конституента единицы с кодом 1000 может быть покрыта ипликантой A или импликантой E: $A \vee E$. Аналогично 1100 может быть покрыта $A \vee C$, а 1101 — дизъюнкцией $B \vee C$. Таким образом, покрытие конституент единицы упрощенной импликантной таблицы функции $f(x, y, z, t)$ простыми импликантами можно записать в виде формулы покрытия:

$$(D \vee E)(A \vee E)(A \vee C)(B \vee C).$$

Если в данной формуле раскрыть скобки, то получится символьное представление всех возможных наборов простых импликант для тупиковых ДНФ, не включающее дизъюнктивные ядра:

$(D \vee E)(A \vee E)(A \vee C)(B \vee C) = (DA \vee EA \vee DE \vee E)(AB \vee$
$\vee CB \vee AC \vee C) = DAB \vee EAB \vee DEAB \vee EAB \vee DACB \vee$
$\vee EACB \vee DECB \vee ECB \vee DAC \vee EAC \vee DEAC \vee EAC \vee$
$\vee DAC \vee EAC \vee DEC \vee EC = ABD \vee ABE \vee ABDE \vee ABCD \vee$
$\vee ABCE \vee ACE \vee BCDE \vee ACDE \vee ACD \vee CDE \vee CBE \vee EC.$

В полученной формуле каждая конъюнкция буквенных обозначений соответствуют набору импликант в некоторой тупиковой ДНФ, в которую обязательно входят также дизъюнктивные ядра. Для того чтобы получить минимальную ДНФ, необходимо выбрать набор с минимальным количеством импликант (в данном примере это EC) и добавить импликанты ядра (0--1, -01-). Получаем минимальную ДНФ исходной функции f:

МДНФ $f(x, y, z, t) = \bar{y}\,\bar{t} \vee x\,y\,\bar{z} \vee \bar{x}\,t \vee \bar{y}\,z.$

Таким образом, с помощью метода Квайна — Мак-Класки получена минимальная ДНФ функции $f(x, y, z, t)$, которая содержит всего лишь 4 элементарные конъюнкции вместо 11 конституент единицы СДНФ.

Недостаток метода минимизации булевых функций Квайна — Мак-Класки состоит в необходимости записывать СДНФ функции, которая уже при семи переменных может содержать более ста конституент единицы. Следующий метод позволяет осуществлять дизъюнктивную минимизацию, используя в качестве исходной произвольную ДНФ функции.

4.11.4. Минимизация функций методом Порецкого — Блейка

Метод минимизации Порецкого — Блейка реализует переход от произвольной ДНФ функции к сокращенной ДНФ посредством операций обобщенного склеивания и поглощения.

Операция ***обобщенного склеивания***:

$$Ax \vee B\bar{x} = Ax \vee B\bar{x} \vee AB.$$

Докажем справедливость данного тождества:

$Ax \vee B\bar{x} = Ax(1 \vee B) \vee B\bar{x}(1 \vee A) = Ax \vee ABx \vee B\bar{x} \vee AB\bar{x} =$
$= Ax \vee B\bar{x} \vee AB(x \vee \bar{x}) = Ax \vee B\bar{x} \vee AB.$

Метод Порецкого — Блейка состоит в применении всевозможных операций обобщенного склеивания к ДНФ функции. Затем в полученной формуле производятся все возможные операции поглощения.

Пример. Найти сокращенную ДНФ по методу Порецкого — Блейка для функции $f(x, y, z) = xy\bar{z} \vee xz \vee \bar{x}y$.

Решение. Осуществим операции обобщенного склеивания:

$$xy\bar{z} \vee xz = xy \vee xy\bar{z} \vee xz,$$

$$xy\bar{z} \vee \bar{x}y = y\bar{z} \vee xy\bar{z} \vee \bar{x}y,$$

$$xz \vee \bar{x}y = yz \vee xz \vee \bar{x}y.$$

Очевидно, дизъюнкция левых частей совпадает с функцией f.

К вновь полученным импликантам может быть применена операция полного дизъюнктивного склеивания (п. 4.11.1):

$$y\bar{z} \vee yz = y, \quad xy \vee \bar{x}y = y.$$

Таким образом, исходная формула примет следующий вид:

$$f(x, y, z) = xy\bar{z} \vee xz \vee \bar{x}y = (xy \vee \bar{x}y) \vee xy\bar{z} \vee xz \vee (yz \vee y\bar{z}) =$$
$$= y \vee xy\bar{z} \vee xz \vee y = y(x\bar{z} \vee 1) \vee xz = xz \vee y.$$

Итак, $f = xz \vee y$ есть сокращенная ДНФ.

Вопросы

1. В чем заключается задача минимизации булевых функций? В чем особенность ее канонической формы?
2. Дайте определение понятию импликанты булевой функции.
3. Что представляет собой полная система импликант?
4. Какая импликанта называется простой?
5. Дайте определения сокращенной, тупиковой и минимальной дизъюнктивных нормальных форм?
6. Что называют дизъюнктивным ядром булевой функции?
7. Для каких целей служит импликантная таблица?
8. Запишите формулы операций дизъюнктивного склеивания и поглощения.
9. Запишите формулы операций конъюнктивного склеивания и поглощения.
10. Что представляет собой карта Карно (диаграмма Вейча)?
11. Сформулируйте правило склеивания ячеек и записи минимальной ДНФ по методу карт Карно.
12. В чем отличие применения диаграмм Вейча для минимизации на множестве КНФ от карт Карно?
13. Каким образом осуществляется минимизация частично определенных функций?
14. Перечислите основные шаги алгоритма минимизации Квайна.
15. Какие модификации предложил внести Мак-Класки в метод минимизации Квайна?

16. Для какой цели предназначен метод Петрика?
17. В чем заключается недостаток метода минимизации булевых функций Квайна — Мак-Класки?
18. В чем суть метода минимизации булевых функций Порецкого — Блейка?
19. Запишите формулу операции обобщенного склеивания.
20. Дайте сравнительную характеристику методов минимизации Порецкого — Блейка и Квайна — Мак-Класки.

Задания

1. Построить карты Карно для следующих функций:
   а) $f(x, y, z, t) = xyzt \vee \bar{x}yzt \vee \overline{xyzt} \vee xyz\bar{t}$;
   б) $f(x, y, z, t) = x\bar{y}zt \vee \bar{x}y\bar{z}t \vee \overline{xyzt} \vee \bar{x}yz\bar{t}$;
   в) $f(x, y, z, t) = \bar{x}yz\bar{t} \vee \overline{x\bar{y}zt} \vee x\bar{y}\bar{z}t \vee xyz\bar{t}$;
   г) $f(x, y, z, t) = xy\bar{z}t \vee \bar{x}yz\bar{t} \vee \overline{x\bar{y}zt} \vee \overline{x\bar{y}z\bar{t}}$;
   д) $f(x, y, z, t) = \bar{x}yzt \vee \overline{x\bar{y}zt} \vee xyzt \vee \overline{xyzt}$;
   е) $f(x, y, z, t) = (x \vee y \vee \bar{z} \vee t)(\bar{x} \vee y \vee z \vee \bar{t})(\bar{x} \vee \bar{y} \vee \bar{z} \vee t)(x \vee \bar{y} \vee z \vee t)(x \vee \bar{y} \vee z \vee \bar{t})$.
2. Найти минимальные ДНФ функций, заданных следующими картами Карно:

а)

zt \ xy	00	01	11	10
00	0	1	1	0
01	0	1	1	0
11	1	0	0	1
10	1	0	0	1

в) $v = 0$

xy \ zt	00	01	11	10
00	0	1	0	0
01	0	0	0	0
11	0	1	1	0
10	0	1	1	0

б)

zt \ xy	00	01	11	10
00	1	1	1	1
01	1	1	1	0
11	0	1	0	0
10	1	0	0	1

г) $v = 1$

xy \ zt	00	01	11	10
00	0	1	1	0
01	0	1	1	0
11	0	1	1	0
10	0	1	1	0

3. Найти минимальные КНФ для функций, заданных следующими диаграммами Вейча:

а)

zt \ xy	00	01	11	10
00	0	0	1	1
01	0	0	1	1
11	1	1	1	1
10	0	0	1	1

б)

zt \ xy	00	01	11	10
00	0	1	0	1
01	0	1	1	0
11	0	0	0	0
10	1	0	0	1

4. Найти минимальные ДНФ частично определенных функций, заданных следующими диаграммами Вейча:

а)

zt \ xy	00	01	11	10
00	-	0	0	-
01	1	-	1	1
11	0	0	-	0
10	-	0	-	-

б)

zt \ xy	00	01	11	10
00	1	-	1	1
01	-	1	-	-
11	0	-	0	0
10	1	0	0	1

5. Найти минимальные КНФ частично определенных функций, заданных следующими картами Карно:

а)

zt \ xy	00	01	11	10
00	1	1	1	-
01	0	0	1	1
11	-	-	1	-
10	1	1	1	-

б)

zt \ xy	00	01	11	10
00	-	0	1	1
01	1	1	-	-
11	0	-	0	0
10	1	1	0	-

6. Найти минимальные ДНФ функций методом Квайна — Мак-Класки. СДНФ функций заданы номерами конституент единицы следующим образом:
   а) $f(x, y, z, t) = \{0, 1, 2, 4, 5, 7, 8, 10, 11, 13, 14, 15\}$;
   б) $f(x, y, z, t) = \{0, 1, 2, 3, 4, 5, 7, 8, 10, 11, 13, 14\}$;
   в) $f(x, y, z, t) = \{0, 1, 2, 3, 5, 6, 7, 9, 10, 11, 13, 14, 15\}$;
   г) $f(x, y, z, t) = \{1, 2, 3, 4, 6, 7, 8, 9, 11, 12, 14, 15\}$;
   д) $f(x, y, z, t) = \{1, 3, 4, 5, 6, 8, 9, 10, 12, 13, 15\}$;
   е) $f(x, y, z, t) = \{2, 4, 5, 6, 7, 8, 9, 11, 12, 13, 14\}$.

7. Найти сокращенные ДНФ методом Порецкого — Блейка для следующих функций:
   а) $f(x, y, z, t) = yzt \vee \bar{x}zt \vee \overline{xy}t \vee xy\bar{z}$;
   б) $f(x, y, z, t) = x\bar{y}z \vee \overline{xz} \vee \overline{xyzt} \vee \bar{x}y$;
   в) $f(x, y, z, t) = \overline{xt} \vee \bar{x}zt \vee x\bar{y}z \vee xyz\bar{t}$;
   г) $f(x, y, z, t) = xy\bar{z} \vee \overline{zt} \vee \bar{x}zt \vee \bar{y}t$;
   д) $f(x, y, z, t) = xy\bar{z} \vee \bar{x}z\bar{t} \vee \overline{x\bar{y}}zt$;
   е) $f(x, y, z, t) = \bar{y}zt \vee \bar{x}t \vee xyzt \vee x\overline{yt}$.

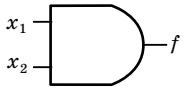
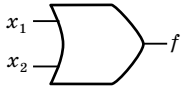
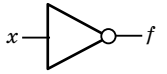
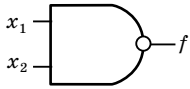
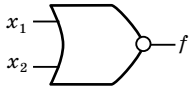
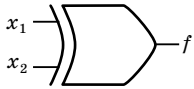
4.12. Логические схемы

Основные логические элементы, полнота набора логических элементов, анализ и синтез логических схем

Логические схемы в компьютерах и других электронных устройствах оперируют с наборами входных и выходных данных,

состоящими из нулей и единиц. Булева алгебра и булевы функции представляют математический аппарат для работы с такими данными и используются для анализа и синтеза ***логических схем*** (***цепей***). Основой построения логических схем является набор ***логических элементов***. Каждый логический элемент реализует некоторую булеву функцию. Его входы соответствуют булевым переменным, а выход — значению функции. Графические символы и названия наиболее часто используемых логических элементов представлены в таблице 4.41.

Таблица 4.41. Основные логические элементы

Название	Символ логического элемента	Булева функция
И		$f = x_1 \wedge x_2$ конъюнкция
ИЛИ		$f = x_1 \vee x_2$ дизъюнкция
НЕ		$f = \overline{x}$ отрицание
И-НЕ		$f = \overline{x_1 \wedge x_2}$ штрих Шеффера
ИЛИ-НЕ		$f = \overline{x_1 \vee x_2}$ стрелка Пирса
ИСКЛЮЧАЮЩЕЕ ИЛИ		$f = x_1 \oplus x_2$ XOR

Набор логических элементов ***полон***, если с его помощью можно реализовать любую булеву функцию. Для полноты такого набора необходимо и достаточно, чтобы соответствующий ему набор булевых функций был полон. В качестве базового набора элементов для решения конкретной задачи выбирают такой набор, с помощью

которого легче всего реализовать необходимые в данной задаче функции.

Логическая схема строится из набора базовых элементов, и представляет собой суперпозицию данных элементов так же, как формула является суперпозицией базовых функций булевой алгебры.

Пример. Построить логическую цепь, реализующую функцию $f(x, y, z) = (x \vee y)z$.

Решение. Используя логические элементы «И» и «ИЛИ», формируем суперпозицию, соответствующую данной функции. Полученная логическая цепь представлена на рис. 4.13.

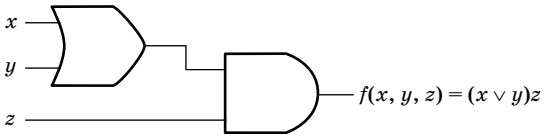


Рис. 4.13. Логическая цепь, реализующая функцию $f(x, y, z) = (x \vee y)z$

Логические элементы, реализующие операции со свойствами ассоциативности и коммутативности, на схемах могут изображаться с числом входов более двух, что означает многократное применение данной операции. На рис. 4.14 изображен элемент «ИЛИ», имеющий три входа и реализующий функцию $y = x_1 \vee x_2 \vee x_3$.

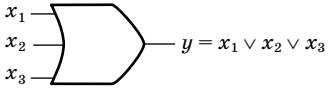


Рис. 4.14. Логический элемент «ИЛИ», имеющий три входа

При исследовании логических цепей возникают две основные задачи: ***анализ*** и ***синтез***. Анализ логической цепи заключается в построении булевой функции, которую реализует данное логическое устройство. Для этого определяется значение выходного сигнала на всех наборах входных данных и составляется таблица истинности функции. Используя таблицу истинности и правила построения СДНФ или СКНФ, можно построить формулу, соответствующую данной логической функции. С другой стороны, используя логическую схему, можно сначала построить формулу, соответствующую искомой функции, а затем, используя полученную

формулу, построить таблицу истинности функции. По данной логической схеме формулу можно построить, записав суперпозицию булевых функций, соответствующую схемной суперпозиции логических элементов.

Задача синтеза заключается в построении логической цепи для булевой функции, заданной таблично или с помощью формулы. Используя правила построения СДНФ и СКНФ, можно перейти от таблицы истинности функции к соответствующей формуле, а затем реализовать переменные и операции формулы логической цепью.

Стоимость логической цепи зависит от ее сложности. Поскольку экономически рентабельно производить логические цепи минимальной стоимости, булевы функции, по которым осуществляется построение цепей, должны быть предварительно минимизированы. Поэтому перед построением логической цепи необходимо получить минимальное представление функции.

Пример. Записать булеву функцию, которую реализует логическая цепь, представленная на рис. 4.17. Построить минимальную цепь, реализующую данную функцию.

Решение. Будем последовательно подавать на вход схемы интерпретации функции и определять ее значение на каждой из них (рис. 4.15).

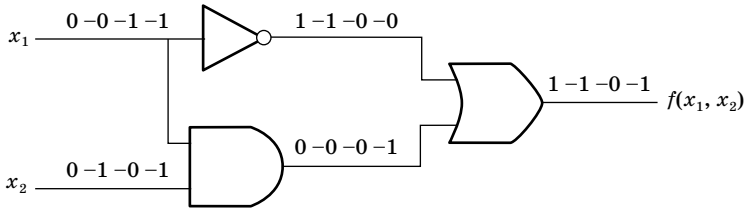


Рис. 4.15. Интерпретации и значения функции, реализованной логической цепью

Построим таблицу истинности искомой функции (таблица 4.42).

Таблица 4.42. Таблица истинности $f(x_1, x_2)$

x_1	x_2	$f(x_1, x_2)$
0	0	1
0	1	1
1	0	0
1	1	1

По таблице построим СКНФ данной функции:

$$f(x_1, x_2) = \bar{x}_1 \vee x_2.$$

Полученная формула является минимальной КНФ данной функции, так как содержит только одну конституенту нуля. Воспользуемся вторым способом и запишем формулу, соответствующую суперпозиции логических элементов, представленной на исходной схеме:

$$f(x_1, x_2) = \bar{x}_1 \vee x_1 x_2.$$

Данная формула эквивалентна полученной ранее, однако является более сложной. Используя минимальную формулу для функции $f(x_1, x_2)$, можно построить минимальную логическую цепь, реализующую данную булеву функцию (рис. 4.16).

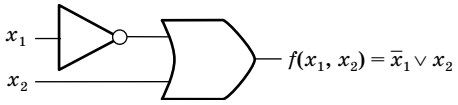


Рис. 4.16. Логическая цепь, реализующая функцию $f(x_1, x_2) = \bar{x}_1 \vee x_2$

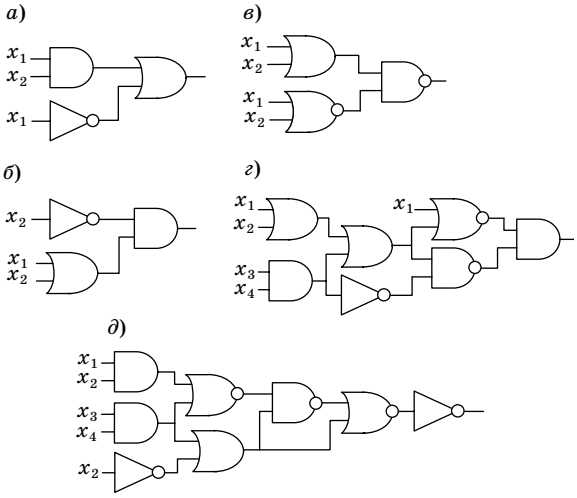
Таким образом, минимизация булевой функции позволила сократить исходную логическую схему на элемент «И».

Вопросы

1. Что является основой для построения логических схем?
2. Как булева алгебра связана с проектированием логических цепей?
3. Нарисуйте графические обозначения основных логических элементов.
4. Какой набор логических элементов называется полным?
5. Какому набору логических элементов следует отдать предпочтение при решении конкретной задачи?
6. Охарактеризуйте основные задачи, возникающие при исследовании логических цепей.
7. В чем заключается анализ логических цепей?
8. Каким образом осуществляется синтез логических цепей?
9. Какую операцию следует произвести над булевой функцией перед ее реализацией в виде логической цепи?

Задания

1. Провести анализ следующих логических цепей:



2. Осуществить синтез логических схем для следующих булевых функций:
   а) $x \to (x \to y)$;
   б) $(x \wedge y) \vee (x \wedge z) \vee (y \wedge z)$;
   в) $xy \oplus y \oplus x$;
   г) $\overline{z}\,\overline{y} \vee tz \vee \overline{y}z \vee t\overline{z}$;
   д) $(x \wedge y) \to z$;
   е) $x\,y\,z \vee x\,\overline{y}\,\overline{z} \vee \overline{x}\,\overline{y}\,z$.

раздел 5

Математическая логика

5.1. История и задачи математической логики

Как наука логика возникла еще в IV в. до н. э. в трудах древнегреческого философа Аристотеля, основоположника ***формальной логики***. Первое дошедшее до нас сочинение, специально посвященное логике, — «Аналитики» Аристотеля (384–322 г. до н. э.). Ему принадлежит открытие формального характера логического вывода, состоящего в том, что в наших рассуждениях одни предложения выводятся из других в силу определенной связи между их формой, структурой независимо от их содержания. В течение многих веков логика почти не развивалась. Истинный прогресс был достигнут только в XIX в., когда в логике стали применять математические методы.

Идею математизации логики высказал еще в XVII в. великий немецкий ученый Лейбниц. Он сформулировал задачу создания новой логики, которая бы была «искусством исчисления». В этой логике, по мысли Лейбница, каждому понятию соответствовал бы символ, а рассуждения имели бы вид вычислений. Время Лейбница было эпохой, когда аксиоматическая геометрия древних греков переживала новый расцвет. Математика представляла собой науку, в которой умозаключение играло более важную роль по сравнению с другими науками. Все математические теоремы опирались на точные доказательства, основанные на выводе следствий из общепринятых математических аксиом или постулатов. Поэтому неудивительно, что ученые при анализе математических

рассуждений открыли огромное количество схем (способов) построения умозаключений. Лейбниц решил так сформулировать правила математического доказательства, чтобы при их применении не приходилось больше думать о содержательном смысле математических выражений. Для этого необходимо создать исчисление, в котором естественные, содержательные доказательства были бы заменены формальными вычислениями и тем самым стали бы предметом математики. Такое исчисление, разумеется, предполагает специальную символику, в которой могут быть представлены аксиомы, теоремы и определения математики.

Только в середине XIX в. ирландский математик Дж. Буль частично воплотил в жизнь идею Лейбница. В его работах «Математический анализ», «Законы мышления», а также в работах Де Моргана были заложены основы булевой алгебры и алгебры логики, которые рассматривались в разделе 4. С помощью алгебры логики можно описывать рассуждения и «вычислять» их результаты, если символам переменных поставить в соответствие некоторые утверждения, называемые ***высказываниями***. Такое применение алгебры логики получило название ***логики высказываний*** и впоследствии было расширено добавлением переменных, принимающих значения из множества понятий, что сформировало ***логику предикатов***.

В связи с тем, что в математической логике принят символический язык, она также называется ***символической логикой***. В работах Пеано, Пирса и Шредера также создавалась и совершенствовалась математическая символика для законов мышления. Эти работы внушали веру в безграничные возможности формализации. Усилиями таких выдающихся ученых, как Рассел, Уайтхед, Гильберт, Бернайс, Гедель и Черч, формализация в рамках современных логических исчислений достигла высокого уровня. При попытках реализовать практически идею формализации возникли трудности логического характера, которые оказалось невозможно преодолеть средствами классической формальной логики. Эти трудности окончательно не устранены и по сей день, но попытки их преодоления дали мощный толчок развитию новых разделов логики — неоклассических, модальных, интуиционистских.

Широкое применение компьютеров потребовало соответствующего математического обеспечения: разработки специальных языков для баз данных и для представления знаний, углубленного логического анализа естественных языков. Эти проблемы и стремление смоделировать на компьютере обширный класс интеллектуальных процедур (идеи искусственного интеллекта) поставили перед ло-

гической наукой новые задачи. Результаты решения указанных задач находят в настоящее время все новые и разнообразные приложения во многих областях науки и техники. Математическая логика занимается формализацией некоторой области человеческого мышления, в том числе с целью предоставления возможности написания программы для вычислительной машины, которая в этом смысле приобретает способность рассуждать.

Вопросы

1. Кто является основателем формальной логики?
2. Каким образом происходило развитие логики как науки?
3. Что представляет собой математическая логика?
4. Перечислите возможные области применения математической логики.

Задания

1. Какие выводы будут правомерными при следующей посылке: *«Если студент не знает логику высказываний, то не сможет решить заданную логическую задачу»?*

   1.1. Студент решил заданную логическую задачу. Значит, студент знает логику высказываний.

   1.2. Студент не знает логику высказываний. Значит, он не решит заданную задачу.

   1.3. Студент знает логику высказываний. Следовательно, он решит заданную задачу.

   1.4. Студент не решил заданную логическую задачу. Следовательно, он не знает логику высказываний.

5.2. Понятия логики высказываний

Высказывание, истинностное значение, атом, логические связки, правильно построенная формула, интерпретация высказывания, приоритет и ранг операций, тавтология, тождественно ложная формула, необщезначимая формула

В естественных языках информация передается с помощью слов, объединенных в предложения. Формальная логика занимается анализом предложений, обращая основное внимание на форму и отвлекаясь от содержания.

Математическая логика изучает и моделирует только повествовательные предложения. Повелительные, восклицательные и вопросительные предложения находятся вне сферы рассмотрения.

Сокращенные предложения исключаются потому, что могут иметь двойное значение. Например, на вопрос *«не опоздали ли вы на лекцию?»* сокращенный ответ «*да*» непонятен. В данном случае в ответе необходимо более полное утверждение, которое могло бы быть формализовано. Для формализации подойдет предложение: «*Я опоздал на лекцию*». Математическая логика не изучает внутреннюю структуру и содержание предложений, а ограничивается рассмотрением их истинностных свойств, т.е. представляют ли они истину или ложь. В связи с этим из рассмотрения исключены повествовательные предложения, для которых невозможно определить, истинно оно или ложно. Очевидно, что такое представление естественного языка не является полным, но оно необходимо для применения логики высказываний.

Определение

Высказывание — это повествовательное предложение, о котором можно сказать, истинно оно или ложно, но ни то и другое одновременно.

Определение

Истина или ложь, приписанная некоторому высказыванию, называется ***истинностным значением*** этого высказывания. Обозначается: «Истина» — *И*, *T* (True) или 1, «Ложь» — *Л*, *F* (False) или 0.

Пример. Определить, какие из данных предложений являются высказываниями: «*Волга впадает в Черное море*», «*Волга впадает в Каспийское море*», «*Какой сегодня день?*», «*Расстояние от Земли до Солнца равно 150 000 000 км*».

Решение. Первые два предложения являются высказываниями, причем первое является ложным высказыванием, а второе — истинным. Третье предложение высказыванием не является (по определению), т.к. оно не повествовательное. Четвертое предложение также не является высказыванием, потому что его истинность или ложность зависит от требуемой точности.

Повествовательные предложения бывают простыми и сложными. Сложные предложения, как правило, состоят из простых предложений, соединенных союзами. Каждое простое предложение является самостоятельным утверждением, и оно уже не может быть разбито на более мелкие предложения. Эти простые предложения и союзы являются элементами словаря, необходимого для формализации естественного языка с помощью логики высказываний.

Определение

***Атомами* (*элементарными высказываниями*)** называются высказывания, которые соответствуют простым повествовательным предложениям, т.е. не имеют составных частей.

В качестве символов для обозначения атомов используются заглавные буквы латинского алфавита A, B, C, ... или заглавные буквы с индексами. Каждая буква в рассуждении должна обозначать одно и только одно элементарное высказывание.

Определение

Логика высказываний — это алгебраическая структура $(\{Л, И\}, \wedge, \vee, \bar{}, \rightarrow, \sim, Л, И)$ с носителем — двоичным множеством $\{Л$: «Ложь», $И$: «Истина»$\}$, операциями — ***логическими связками*** $\wedge$ — конъюнкция, $\vee$ — дизъюнкция, $\bar{}$ — отрицание, $\rightarrow$ — импликация, $\sim$ — эквивалентность и константами: $Л$ — ложь и $И$ — истина.

В разделе 4.2.3 главы 4 была определена *алгебра логики*, как алгебраическая структура $(B, \wedge, \vee, \bar{}, \rightarrow, \sim, 0, 1)$, образованная двоичным множеством $B = \{0, 1\}$ вместе с операциями конъюнкции, дизъюнкции, отрицания, импликации, эквивалентности и константами 0 и 1. Поскольку операции алгебры логики и логики высказываний одинаковы, а между множествами-носителями данных алгебраических структур $\{0, 1\}$ и $\{Л, И\}$ можно провести взаимно однозначное соответствие, приходим к выводу, что указанные алгебраические структуры изоморфны. Поэтому все утверждения, сделанные в главе 4 относительно алгебры логики, справедливы и для логики высказываний, в частности, коммутативный, ассоциативный и другие законы из п. 4.2.

Сложные предложения естественного языка состоят из простых предложений и служебных слов («если», «и», «то», «или» и т.п.) В грамматике указанные служебные слова называются связками (или союзами), поскольку они объединяют простые предложения в одно сложное. Например, два предложения «*Мы поедем летом в Крым*» и «*Мы поедем летом в горы*» можно объединить связкой «или» в одно сложное предложение «*Мы поедем летом в Крым, или мы поедем летом в горы*». Здесь связку «или» нельзя присоединить ни к первому, ни ко второму простому предложению, она обслуживает одновременно оба простых предложения и поэтому называется ***бинарной***. Рассмотрим оборот «*неверно, что...*», который

употребляется с целью отрицания. Например, в предложении «*Неверно, что жителей в Киеве меньше, чем в Харькове*» происходит отрицание предложения «*В Киеве меньше жителей, чем в Харькове*». Связка «неверно, что ...» является унарной, так как применяется к одному предложению. Кроме рассмотренных, в естественном языке существуют связки: «если..., то...», «или», «и», «либо», «ни... ни...», «...тогда и только тогда, когда...» и др.

Операции логики высказываний — логические связки — рассматриваются как формальные обозначения соответствующих им связок естественного языка (таблица 6.1).

Таблица 6.1. Логические связки в логике высказываний

Название	Обозначение	Аналоги естественного языка
эквивалентность	$\sim, \equiv, \leftrightarrow$	эквивалентно, равносильно, «тогда и только тогда», если и только если
импликация	$\rightarrow, \supset$	влечет, «если ..., то», «только если»
конъюнкция	$\wedge$, &	и
дизъюнкция	$\vee$	или, «или одно из двух... или оба», либо
отрицание	$\neg, \bar{}$	не, «неверно что»

Операции $\wedge, \vee, \rightarrow, \sim$ являются бинарными логическими связками, в отличие от операции $\neg$, которая является унарной. Пользуясь введенными логическими связками, можно из элементарных высказываний строить сложные высказывания, называемые ***формулами*** или ***молекулами***.

Определение

В логике высказываний ***правильно построенная формула*** определяется рекурсивно следующим образом:

1. Атом есть формула.
2. Если A и B — формулы, то (AB), $(A \vee B)$, $(A \rightarrow B)$, $(A \sim B)$ и $\neg A$ — также формулы.
3. Никаких формул, кроме порожденных указанными выше правилами, не существует.

Формулы логики высказываний, соответствующие сложным высказываниям, принимают значение *И* или *Л* в зависимости от значений элементарных высказываний, из которых они построены, и логических связок.

Определение
Приписывание истинностных значений атомам, из которых построено высказывание, называется ***интерпретацией высказывания***.

Для высказывания, содержащего n атомов, можно составить 2^n интерпретаций, так же, как и для n-местной булевой функции.

Наряду с высказываниями, истинностное значение которых независимо от ситуации можно считать однозначно определенным, как, например, «$2 \times 2 = 4$» $= И$, существуют выказывания, которые могут принимать различные значения. Например, высказыванию «*Завтра будет дождь*» можно придавать значения и «Истина», и «Ложь» в зависимости от конкретной ситуации.

Формулы логики высказываний можно задавать таблицами истинности, подобно булевым функциям. Приведем таблицу истинности для логических связок логики высказываний (таблица 5.2).

Таблица 5.2. Таблица истинности логических связок

A	B	$\neg A$	$A \wedge B$	$A \vee B$	$A \to B$	$A \sim B$
Л	*Л*	*И*	*Л*	*Л*	*И*	*И*
Л	*И*	*И*	*Л*	*И*	*И*	*Л*
И	*Л*	*Л*	*Л*	*И*	*Л*	*Л*
И	*И*	*Л*	*И*	*И*	*И*	*И*

Рассмотим примеры и выражения естественного языка, которые соответствуют логическим связкам.

Отрицание. Отрицание $\neg A$ истинно тогда и только тогда, когда A ложно. Данная унарная операция соответствует отрицанию в обычной речи, которое может иметь разные синтаксические выражения, например, предложение «*Неверно, что* у Ивана есть время» равнозначно предложению «У Ивана *нет* времени».

Конъюнкция. Высказывание $A \wedge B$, называемое ***конъюнкцией*** A и B, истинно тогда и только тогда, когда истинны оба высказывания A и B. Эта логическая операция соответствует в естественном языке связке «и», соединяющей два предложения.

Пример. Записать в виде формулы логики высказываний и определить истинностное значение следующих высказываний:

I — «6 *делится на* 3, *и* 10 *больше* 5»;
II — «6 *делится на* 3, *и* 7 *больше* 10».

Решение. Выделим атомы. Их три:

A — «6 делится на 3», B — «10 больше 5», C — «7 больше 10».

Тогда высказывание I будет соответствовать формуле $A \wedge B$, высказывание II — формуле $A \wedge C$. Будем считать, что высказывания A и B истинны, а высказывание C ложно. Используя истинностные значения высказываний A, B, C, определим значения высказываний I и II:

$$A \wedge B = И \wedge И = И; \quad A \wedge C = И \wedge Л = Л.$$

Дизъюнкция. Высказывание $A \vee B$, называемое ***дизъюнкцией*** A и B, ложно тогда и только тогда, когда ложны оба высказывания A и B.

Эта логическая операция соответствует соединению высказываний естественного языка с помощью связки «или», употребленной в смысле «неисключающее или»: «верно A, или верно B, или оба высказывания верны».

Пример. Записать в виде формулы логики высказываний и определить истинностное значение следующих высказываний:

I — «5 + 2 = 10 или 5 × 2 = 10», II — «6 − 3 = 2 или 3 × 2 = 5».

Решение. Выделим атомы:

A — «5 + 2 = 10»; C — «6 − 3 = 2»;

B — «5 × 2 = 10»; D — «3 × 2 = 5».

Тогда высказывание I будет соответствовать формуле $A \vee B$, высказывание II — формуле $C \vee D$. Высказывание B истинно, а высказывания A, C и D ложны, поэтому:

$$A \vee B = Л \vee И = И; \quad C \vee D = Л \vee Л = Л.$$

Импликация. Высказывание $A \to B$, называемое ***импликацией*** (условным предложением), ложно тогда и только тогда, когда A истинно, а B ложно.

В импликации $A \to B$ высказывание A называется ***посылкой*** (***условием***, ***антецедентом***), B — ***следствием*** (***заключением***, ***консеквентом***). Выражаемая импликацией причинно-следственная связь между A и B на естественном языке описывается следующими оборотами: «если A, то B», «A является достаточным основанием для B», «B, потому что A», «B, при условии выполнения A», «A влечет B» и т.д.

Используемые в точных науках понятия достаточного и необходимого условий можно формально выразить с помощью импликации. Именно для двух фактов (событий) A и B высказывание $A \to B$ означает, что «*A является достаточным условием для B*»

и одновременно, что «*B является необходимым условием для A*». Необходимость *B* для *A* выражается также в форме «*B только, если A*». Утверждение «*A является необходимым и достаточным условием для B*» эквивалентно двойной импликации $A \leftrightarrow B$, ибо

$$(A \to B) \wedge (B \to A) \equiv A \leftrightarrow B.$$

Рассмотрим несколько примеров.

В высказывании «*Если число n — четное (A), то n делится на 4 (B)*» условие *A* будет необходимым, но недостаточным. Ни одно нечетное число на 4 не делится, и в то же время есть четные числа, которые не делятся на 4. Итак, верна импликация $B \to A$, исходное высказывание $A \to B$ ложно.

В высказывании «*Если идет дождь (A), то на небе тучи (B)*» условие *A* будет достаточным, но не необходимым. Бывают случаи, когда на небе есть тучи, но дождя нет. Здесь верно исходное высказывание $A \to B$.

В высказывании «*Если геометрическая фигура — квадрат* (*A*), *то она — равносторонний прямоугольник* (*B*)» условие *A* будет и необходимым, и достаточным для выполнения *B*: $A \leftrightarrow B$.

Пример. Записать в виде формулы логики высказываний и построить таблицу истинности высказывания «*Если идет дождь, то над моей головой открыт зонтик*».

Решение. Введем атомы:

A — «идет дождь»;

B — «над моей головой открыт зонтик».

Тогда высказывание «Если идет дождь, то над моей головой открыт зонтик» будет соответствовать формуле $A \to B$. Результаты интерпретации данного высказывания сведены в таблицу 5.3.

Таблица 5.3. Таблица истинности высказывания

A	*B*	$A \to B$	Результат
Л	*Л*	*И*	останусь сухим
И	*Л*	*Л*	промокну
Л	*И*	*И*	останусь сухим
И	*И*	*И*	останусь сухим

Вторая строка таблицы 5.3 указывает на отсутствие причинно-следственной связи между событиями *A* и *B*.

Эквивалентность (эквиваленция). Если *A* и *B* — высказывания, то высказывание $A \sim B$ истинно тогда и только тогда, когда *A* и *B* либо оба истинны, либо оба ложны.

Эта операция соответствует в естественном языке оборотам: «...тогда и только тогда, когда...», «для того чтобы..., необходимо

и достаточно...». Например, «*Изучение дискретной математики будет успешным тогда и только тогда, когда будет освоена математическая логика*».

Используя таблицу истинности эквивалентности, можно доказать, что выражение $A \sim B$ эквивалентно выражению $(A \to B) \wedge (B \to A)$. Таким образом, логическая эквивалентность представляет собой импликацию в обоих направлениях, поэтому выражение «*A истинно тогда и только тогда, когда B истинно*» означает, что «*A* влечет *B*, и *B* влечет *A*».

Заменяя импликацию ее записью в виде СКНФ ($A \to B = \neg A \vee B$), получим:

$$A \sim B = (A \to B) \wedge (B \to A) = (\neg A \vee B) \wedge (\neg B \vee A),$$

$$A \sim B = (\neg A \vee B) \wedge (\neg B \vee A). \qquad (5.1)$$

Формула (5.1) дает СКНФ для эквивалентности. По принципу двойственности (п. 4.3) СДНФ для эквивалентности имеет вид:

$$A \sim B = A \wedge B \vee \neg A \wedge \neg B. \qquad (5.2)$$

Пример. Записать в виде формулы логики высказываний и определить истинностное значение высказываний:

I — «*Для того чтобы* $2 \times 2 = 4$, *необходимо и достаточно, чтобы* $2 + 2 = 4$»;

II — «$2 \times 2 = 5$ *равносильно тому, что* $3 \times 3 = 8$».

Решение. Введем обозначения атомов:

$$A \text{ — } 2 \times 2 = 4; \quad B \text{ — } 3 \times 3 = 8;$$

$$C \text{ — } 2 + 2 = 4; \quad D \text{ — } 2 \times 2 = 5.$$

Высказывание I соответствует формуле $A \sim C$, высказывание II — формуле $D \sim B$. Будем считать, что атомы A и C истинны, а атомы B и D — ложны, и определим истинностные значения сложных высказываний:

$$A \sim C = И \sim И = И; \quad D \sim B = Л \sim Л = И.$$

Прочтение формул сложных высказываний может быть неоднозначным, если не ввести скобки, указывающие, в каком порядке связываются между собой символы. Некоторые скобки можно опустить, введя последовательность выполнения или приоритет операций, таким же образом, как для операций алгебры логики:

$$\neg, \wedge, \vee, \to, \sim.$$

Например, следующие выражения без скобок равны формулам со скобками:

$$A \to B \wedge C = A \to (B \wedge C);$$

$$C \sim A \wedge B \to C = C \sim ((A \wedge B) \to C).$$

Всякой формуле логики высказываний можно поставить в соответствие некоторое сложное высказывание естественного языка и наоборот, «правильные» сложные предложения можно записать в виде формулы логики высказываний. Анализ сложного предложения необходимо начинать с определения следующего факта: является ли оно сокращенным вариантом более распространенного сложного предложения? Сокращенный вариант следует заменить полным вариантом предложения. Далее выделить простые предложения и заключить их в скобки, оставляя за скобками служебные слова, соединяющие простые предложения. Процесс заключения в скобки повторяется до тех пор, пока целиком все сложное предложение не окажется заключенным в скобки. После этого союзы и обороты естественного языка заменяются соответствующими логическими связками, а простые предложения — атомарными формулами.

Пример. Записать в виде формулы логики высказываний следующее предложение: «*Так как я лег поздно спать, я проспал и из-за этого не пошел на пару*».

Решение. Выделим простые предложения в данном сложном предложении и заключим их в скобки, оставляя служебные слова за их пределами:

«(Так как (я лег поздно спать),
(я проспал)) и из-за этого не (пошел на пару)».

Все три предложения связаны служебными словами, выражающими логические отношения. Кроме этого, перед третьим простым предложением стоит частица «*не*», соответствующая логической операции «отрицание». Третье простое предложение не является полным, поскольку разделяет общее подлежащее «*я*» со вторым простым предложением. Дополним третье предложение недостающим подлежащим и введем атомы P, Q, S следующим образом:

P — «Я лег поздно спать»;

Q — «Я проспал»;

S — «Я пошел на пару».

Заменим простые предложения символами атомов, а служебные слова — логическими связками, получим формулу логики высказываний:

$$(P \to Q) \to \neg S.$$

Пример. Построить формулу и таблицу истинности для высказывания: «*Если студент не подготовился к экзамену или ему попался сложный билет, то он не сдаст экзамен на положительную оценку*». Определить, в каких случаях данное высказывание окажется ложным.

Решение. Выделим простые высказывания и последовательность их соединения служебными словами с помощью скобок: «Если ((студент не подготовился к экзамену) или (ему попался сложный билет)), то (он не сдаст экзамен на положительную оценку)». Обозначим атомы:

A — «Студент подготовился к экзамену»;

B — «Студенту попался сложный билет»;

C — «Студент сдаст экзамен на положительную оценку».

Полученная формула имеет вид:

$$(\neg A \vee B) \to \neg C.$$

Построим соответствующую таблицу истинности (таблица 5.4).

Таблица 5.4. Таблица истинности $(\neg A \vee B) \to \neg C$

A	B	C	$\neg A$	$\neg A \vee B$	$\neg C$	$\neg A \vee B \to \neg C$
Л	*Л*	*Л*	*И*	*И*	*И*	*И*
Л	*Л*	*И*	*И*	*И*	*Л*	*Л*
Л	*И*	*Л*	*И*	*И*	*И*	*И*
Л	*И*	*И*	*И*	*И*	*Л*	*Л*
И	*Л*	*Л*	*Л*	*Л*	*И*	*И*
И	*Л*	*И*	*Л*	*Л*	*Л*	*И*
И	*И*	*Л*	*Л*	*И*	*И*	*И*
И	*И*	*И*	*Л*	*И*	*Л*	*Л*

По таблице мы видим, что существует три интерпретации: (*Л, Л, И*), (*Л, И, И*), (*И, И, И*), на которых исходное утверждение оказывается ложным. Интерпретация (*Л, Л, И*), означает, что студент не подготовился к экзамену, но получил несложный билет, и ему удалось сдать экзамен на положительную оценку. В случае (*И, И, И*) студенту попался сложный билет, но он подготовлен к этому экзамену и сдал экзамен на положительную оценку. В интерпретации (*Л, И, И*) студент не подготовился к экзамену, ему попался трудный билет, но он все равно сдал экзамен на положительную оценку.

Исходя из принимаемых формулами логики высказываний истинностных значений, формулы разделяются на тождественно истинные, тождественно ложные и необщезначимые.

Определение

Формула называется ***тождественно истинной*** (***тавтологией*** или ***общезначимой***), если она принимает значение «Истина» на всех интерпретациях (наборах значений переменных). Формула называется ***тождественно ложной*** (***противоречивой*** или ***невыполнимой***), если она принимает значение «Ложь» на всех интерпретациях. Формула называется ***необщезначимой***, ***нейтральной*** или ***непротиворечивой***, если она на одних интерпретациях принимает значение «Истина», а на других — «Ложь». Все формулы, не относящиеся к противоречивым, образуют множество ***выполнимых*** формул.

Определение

Рассуждение называется ***правильным***, если оно выражается тождественно истинной формулой.

Таким образом, проверить правильность рассуждения можно, построив соответствующую ему формулу и определив, является ли она тождественно истинной.

Доказать, что формула является тавтологией, можно двумя способами:

1. Построить таблицу истинности этой формулы. Тогда, если в таблице истинности на всех интерпретациях функция принимает значение «Истина», то соответствующее формуле рассуждение является тавтологией.
2. Применив к формуле тождественные преобразования, привести ее с помощью тождественных преобразований к виду одного из логических законов. Если в результате преобразований получим значение «Истина», то формула — тавтология.

Вопросы

1. Какой вид предложений моделирует формальная логика?
2. Приведите примеры предложений, которые не рассматриваются в формальной логике.
3. Дайте определение понятию «высказывание».
4. Что подразумевают под истинностным значением высказывания?
5. Какие высказывания называются атомами?
6. Дайте определение логики высказываний.
7. Что в логике высказываний называют логическими связками, перечислите их.

8. Покажите, что алгебра логики и логика высказываний изоморфны. Какой из этого следует вывод?
9. Дайте определение правильно построенной формулы.
10. Приведите примеры формул логики высказываний, содержащих любые логические связки, и соответствующих им предложений естественного языка.
11. Сформулируйте алгоритм записи сложного предложения естественного языка в виде формулы логики высказываний.
12. Перечислите виды функций логики высказываний с точки зрения принимаемых ими истинностных значений.

Задания

1. Являются ли следующие формулы общезначимыми, противоречивыми или непротиворечивыми:
   а) $\neg(\neg A) \to A$;
   б) $(A \to B) \to (B \to A)$;
   в) $(A \wedge (B \to A)) \to A$;
   г) $(A \vee \neg B) \vee (\neg A \vee B)$.
2. Расставить всевозможными способами скобки в следующих формулах:
   а) $\neg A \vee \neg B \wedge C$;
   б) $A \to B \to C \to D$.
3. Исключить как можно большее число скобок в формуле:
   а) $(\neg((A) \vee (C))) \vee (B)$;
   б) $(((A) \to (B)) \to (C)) \vee ((A) \to ((B) \to (C)))$;
   в) $((B) \sim (\neg(C))) \vee (((A) \to (A)) \to ((B) \vee (D)))$;
   г) $\neg((\neg((A) \vee (B))) \neg (C)) \to ((\neg((C) \to (D))) \vee E))$;
   д) $(\neg((B) \sim (C))) \wedge ((\neg(E)) \vee (\neg(A)))$;
   е) $((\neg((A) \to (B))) \vee (\neg((C) \vee (D))) \wedge \neg(F))$.
4. Постройте сложные высказывания с использованием только указанных операций:
   а) эквивалентность;
   б) импликация и конъюнкция;
   в) отрицание, конъюнкция и дизъюнкция.
5. Докажите, что отрицание высказывания «A есть достаточное и необходимое условие для B» эквивалентно высказыванию «$\neg A$ есть достаточное и необходимое условие для $\neg B$».
6. Постройте высказывание, эквивалентное $A \vee B$, используя только операции отрицания и конъюнкции.
7. Постройте сложное высказывание, эквивалентное $A \vee B$, используя только операции конъюнкции и отрицания.
8. Постройте сложное высказывание, эквивалентное $A \wedge B$, используя только операции дизъюнкции и отрицания.
9. Постройте два сложных высказывания, эквивалентных $A \to B$, используя только:

а) операции дизъюнкции и отрицания;
б) отрицания и конъюнкции.

10. Используя тождества, упростите формулы логики высказываний:
а) $\neg (A \vee B \vee C)\ (A\ (B \vee \neg C)) \wedge \neg B$;
б) $(A \vee B) \wedge \neg C \vee A \vee \neg C \vee B \vee A$.

5.3. Дедуктивные выводы в логике высказываний

Логическое следствие и его свойства, аксиомы, доказательство, правила дедуктивных выводов

Важнейшей характеристикой логического вывода является отношение совместимости между его посылками и заключением. В логике правила вывода используются, чтобы выводить одни истинные предложения из других истинных предложений.

Определение
Высказывание B является ***логическим следствием*** высказывания A, если формула $A \to B$ является *тождественно истинной*. Данное определение может быть обобщено на случай произвольного числа посылок следующим образом: **высказывание B называется *логическим следствием* высказываний $A_1\ A_2$, ..., A_n, если $A_1 \wedge A_2 \wedge ... \wedge A_n \to B$ — *тождественно истинная формула*.**

Пример. Показать, что высказывание $(A \wedge B) \vee \neg C$ является логическим следствием высказывания $A \wedge \neg C$.

Решение. Достаточно убедиться, что формула $(A \wedge \neg C) \to ((A \wedge B) \vee \neg C)$ является общезначимой. Используем тождества логики высказываний для эквивалентных преобразований, учитывая, что $x \to y = f_{13}(x, y) = \bar{x} \vee y$ (см. таблицу 4.3 п. 4.2).

$$(A \wedge \neg C) \to ((A \wedge B) \vee \neg C) = \neg (A \wedge \neg C) \vee ((A \wedge B) \vee \neg C) =$$
$$= \neg A \vee C \vee (A \wedge B) \vee \neg C = \neg A \vee (A \wedge B) \vee \neg C \vee C =$$
$$= \neg A \vee (A \wedge B) \vee И = И.$$

Таким образом, доказана общезначимость формулы.

Определение
Дедуктивным выводом называется вывод формулы B из формулы A, основанный на том, что B является логическим следствием A.

Пример. Доказать правильность рассуждения по дедукции: «*Резолюция принимается, если и только если за нее голосует большинство депутатов. За резолюцию не проголосовало большинство депутатов, поэтому резолюция не принимается*».

Решение. Посылками в этом выводе являются высказывания «Резолюция принимается, если и только если за нее голосует большинство депутатов» и «За резолюцию не проголосовало большинство депутатов», а заключением — «Резолюция не принимается». Введем следующие атомы:

P — «за резолюцию проголосовало большинство депутатов»;

Q — «резолюция принимается».

Тогда посылки и заключение обозначим соответственно через $P \sim Q$, $\neg P$, $\neg Q$, присоединим с помощью импликации к конъюнкции посылок $(P \sim Q) \wedge (\neg P)$ заключение $(\neg Q)$ и проверим, например, посредством тождественных преобразований, является ли импликация $((P \sim Q) \wedge (\neg P)) \to (\neg Q)$ логическим законом.

$((P \sim Q) \wedge (\neg P)) \to (\neg Q) = ((\neg P \vee Q) \wedge (P \vee \neg Q) \wedge (\neg P)) \to$ $\to (\neg Q) =$ — выразили эквивалентность через конъюнкцию, дизъюнкцию и отрицание (см. (5.1)).

$((\neg P \vee Q) \wedge (\neg P) \wedge (P \vee \neg Q)) \to (\neg Q) =$ — применили коммутативный закон.

$((\neg P) \wedge (P \vee \neg Q)) \to (\neg Q) =$ — использовали закон поглощения.

$((\neg P \wedge P) \vee (\neg P \wedge \neg Q)) \to (\neg Q) =$ — применили дистрибутивный закон.

$(\neg P \wedge \neg Q) \to (\neg Q) =$ — использовали закон противоречия.

$\neg(\neg P \wedge \neg Q) \vee (\neg Q) =$ — выразили импликацию через отрицание и дизъюнкцию. Теперь раскроем скобки и применим закон исключенного третьего:

$$P \vee Q \vee \neg Q = И.$$

Получено истинное высказывание. Следовательно, заключение следует из посылок, и заданное рассуждение удовлетворяет определению дедуктивного вывода. Истинность заключения в дедуктивном выводе гарантируется истинностью посылок.

Утверждение 1. Высказывание B является логическим следствием высказывания A, если высказывание $A \to \neg B$ является тождественно ложным.

Утверждение 2. Высказывание B является логическим следствием высказывания A, если на всех интерпретациях, на которых A истинно, B тоже истинно.

Тождественная истинность или ложность посылки импликации позволяет сделать вывод об истинности или ложности следствия.

Утверждение 3. Если высказывание B является логическим следствием высказывания A и высказывание A — тождественно истинное высказывание, высказывание B также является тождественно истинным.

Утверждение 4. Если высказывание A является тождественно ложным, то для любого высказывания B верно, что $A \to B$.

При создании математической логики преследовалась цель построения формального языка для математических рассуждений и доказательств. В математике и «чистой» логике доказывают теоремы, т.е. выводят следствия из определенных допущений. Допущения называются ***аксиомами*** или ***гипотезами***, при этом предполагается, что они тождественно истинны во всей рассматриваемой теории. ***Доказательство*** представляет собой логический вывод списка высказываний. Добавление высказывания в список доказательства возможно, если данное высказывание является следствием высказываний, внесенных в этот список ранее, или если оно является аксиомой или гипотезой. Теорема считается доказанной, если утверждение теоремы записано в список доказательства, то есть если установлено, что утверждение теоремы является логическим следствием введенных аксиом.

Правила для дедуктивного вывода строятся на основе общезначимых формул логики высказываний вида $A \to B$. Эти правила часто записывают как правила формального вывода в следующем виде:

$$\frac{A_1, ..., A_n}{B}.$$

Здесь A_1, ..., A_n — посылки вывода, а B — следствие. Тавтология, соответствующая такому правилу, — $A_1 \wedge A_2 \wedge ... \wedge A_n \to B$. Наиболее часто используемые правила дедуктивного вывода представлены в таблице 5.5.

Таблица 5.5. Правила дедуктивных выводов логики высказываний

Правило дедуктивного вывода	Тавтология	Название правила
$\frac{A}{A \vee B}$	$A \to (A \vee B)$	Правило введения дизъюнкции
$\frac{\begin{matrix}A,\\ B\end{matrix}}{A \wedge B}$	$((A) \wedge (B)) \to (A \wedge B)$	Правило введения конъюнкции

Продолжение таблицы 5.5

Правило дедуктивного вывода	Тавтология	Название правила
$\dfrac{A \vee B,\ \neg A}{B}$	$(A \vee B) \wedge \neg A \to B$	Правило удаления дизъюнкции (Дизъюнктивный силлогизм)
$\dfrac{A \wedge B}{A}$	$(A \wedge B) \to A$	Правило удаления конъюнкции
$\dfrac{A \to B}{\neg B \to \neg A}$	$(A \to B) \to (\neg B \to \neg A)$	Правило контрапозиции импликации
$\dfrac{A \to B,\ A}{B}$	$(A \wedge (A \to B)) \to B$	Правило отделения (Modus Ponens)
$\dfrac{\neg B,\ A \to B}{\neg A}$	$(\neg B \wedge (A \to B)) \to \neg A$	Отрицательная форма правила отделения (Modus Tollens)
$\dfrac{A \to B,\ B \to R}{A \to R}$	$((A \to B) \wedge (B \to R)) \to \to (A \to R)$	Гипотетический силлогизм

Из всех правил, приведенных в таблице 5.5, наиболее часто используется правило отделения. ***Правило отделения*** имеет следующий логический смысл: если посылка верна, то верно и следствие из нее. Приведем примеры рассуждений с помощью правила отделения:

«*Если студент не выучил теорию, то он не выполнит задание. Студент не выучил теорию. Следовательно, студент не выполнит задание*».

«Если студент получил пять, значит, он решил задачу. Студент получил пять. Следовательно, студент решил задачу».

Пример. Дано истинное высказывание «*Если n делится на* 9, *то n делится на* 3». Пусть также известно, что «*n делится на* 9». Какой вывод можно сделать, исходя из этих двух высказываний?

Решение. Введем атомарные высказывания:

A — «n делится на 9»;

B — «n делится на 3».

Высказывание «Если n делится на 9, то n делится на 3» можно представить в виде формулы $A \to B$. Из одновременного выполнения посылок $A \to B$ и A можем сделать вывод B по правилу отделения: «n делится на 3».

Пример. Определите тип правила дедуктивного вывода, которое было использовано в следующем рассуждении: «*Температура воздуха* $+1\,^{\circ}C$*, и идет дождь. Следовательно, температура воздуха* $+1\,^{\circ}C$».

Решение. Введем атомы:

A — «Температура воздуха $+1\,^{\circ}$C»;

B — «Идет дождь».

Высказывание «Температура воздуха $+1\,^{\circ}$C, и идет дождь» можно представить в виде формулы $A \wedge B$, а полученный по правилу удаления конъюнкции вывод «температура воздуха $+1\,^{\circ}$C» есть высказывание A. Очевидно, что вывод сделан в соответствии со строкой 2 таблицы 5.5).

Вопросы

1. В чем заключается отличие дедуктивных выводов от недедуктивных?
2. Дайте определение логического следствия одного (нескольких) высказываний.
3. Каким образом строится дедуктивный вывод?
4. Дайте краткую характеристику основных правил дедуктивного ввода.

Задания

1. Если конгресс отказывается принять новые законы, то забастовка не будет окончена, если она не длится более года и президент фирмы не уходит в отставку. Закончится ли забастовка, если конгресс отказывается действовать и забастовка только что началась? Постройте логический вывод и получите ответ.
2. Докажите следующее утверждение: «Из тождественно ложной формулы логически следует любая формула».
3. Докажите утверждение логики высказываний: «Тождественно истинная формула логически следует из любой формулы».

5.4. Исчисление высказываний

Язык, аксиомы и правила вывода, полнота и непротиворечивость, правила отделения и подстановки, теорема дедукции и ее следствие, доказательство методом от противного

Для логического анализа необходимо создать совокупность правил определения истинности или ложности высказываний.

Доказать то, что некоторая формула логики высказываний является тавтологией, можно, используя таблицу истинности, эквивалентные преобразования формулы, а также посредством

дедуктивного вывода. На базе логики высказываний создана ***формальная система — исчисление высказываний***, которая позволяет с помощью правил дедуктивного вывода проверить, является ли заданная формула общезначимой, а также получать общезначимые формулы логики высказываний.

На самом деле существуют различные формализации логики высказываний, то есть ***различные исчисления высказываний*** или по другой терминологии ***различные формальные системы***. Термин ***формальный*** означает, что объекты рассматриваются без интерпретации их содержания, значения или смысла. Операции над объектами производятся по строгим формальным правилам, не рассматривается смысловое значение проводимых операций.

Исчисление высказываний содержит язык, систему аксиом и правила вывода.

Язык исчисления высказываний состоит из правильно построенных формул логики высказываний.

Аксиомами исчисления высказываний является некоторое множество общезначимых формул логики высказываний.

Правила вывода позволяют получать новые формулы, которые являются истинными при условии истинности всех посылок, входящих в правило.

Теорема 1

Теоремы исчисления высказываний являются тождественно истинными формулами.

□ *Доказательство.* Теоремы — это формулы, которые являются логическим следствием множества аксиом данного исчисления. Аксиомы исчисления высказываний являются тождественно истинными формулами, а логические следствия тожественно истинных формул также являются тождественно истинными (утверждение 3, п. 5.3). Таким образом, теоремы исчисления высказываний являются тождественно истинными формулами, что и требовалось доказать. ■

Системы аксиом исчисления высказываний подбираются таким образом, чтобы исчисление обладало свойством *полноты*.

Полнота исчисления высказываний заключается в том, что в данной системе имеется достаточное количество аксиом для того, чтобы вывести любую формулу логики высказываний, которая является тождественно истинной.

Кроме того, исчисление высказываний обладает свойством непротиворечивости.

Теорема 2. *Непротиворечивость исчисления высказываний*

Не существует формулы A такой, что формулы A и $\neg A$ являются теоремами данного исчисления.

□ *Доказательство.* Воспользуемся методом от противного. Пусть верно, что A и $\neg A$ одновременно есть теоремы данного исчисления. По закону противоречия хотя бы одна из них не является общезначимой. С другой стороны, по теореме 1 все формулы исчисления высказываний являются общезначимыми. Полученное противоречие доказывает теорему. ■

Если ни одну из аксиом системы исчисления высказываний нельзя вывести из остальных, применяя правила вывода данной системы, то говорят, что система аксиом ***независима***. Аксиомы исчисления высказываний подбираются таким образом, чтобы они были независимы.

Кроме правила отделения (Modus Ponens, п. 5.3), в исчислении высказываний часто используется так называемое правило подстановки.

Правило подстановки

Пусть F_1 и F_2 — формулы логики высказываний, A — атомарная формула. Если $F_1(A)$ — формула, выводимая в исчислении высказываний, содержащая атом A, то $F_1(B)$ — выводимая формула, полученная заменой всех вхождений A в формуле F_1 на формулу F_2.

Правило подстановки записывается следующим образом:

$$\frac{F_1(A \parallel F_2)}{F_1(B)}.$$

Правило подстановки выражает тот факт, что если в тождественно истинной формуле все вхождения какого-либо атома заменить на некоторую формулу, то полученное выражение останется тождественно истинным.

Пример. Используя правило подстановки и коммутативный закон для дизъюнкции, доказать общезначимость следующей формулы:

$$A \vee B \wedge C \sim B \wedge C \vee A.$$

Решение. Запишем тождество, соответствующее коммутативному закону для дизъюнкции:

$$A \vee D \sim D \vee A.$$

Определим подстановку — атомарную формулу D заменим на $(B \wedge C)$:

$$A \vee (B \wedge C) \sim (B \wedge C) \vee A.$$

Опустив скобки в соответствии с приоритетом операций, убеждаемся в истинности исходной формулы.

Часто правило подстановки не упоминают явно, а используют его как очевидный факт, верный для тождеств. В этом случае аксиомы исчисления высказываний называют ***схемами аксиом***, подчеркивая то, что каждый атомарный символ в них может быть заменен на некоторую формулу.

Опишем две формальные системы исчисления высказываний.

Система S_1

I. Язык состоит из правильно построенных формул логики высказываний, содержащих операции $\{\neg, \rightarrow\}$. Алфавит языка совпадает с алфавитом логики высказываний и содержит, кроме символов перечисленных логических операций, символы скобок и символы для обозначения высказываний.

II. Аксиомы:

1. $A \rightarrow (B \rightarrow A)$;
2. $(A \rightarrow (B \rightarrow C)) \rightarrow ((A \rightarrow B) \rightarrow (A \rightarrow C))$;
3. $(\neg B \rightarrow \neg A) \rightarrow ((\neg B \rightarrow A) \rightarrow B)$.

III. Правила вывода:

1. Правило отделения (Modus Ponens);
2. Правило подстановки.

Система S_2

I. Язык состоит из правильно построенных формул логики высказываний, содержащих операции $\{\wedge, \vee, \neg, \rightarrow\}$. Алфавит языка совпадает с алфавитом логики высказываний и содержит, кроме символов перечисленных логических операций, символы скобок и символы для обозначения высказываний.

II. Аксиомы:

1. $A \rightarrow (B \rightarrow A)$;
2. $(A \rightarrow B) \rightarrow ((A \rightarrow (B \rightarrow C)) \rightarrow (A \rightarrow C))$;
3. $(A \wedge B) \rightarrow A$;
4. $(A \wedge B) \rightarrow B$;
5. $A \rightarrow (B \rightarrow (A \wedge B))$;
6. $A \rightarrow (A \vee B)$;
7. $B \rightarrow (A \vee B)$;
8. $(A \rightarrow C) \rightarrow ((B \rightarrow C) \rightarrow ((A \vee B) \rightarrow C))$;

9. $(A \to B) \to ((A \to \neg B) \to \neg A)$;
10. $\neg\neg A \to A$.

III. Правила вывода:

1. Правило отделения (Modus Ponens);
2. Правило подстановки.

Пример. Доказать выводимость формулы $A \to A$ в системе S_1. Процедуру доказательства проведем пошагово:

1. $(A \to ((A \to A) \to A)) \to ((A \to (A \to A)) \to (A \to A))$ — подстановка в аксиому 2 $(A \to A)$ вместо B, A вместо C.

2. $A \to ((A \to A) \to A)$ — подстановка в аксиому 1 $(A \to A)$ вместо B.

3. $(A \to (A \to A)) \to (A \to A)$ — по правилу 1, шаги 1, 2.

4. $A \to (A \to A)$ — подстановка в аксиому 1 A вместо B.

5. $A \to A$ — по правилу 1, шаги 3, 4.

Доказательство проведено.

Часто в математических рассуждениях истинность утверждения B доказывают в предположении истинности утверждения A, после чего приходят к выводу, что верно утверждение «если A, то B». Такой прием доказательства является верным и сформулирован в следующей теореме. Предварительно введем символ $\vdash$ тавтологии.

Теорема 3. *Теорема дедукции*

Если $A_1, \ldots, A_n, C \vdash B$, *то* $A_1, \ldots, A_n \vdash C \to B$. *В частности, если* $A \vdash B$, то $\vdash A \to B$.

Теорема 4. *Следствие из теоремы дедукции*

Из посылок $A \to B$, $B \to C$ *выводим* $A \to C$:

$$A \to B,\ B \to C \vdash A \to C.$$

Пример. Доказать, что формула $(\neg A \to \neg B) \to ((\neg A \to B) \to A)$ выводима в системе S_2.

1. $(\neg A \to B) \to (\neg A \to \neg B) \to \neg\neg A)$ — подстановка в аксиому 9 $\neg A$ вместо A.

2. $(\neg A \to B), (\neg A \to \neg B) \vdash \neg\neg A$ — теорема 3, шаг 1.

3. $\neg\neg A \vdash A$ — аксиома 10.

4. $(\neg A \to B), (\neg A \to \neg B) \vdash A$ — теорема 4, шаг 2, 3.

5. $(\neg A \to \neg B) \vdash (\neg A \to B) \to A$ — теорема 3, шаг 4.

6. $\vdash (\neg A \to \neg B) \to ((\neg A \to B) \to A)$ — теорема 3, шаг 5.

Доказательство проведено.

Все теоремы исчисления высказываний связаны с решением следующей задачи: «Вытекает ли данное утверждение из некоторой совокупности других утверждений?» Иначе говоря, доказывается тождественная истинность формулы вида $A \to B$, что можно осуществить несколькими методами. Вместо прямого логического вывода формулы B из формулы A часто оказывается удобным доказать противоречивость формулы $A \to \neg B$, тем самым доказав истинность формулы $A \to B$. В формуле $A \to \neg B$ присутствует отрицание следствия $\neg B$, поэтому такой метод доказательства называется ***доказательством от противного***.

Две схемы доказательства методом от противного

1. $(A \wedge \neg B) \to (C \wedge \neg C) \equiv A \to B$ — если из предположения, что A — верно, а B — неверно, следуют два противоречащих друг другу высказывания, то это означает, что из A следует B (см. утверждение 1).

2. $\neg B \to \neg A \equiv A \to B$ — если из предположения, что B — неверно, следует, что A неверно, то это означает, что из A следует B.

Таким образом, доказав истинность левой части одной из приведенных схем, доказывают истинность высказывания $A \to B$.

Вопросы

1. Что представляет собой исчисление высказываний?
2. Поясните понятия языка, аксиом и правил вывода исчисления высказываний.
3. Что является теоремами исчисления высказываний?
4. В чем состоит полнота и непротиворечивость исчисления высказываний?
5. Дайте определение независимой системы аксиом.
6. Перечислите правила вывода, которые наиболее часто применяются при построении исчисления высказываний?
7. Сформулируйте теорему дедукции и ее следствие.
8. В чем заключается метод доказательства от противного?
9. Дайте сравнительную характеристику двум схемам доказательства от противного.

Задания

1. Пусть A — «дверной замок сломан», B — «входная дверь открыта». Выпишите соответствующий логический вывод по правилу Modus Ponens.
2. Проверьте правильность следующих выводов:

а) $\dfrac{A \to B,\ \neg A \to B}{B}$; в) $\dfrac{A \to B,\ C \to \neg B}{A \to \neg C}$;

б) $\dfrac{A \to B,\ \neg B \to \neg C}{C \to A}$; г) $\dfrac{A \to B,\ \neg A \to C}{B \to C}$.

5.5. Логика предикатов

Порядок предиката, область определения предиката, терм, предметные переменные и константы

Напомним, что формализация естественного языка в логике высказываний осуществляется разбиением «языковых сообщений» на неделимые повествовательные предложения (атомы) и их смысловое объединение с помощью связок, причем внутренняя структура атомов не учитывается. Однако в естественном языке существует большое количество умозаключений, которые не могут быть формализованы описанным способом. Рассмотрим классическое умозаключение:

«*Каждый человек смертен.*
Так как Сократ человек, то он смертен».

Интуитивно указанный логический вывод представляется корректным. Введем следующие атомы:
A — «каждый человек смертен»;
B — «Сократ — человек»;
C — «Сократ смертен».

Тогда исходное умозаключение будет соответствовать формуле логики высказываний $A \wedge B \to C$.

Приведем данную формулу к нормальной форме:

$$A \wedge B \to C = \neg(A \wedge B) \vee C = \neg A \vee \neg B \vee C.$$

На интерпретации (1,1,0) полученная формула равна нулю, следовательно, данная формула не является общезначимой, т.е. в рамках логики высказываний C не является логическим следствием A и B. Такая ограниченность возможностей нашей формализации связана с тем, что в атоме A не учитывается внутренняя смысловая особенность обобщения «каждый». Это вызывает необходимость усовершенствовать логику высказываний с тем, чтобы она полнее объясняла способности человека делать логические выводы. Для этой цели в ***логику предикатов*** введены дополнительные, новые по сравнению с логикой высказываний, логические понятия, а именно: ***терм***, ***предикат*** и ***квантор***.

Определение

Определен некоторый ***предикат***, если:

а) задано некоторое (произвольное) множество, называемое областью определения предиката (предметная область);

б) фиксировано множество {1, 0}, называемое областью значений;

в) указано правило, с помощью которого каждому элементу, взятому из предметной области, ставится в соответствие один из двух элементов из области значений.

Понятие предиката является частным случаем понятия функции, для которой четко фиксирована область значений.

Название предикат происходит от английского слова predicate, означающего высказывание или сказуемое. Предикатом чаще всего обозначают свойство или действие, выраженное в высказывании сказуемым, а объекты и субъекты этого действия, а также другие члены предложения являются аргументами данного предиката. В качестве обозначения предиката часто выбирают слово, отражающее его смысловое значение, или заглавную букву латинского или другого алфавита.

Определение

Предикат P, имеющий n аргументов, называется ***n-местным предикатом***, обозначается $P(x_1, x_2, ..., x_n)$.

Определение

Количество аргументов предиката $P(x_1, x_2, ..., x_n)$ называется его ***порядком***.

Так, например, высказывание «x — действительное число» можно представить одноместным предикатом, «y меньше z» — двуместным предикатом, а «x и y родители z» — трехместным предикатом. Если x, y и z замещены конкретными значениями (объектами), то предикат переходит в высказывание, которое рассматривается как *нульместный предикат*. Например: «Терм и квантор — понятия логики предикатов». Таким образом, если количество аргументов предиката $P(x_1, x_2, ..., x_n)$ n переменных равно нулю, то предикат является высказыванием; если $n = 1$, то предикат соответствует свойству; если $n = 2$, то предикат является бинарным отношением; если $n = 3$, то предикат — тернарное отношение.

Пример. Представить в виде предикатов высказывания: «x делится на 13», «x делится на y», «x — простое число».

Решение. Выберем в качестве названий предикатов действия или свойства данных предложений: ДЕЛИТСЯ, ПРОСТОЕ. Тогда заданные высказывания можно записать в виде предикатов следующим образом: ДЕЛИТСЯ(x, 13), ДЕЛИТСЯ(x, y), ПРОСТОЕ(x). Здесь первый и третий предикаты являются одноместными и каждый выражает некоторое свойство чила x; второй предикат — двуместный и выражает бинарное отношение делимости на множестве чисел.

В логике предикатов существует понятие ***функционального символа***. Например: минус (x, y) — функциональный символ «$x - y$»; отец(x) — функциональный символ «отец человека x». Если функциональный символ имеет n аргументов, то он называется ***n-местным функциональным символом***, например: минус(x, y) — двухместный функциональный символ. Индивидуальный символ или константа может рассматриваться как функциональный символ без аргументов.

Итак, для построения атомов логики предикатов разрешается использовать следующие типы символов:

1. ***Индивидуальные символы*** или ***константы***, которые обычно являются именами объектов, например: Сократ, 13.
2. ***Символы предметных переменных***, в качестве которых обычно выступают буквы латинского алфавита, возможно, с индексами, например: x, y, b_2.
3. ***Функциональные символы*** — строчные буквы латинского алфавита или осмысленные слова из строчных букв, например: минус, отец.
4. ***Предикаты*** — прописные буквы или осмысленные слова из прописных букв, например: P, Q, ДЕЛИТСЯ, БОЛЬШЕ, ПРОСТОЕ.

Определение

Аргументы предиката называются ***термами***. ***Терм*** определяется рекурсивно следующим образом:

1. Константа есть терм.
2. Переменная есть терм.
3. Если f является n-местным функциональным символом, а t_1, t_2,..., t_n — термы, то $f(t_1, t_2, ..., t_n)$ есть терм.
4. Никаких термов, кроме порожденных с помощью указанных выше правил, не существует.

Определение
Термы принимают значения из заранее определенного множества, которое называется ***предметной областью***.

Определение
Термы-константы и термы-переменные называются ***предметными константами*** и ***предметными переменными***.

Пример. Представить в виде предикатов следующие предложения:

1) «Студенты сдают сессию».

2) «Число $x + 1$ больше числа x».

3) «Брат Марины».

Решение. 1) Предложение «Студенты сдают сессию» может принимать значение «Истина» или «Ложь», поэтому его можно представить в виде предиката. Во внутренней структуре данного предложения можно выделить сказуемое «сдают», подлежащее «студенты» и дополнение «сессию». Последние можно рассматривать как предметные константы. Таким образом, получаем двухместный предикат СДАВАТЬ(студенты, сессию).

2) Сказуемым в данном предложении является слово «больше». Представим подлежащее «$x + 1$» и дополнение «x» в виде термов. Причем терм «$x + 1$» имеет внутреннюю структуру, поскольку его можно представить с помощью функционального символа плюс(x, 1). Тогда исходное предложение примет вид двухместного предиката: БОЛЬШЕ(плюс(x, 1), x). Здесь x — предметная переменная, а 1 — константа.

3) Предложение «брат Марины» нельзя представить в виде предиката, поскольку его значением является не «Истина» или «Ложь», а некоторый элемент предметной области, которая соответствует множеству людей.

Пример. Перевести на естественный язык следующие высказывания логики предикатов:

1) *РАВНЯТЬСЯ(x, 5)*.

2) *ЗНАТЬ(папа (Вася), математика)*.

Решение. 1) Предикат РАВНЯТЬСЯ(x, 5) соответствует утверждению «x равняется 5» естественного языка. Здесь 5 — константа, x — предметная переменная.

2) В высказывании ЗНАТЬ(папа(Вася), математика) функциональный символ «папа(x)» принимает значение из множества

людей, соответствующее отношению «быть отцом x». Поэтому выражение папа(Вася) следует интерпретировать как «Васин папа». Таким образом, предикат ЗНАТЬ(папа(Вася), математика) соответствует предложению «папа у Васи знает математику» естественного языка. Здесь «Вася» и «математика» являются константами, а x — предметная переменная.

Вопросы

1. Дайте определение понятию предикат.
2. Что называется порядком предиката?
3. Приведите примеры n-местных предикатов.
4. Перечислите способы определения предикатов.
5. Приведите примеры функциональных символов.
6. Какой функциональный символ называют n-местным?
7. Дайте определение терма.
8. Что понимают под предметной областью?
9. Дайте определения понятий предметная переменная и предметная константа. Приведите примеры.

Задания

1. Из приведенных ниже предложений выпишите отдельно высказывания, отдельно предикаты:

   а) $x > 0$; в) о нем что-то говорят;

   б) $2 + 3 = 6$; г) x брат y;

   д) противоположные стороны A и B параллелограмма равны;

   е) каждое явление x имеет свою причину y.

2. В приведенных одноместных предикатах произведите возможные подстановки переменной x так, чтобы получить истинные высказывания. Какие из них допускают одну, а какие — многие подстановки?

   а) x — самая высокая горная вершина в мире;

   б) x — представитель диалектической логики;

   в) $x + 7 = 15$; г) x — логическая связка;

   д) x — выдающийся античный логик.

3. Переменные функции «$x > y$» принимают значения на множестве $\{1, 2, 3\}$; B_1, B_2 — предикаты, задаваемые этой функцией соответственно при алфавитном и обратном ему порядках. Установите:

   а) область определения предикатов B_1 и B_2;

   б) значения истинности $B_1(2, 3)$ и $B_2(2, 3)$.

4. Сколько различных предикатов определяет высказывание «$x + y = z$», если M_x, M_y и M_z — множества значений переменных x, y, z:
   а) $M_x = M_y = M_z = \{1, 2\}$;
   б) $M_x = \{1\}$, $M_y = \{1, 2\}$, $M_z = \{2, 3\}$?
5. Определите, эквивалентны ли следующие предикаты:
   а) $x^2 = 1$ и $x = 1$; б) $x^2 = x$ и $x = 1$.

5.6. Кванторы

Квантор всеобщности, квантор существования, связанная и свободная переменная, уменьшение порядка n-местных предикатов

При определении истинностного значения предиката особый интерес представляет вопрос: является ли он истинным при любом значении предметной переменной или существует ли хотя бы одно значение переменной, при котором данный предикат истинен. Например, утверждение «*Все простые числа имеют два делителя*» можно формализовать с помощью предиката ИМЕТЬ_ДВА_ДЕЛИТЕЛЯ(x), который является истинным для всех x в предметной области простых чисел. Утверждение «*Существуют натуральные числа, которые не делятся на 2*» означает, что предикат ДЕЛИТЬСЯ_НА_2(x) истинен не для всех x в предметной области натуральных чисел.

Определение
Пусть $P(x)$ — предикат, определенный на M. Высказывание «для всех $x \in M$, $P(x)$ истинно» обозначается $\forall x\, P(x)$. Знак $\forall$ называется ***квантором всеобщности***.

Кванторы управляют областью значения переменной, следующей за символом квантора. Если применяется квантор всеобщности, то мы говорим, что высказывание истинно для всех x из некоторого множества.

Определение
Высказывание «существует такой $x \in M$, что $P(x)$ истинно» обозначается $\exists x\, P(x)$, где знак $\exists$ называется ***квантором существования***.

Квантор существования применяется, когда нужно указать, что существует хотя бы одно значение переменной, для которого истинно данное высказывание.

В логике первого порядка существует следующее ограничение: нельзя применять кванторы к предикатам. Например, нельзя

записать $\forall P\ P(x)$. Однако такие операции осуществимы в логиках более высоких порядков.

Определение
Переход от $P(x)$ к $\forall x\ P(x)$ или $\exists x\ P(x)$ называется ***связыванием*** переменной x, а сама переменная x в этом случае — ***связанной***.

Определение
Переменная, не связанная никаким квантором, называется ***свободной***.

От того, является ли переменная связанной или свободной, зависит значение предиката. Свободная переменная — это предметная переменная, которая может принимать различные значения из множества M, и значение предиката $P(x)$ зависит от значения переменной x. Напротив, выражение $\forall x\ P(x)$ не зависит от переменной x и при заданных P и M имеет определенное значение; здесь x — связанная переменная.

Связанные переменные встречаются не только в логике. Например, в выражениях $\sum_{x=1}^{10} f(x)$ или $\int_a^b f(x)dx$ переменная x связана и данные выражения при фиксированных a, b и f имеют определенные значения, не зависящих от какого-либо значения x.

Пример. Записать в виде предикатов с кванторами следующие высказывания: «*Все студенты сдают экзамены*», «*Некоторые студенты сдают экзамены на отлично*».

Решение. Введем предикаты: P — «сдавать экзамены» и Q — «сдавать экзамены на отлично». Предметная область данных предикатов представляет собой множество студентов. Тогда исходные выражения примут вид:

$$\forall(x)\ P(x) \quad \text{и} \quad \exists(x)\ Q(x).$$

Пример. Рассматривая в качестве предметной области множество действительных чисел, записать в виде выражения логики предикатов математические утверждения:

«*Для всех x верно, что* $(x-1)^2 = x^2 - 2x + 1$»;

«*Существует число, квадрат которого равен* 4».

Решение. Введем предикат РАВНО(x, y), который истинен в том случае, если значение переменной x равно значению y. Тогда, используя кванторы, можно записать:

$\forall x$ РАВНО$((x-1)^2, (x^2-2x+1))$;

$\exists x$ РАВНО$(x^2, 4)$.

Применение кванторов к многоместным предикатам уменьшает количество свободных переменных, от которых зависит данный предикат. Пусть $A(x, y)$ — некоторый двуместный предикат, определенный на произвольном множестве M. Квантор всеобщности и квантор существования можно применять к нему как для переменной x, так и для переменной y:

$$\forall x\, A(x, y); \quad \forall y\, A(x, y); \quad \exists x\, A(x, y); \quad \exists y\, A(x, y).$$

Все четыре приведенные выражения являются записями одноместных предикатов от соответствующей свободной переменной. Так, $\forall x\, A(x, y)$ — одноместный предикат от переменной y: $\forall x\, A(x, y) = F(y)$. Предикат F истинен в точности для таких элементов $b \in M$, для которых предикат $A(x, b)$ истинен на всех значениях аргумента x. Если представить множество значений истинности предиката $A(x, y)$ в виде матрицы, то предикат $F(y) = \forall x\, A(x, y)$ истинен для таких $y = b$, для которых столбец аргумента $y = b$ содержит исключительно букву *И*. Для иллюстрации ниже приведена матрица предиката (таблица 5.6), определенного на множестве M из пяти элементов a_1, a_2, a_3, a_4, a_5, и матрицы одноместных предикатов (таблицы 5.7–5.10), полученных из исходного посредством применения кванторов.

Подобным образом $\forall y\, A(x, y) = C(x)$ — одноместный предикат от x, который истинен для таких $a \in M$, для которых строка аргумента $x = a$ содержит только букву *И*. Предикат $\exists x\, A(x, y) = H(y)$ истинен для таких $a \in M$, для которых соответствующий столбец $y = b$ содержит, по крайней мере, один раз букву *И*. Наконец, предикат $\exists y\, A(x, y)$ истинен для таких $x = a \in M$, для которых в соответствующей строке $x = a$ встречается буква *И*.

Таблица 5.6. Предикат $A(x, y)$

X \ Y	a_1	a_2	a_3	a_4	a_5
a_1	*И*	*И*	*Л*	*И*	*Л*
a_2	*Л*	*И*	*Л*	*И*	*Л*
a_3	*И*	*И*	*Л*	*И*	*И*
a_4	*Л*	*И*	*Л*	*И*	*И*
a_5	*И*	*И*	*Л*	*И*	*И*

Таблица 5.7. Предикат $\forall x\, A(x, y)$

y	$\forall x\, A(x, y)$
a_1	*Л*
a_2	*И*
a_3	*Л*
a_4	*И*
a_5	*Л*

Таблица 5.8.
Предикат $\exists x\, A(x, y)$

y	$\exists x\, A(x, y)$
a_1	*И*
a_2	*И*
a_3	*Л*
a_4	*И*
a_5	*И*

Таблица 5.9.
Предикат $\forall y\, A(x, y)$

x	$\forall y\, A(x, y)$
a_1	*Л*
a_2	*Л*
a_3	*Л*
a_4	*Л*
a_5	*Л*

Таблица 5.10.
Предикат $\exists y\, A(x, y)$

x	$\exists y\, A(x, y)$
a_1	*И*
a_2	*И*
a_3	*И*
a_4	*И*
a_5	*И*

Таким образом, применение квантора по одной из переменных двухместного предиката превращает его в одноместный. В случае трехместных предикатов применение квантора приводит к двухместному предикату. Аналогично, для n-местных предикатов применение квантора по любой переменной превращает предикат в $(n - 1)$-местный, т.е. уменьшает его порядок на единицу.

Квантор общности можно истолковать как обобщение конъюнкции, а квантор существования — как обобщение дизъюнкции. В самом деле, если область определения M предиката P конечна, например, $M = \{a_1, a_2, \ldots, a_n\}$, то высказывание $\forall x\, P(x)$ эквивалентно конъюнкции $P(a_1) \wedge P(a_2) \wedge \ldots \wedge P(a_n)$, а высказывание $\exists x\, P(x)$ — дизъюнкции $P(a_1) \vee P(a_2) \vee \ldots \vee P(a_n)$.

В качестве примера рассмотрим предикат $P(x)$, который означает «x — нечетное число» и определен на области $M = \{a, b, c\}$. Высказывание $\forall x\, P(x)$ означает: «a — нечетное число, и b — нечетное число, и c — нечетное число»; а высказывание $\exists x\, P(x)$ означает то же, что и дизъюнкция «a — нечетное число, или b — нечетное число, или c — нечетное число».

Вопросы

1. Что понимают под квантором всеобщности?
2. Дайте определение понятию квантор существования.
3. Какие переменные называются связанными, а какие — свободными?
4. Поясните на примерах назначение связанных переменных.
5. К каким последствиям приводит применение квантора по одной из переменных n-местного предиката?
6. Сколькими различными способами может быть применен квантор существования к n-местному предикату?
7. Когда предикат $\forall x\, A(x, y)$ принимает значение «Истина»?

Задания

1. Запишите следующие высказывания, используя знаки кванторов:
   а) существует число x такое, что $x + 1 = 5$;
   б) каково бы ни было число y, $y + 0 = y$;
   в) любое число либо положительно, либо отрицательно, либо равно нулю.
2. Укажите свободные и связанные вхождения каждой из переменных в следующих формулах:
   а) $\forall x\, P(x, y) \wedge \forall y\, Q(y)$;
   б) $\forall x\, (P(x) \to P(y))$;
   в) $\forall x\, (P(x) \to Q(y)) \vee \exists y\, R(x, y)$;
   г) $\forall x\, [(P(x) \to Q(y)) \vee \exists y\, R(x, y)]$.
3. Пусть x и y — любые люди, $Q(x, y)$ означает «x отец y». Следующие высказывания сформулируйте естественным языком, определив их значения истинности:
   а) $\forall x\, \exists y\, Q(x, y)$; в) $\forall y\, \exists x\, Q(x, y)$; д) $\forall x\, \forall y\, Q(x, y)$;
   б) $\exists x\, \forall y\, Q(x, y)$; г) $\exists y\, \forall x\, Q(x, y)$; е) $\exists x\, \exists y\, Q(x, y)$.
4. Пусть $N(x)$ — «x — натуральное число», $C(x)$ — «x — целое число», $P(x)$ — «x — простое число», $E(x)$ — «x — четное число», $O(x)$ — «x — нечетное число», $D(x, y)$ — «y делится на x». Сформулируйте естественным языком следующие высказывания, установив их значения истинности:
   а) $P(z)$;
   б) $E(2) \wedge P(2)$;
   в) $\forall x\, (D(2, x) \to E(x))$;
   г) $\exists x\, (E(x) \wedge D(x, 6))$;
   д) $\forall x\, [P(x) \to \exists y\, (E(y) \wedge D(x, y))]$;
   е) $\forall x\, (N(x) \to C(x))$;
   ж) $\exists x\, (N(x) \to C(x))$;
   з) $\forall x\, (C(x) \to N(x))$;
   и) $\forall x\, \forall y\, [O(x) \to (P(y) \to D(x, y))]$;
   к) $\forall x\, [C(x) \to (E(x) \vee \neg E(x))]$;
   л) $\exists x\, \forall y\, [(C(x) \wedge C(y)) \to D(x, y)]$;
   м) $\forall x\, \forall y\, [(E(x) \wedge O(x)) \to \neg D(x, y)]$.
5. Предикат $P(x, y)$ задан в предметной области $D = \{a, b\}$ следующей матрицей:

x	a	a	b	b
y	a	b	a	b
$P(x, y)$	0	1	1	1

   Какая из нижеприведенных формул определяет данный предикат?
   а) $\forall x\, P(x, a)$; в) $\exists y\, \forall x\, P(x, y)$; д) $\forall y\, \forall x\, \neg P(x, y)$.
   б) $\forall y\, P(a, y)$; г) $\forall y\, \forall x\, P(x, y)$;

5.7. Формулы в логике предикатов

Элементарная формула, правильно построенные формулы, область действия квантора, интерпретация формул логики предикатов, общезначимые и противоречивые формулы, логическое следствие

Используя понятия предиката, квантора и терма, можно определить понятие формулы в логике предикатов.

Определение
Если P — n-местный предикат и $t_1, ..., t_n$ — термы, то $P(t_1, ..., t_n)$ называется ***атомом*** или ***элементарной формулой*** логики предикатов. Например: ДЕЛИТСЯ(x, 13), ДЕЛИТСЯ(x, y), БОЛЬШЕ(плюс(x, 1), x), РАВНЯТЬСЯ(x, 1), СДАВАТЬ(студенты, сессии).

Определение
Правильно построенными формулами логики первого порядка называются формулы, которые можно рекурсивно определить следующим образом:
1. Атом является формулой.
2. Если F и G — формулы, то $(\neg F)$, $(F \vee G)$, $(F \wedge G)$, $(F \to G)$, $(F \sim G)$ также являются формулами.
3. Если F — формула, а x — свободная переменная, то $(\forall x)F$ и $(\exists x)F$ тоже формулы.
4. Никаких формул, кроме порожденных указанными выше правилами, не существует.

Определение
Часть формулы, на которую распространяется действие квантора, называется ***областью действия квантора***.

Поскольку действие квантора может распространяться не на всю формулу, а только на ее часть, то переменная может быть связанной в одной части формулы и свободной в другой. В этом случае полагают, что переменная является и связанной, и свободной одновременно.

Пример. Определить, какие переменные являются связанными, а какие — свободными в следующих формулах:

1. $A(x, y)$.
2. $\exists y\,(B(x) \to \forall x\, A(x, y))$.
3. $\exists x\,(B(x) \to \forall x\, A(x, y))$.

Решение. Обе переменные в формуле 1 являются свободными. В формуле 2 переменная y является связанной, а переменная x — и связанной, и свободной (переменная x свободна в предикате $B(x)$ и связана в предикате $A(x, y)$). В формуле 3 переменная x является связанной, а переменная y — свободной.

В логике высказываний интерпретация формулы заключается в приписывании атомам истинностных значений. В логике предикатов понятие интерпретации формулы несколько расширяется: необходимо указать предметную область (область значений предметных переменных) и значения констант, а также функциональных символов и предикатов, встречающихся в формуле.

Определение

Интерпретация формулы F логики предикатов состоит из элементов непустой предметной области D, значений всех констант, функциональных символов и предикатов, встречающихся в F. Указанные значения задаются следующим образом:

1. Каждой константе ставится в соответствие некоторый элемент из D.
2. Каждому n-местному функциональному символу ставится в соответствие отображение из D^n в D. Здесь $D^n = (x_1, x_2, \ldots, x_n)$, где $x_1, \ldots, x_n \in D$.
3. Каждому n-местному предикату ставится в соответствие отображение из D^n в $\{И, Л\}$.

Для каждой интерпретации на области D формула может получить истинностное значение $И$ или $Л$ согласно следующим правилам:

1. Если заданы значения формул F и G, то истинностные значения формул $(\neg F)$, $(F \vee G)$, $(F \wedge G)$, $(F \to G)$, $(F \sim G)$ получаются с помощью таблиц истинности соответствующих логических операций.
2. Формула $(\forall x)F$ получает значение $И$, если F получает значение $И$ для каждого x из D, в противном случае она получает значение $Л$.
3. Формула $(\exists x)F$ получает значение $И$, если F получает значение $И$ хотя бы для одного x из D, в противном случае она получает значение $Л$.
4. Формула, содержащая свободные переменные, не может получить истинностное значение.

После уточнения понятия интерпретации в логике предикатов такие понятия, как ***общезначимость***, ***противоречивость***, ***выполнимость***, ***нейтральность*** (***необщезначимость***) ***формулы*** и ***логическое следствие*** могут быть определены в точности как для формул логики высказываний (см. п. 5.2, 5.3).

Вопросы

1. Дайте определение атома в логике первого порядка.
2. Что называется формулой в логике первого порядка?
3. Сформулируйте понятие интерпретации формулы логики первого порядка.
4. Что понимают под областью действия квантора?
5. Сравните понятие интерпретации в логике высказываний и логике предикатов.
6. Перечислите виды формул логики первого порядка в зависимости от принимаемых ими истинностных значений.
7. Запишите правила, согласно которым формула логики первого порядка может получить истинностное значение.

Задания

1. Пусть предикат $M(x, y)$ означает «x меньше y», предикат $D(x, y)$ — «x равно y», а предикат $S(x, y, z)$ — «$x = y + z$». Следующие высказывания сформулируйте естественным языком:
   а) $(\forall x)\, M(0, x)$;
   б) $(\forall x)\, S(x, x, x)$;
   в) $(\exists x)\, M(x, x - 1)$;
   г) $\neg((\exists x)\, S(x, x, x))$;
   д) $(\exists y)(\exists x)\, S(x, y, z)$;
   е) $(\exists y)(\forall x)\, (D(x, y) \vee M(x, y))$;
   ж) $(\forall x)(\forall y)\, (S(x, y, y) \to M(y, x))$.
2. Пусть X — множество сотрудников отдела, а $P(x)$, $Q(x)$ и $R(x)$ — предикаты, означающие соответственно: x занимается спортом, x изучает иностранные языки, x имеет изобретения, $x \in X$. Расшифруйте:
   а) $(\exists x)\, P(x)Q(x)$; б) $(\forall x)\, P(x)(Q(x) \to R(x))$.
3. Указать свободные и связанные вхождения переменных в следующие формулы, содержащие предикаты A и B:
   а) $(\forall x_2)\, A(x_1, x_2)$;
   б) $(\forall x_3)((\forall x_1)A(x_1, x_2) \leftarrow A(x_3, x_1))$;
   в) $(\forall x_2)\, A(x_3, x_2) \to \forall x_3\, B(x_3, x_2)$;
   г) $(\forall x_2)\, A(x_1, x_2) \to \exists x_1\, A(x_1, x_2)$;
   д) $(\exists x_2)(\forall x_1)\, A(x_1, x_2) \to B(x_1)$.
4. Задана предметная область $D = \{1, 2\}$, значения констант a и b, функциональных символов f, а также предиката P:

a	b
1	2

$f(1)$	$f(2)$
2	1

$P(1,1)$	$P(1,2)$	$P(2,1)$	$P(2,2)$
1	1	1	0

Найти истинностные значения следующих предикатных выражений:

а) $P(a, f(a)) \wedge P(b, f(b))$;

б) $(\forall x)(\forall y)(P(x, y) \to P(f(x), f(y)))$.

5. Оценить формулу $(\forall x)(P(x)) \to Q(f(x), a)$ на интерпретации

$$d = \{1, 2\}, \quad a = 1, \quad f(1) = 2, \quad f(2) = 1.$$

$P(1)$	$P(2)$	$Q(1, 1)$	$Q(1, 2)$	$Q(2, 1)$	$Q(2, 2)$
0	1	1	1	0	1

5.8. Законы и тождества в логике первого порядка

Коллизия переменных, замена связанной переменной, коммутативные и дистрибутивные свойства кванторов, закон де Моргана для кванторов

Все законы и тождества, которые справедливы в логике высказываний, остаются справедливыми и в логике предикатов. Кроме того, в логике предикатов существуют дополнительные законы, предназначенные для эквивалентного преобразования формул, содержащих кванторы и переменные.

Следует отметить, что перенос квантора в начало формулы может изменить смысл предикатного высказывания. Так, например, $\forall x\, F(x) \wedge \forall x\, G(x)$ не эквивалентно $\forall x\, (F(x) \wedge G(x))$. В общем случае следует переименовать связанные переменные во избежание *коллизии* — ситуации, когда в формуле одна и та же переменная находится в области действия противоположных кванторов. Например, в формуле $\forall x\, (F(x)) \to \exists x\, (Q(x))$ переменная x одновременно находится в области действия кванторов $\forall$ и $\exists$.

Рассмотрим высказывание, использующее предикат равенства РАВНО двух чисел: $\forall x\ \exists y$ РАВНО$(x + 1, y)$. Данное высказывание означает, что для любого числа x существует число y, которое больше его на единицу. Приведенное высказывание является истинным. Однако, если поменять порядок расположения кванторов на противоположный, то получим следующее высказывание: $\exists y\ \forall x$ РАВНО$(x + 1, y)$. Полученное высказывание означает, что существует такое число y (одно!), которое на единицу больше любого числа x. Данное высказывание не соответствует предыдущему и является ложным.

Для эквивалентных преобразований предикатных высказываний с кванторами необходимо использовать приведенные ниже законы. Прежде чем непосредственно перейти к рассмотрению законов действий с кванторами, введем следующие обозначения: $F(x)$ и $H(x)$ — одноместные предикаты, $P(x, y)$ — двухместный предикат.

1. ***Замена связанной переменной***:

$$(\exists x)\ F(x) = (\exists y)\ F(y);$$

$$(\forall x)\ \mathrm{F}(x) = (\forall y)\ F(y).$$

Введение нового обозначения связанной переменной (т.е. переименование связанной переменной) не изменяет смысл формулы логики предикатов, если выполняется следующее условие: никакая свободная переменная в любой части формулы не должна после переименования оказаться связанной. Другими словами, для нового обозначения связанной переменной следует выбирать букву, которая отсутствует в формуле. Например:

$$(\forall x)\ (\exists y)\ P(x, y) = (\forall x)\ (\exists z)\ P(x, z).$$

В данном примере осуществлена операция переименования связанной переменной y.

2. ***Коммутативные свойства кванторов***:

$$(\forall x)\ (\forall y)\ P(x, y) = (\forall y)\ (\forall x)\ P(x, y);$$

$$(\exists x)\ (\exists y)\ P(x, y) = (\exists y)\ (\exists x)\ P(x, y).$$

Менять местами можно только одноименные кванторы.

$$(\forall x)\ (\exists y)\ P(x, y) \neq (\exists y)\ (\forall x)\ P(x, y).$$

3. ***Дистрибутивные свойства кванторов***:

$$(\forall x)F(x) \vee G = (\forall x)(F(x) \vee G);$$

$$(\exists x)F(x) \vee G = (\exists x)(F(x) \vee G);$$

$$(\forall x)F(x) \wedge G = (\forall x)(F(x) \wedge G);$$

$$(\exists x)F(x) \wedge G = (\exists x)(F(x) \wedge G),$$

где G — формула логики предикатов, которая не содержит x;

$$(\forall x)F(x) \wedge (\forall x)H(x) = (\forall x)(F(x) \wedge H(x));$$

$$(\exists x)F(x) \vee (\exists x)H(x) = (\exists x)(F(x) \vee H(x)).$$

Сформулированный дистрибутивный закон справедлив только для квантора всеобщности $\forall$ при конъюнкции $\wedge$ и квантора

существования $\exists$ при дизъюнкции $\vee$, так как другие комбинации приводят к неравенствам:

$$(\forall x)F(x) \vee (\forall x)H(x) \neq (\forall x)(F(x) \vee H(x));$$
$$(\exists x)F(x) \wedge (\exists x)H(x) \neq (\exists x)(F(x) \wedge H(x)).$$

Для преодоления данного ограничения дистрибутивного закона, следует использовать замену связанной переменной:

$$(\forall x)F(x) \vee (\forall x)H(x) = (\forall x)F(x) \vee (\forall y)H(y) =$$
$$= (\forall x)(\forall y)\ (F(x) \vee H(y));$$
$$(\exists x)\mathrm{F}(x) \wedge (\exists x)H(x) = (\exists x)\mathrm{F}(x) \wedge (\exists y)H(y) =$$
$$= (\exists x)(\exists y)\ (F(x) \wedge H(y)).$$

Таким образом, в общем случае дистрибутивные свойства кванторов можно записать следующей схемой:

$$(Q_1x)F(x) \vee (Q_2y)\ H(y) = (Q_1x)(Q_2y)\ (F(x) \vee H(y));$$
$$(Q_1x)F(x) \wedge (Q_2y)\ H(y) = (Q_1x)(Q_2y)\ (F(x) \wedge H(y)),$$

где Q_1, Q_2 — любой из кванторов $\exists$ или $\forall$.

4. ***Закон де Моргана для кванторов***:

$$\neg((\forall x)F(x)) = (\exists x)\neg F(x);$$
$$\neg((\exists x)F(x)) = (\forall x)\neg F(x).$$

Проиллюстрируем данный закон следующим примером. Пусть предикат $F(x)$ означает, что «x является простым числом». Когда x последовательно принимает значения ряда натуральных чисел — $x = \{1, 2, 3, 4, 5, 6, 7, ...\}$, предикат соответственно меняет истинностное значение:

$$F(1) = Л,\ F(2) = И,\ F(3) = И;$$
$$F(4) = Л,\ F(5) = И,\ F(6) = Л,\ F(7) = И,\ ....$$

Убедимся в справедливости первой формулы для отрицания квантора всеобщности:

$\neg((\forall x)F(x))$ = «не все x являются простыми числами» = = «существуют такие x, которые являются непростыми числами» = = $(\exists x)\neg F(x) = И$.

Оба приведенные высказывания истинны. Теперь убедимся в справедливости второй формулы для отрицания квантора существования:

$\neg((\exists x)F(x))$ = «нет ни одного x, которое было бы простым» = = «все x являются непростыми числами» = $(\forall x)\neg F(x) = Л$.

Данные высказывания являются ложными.

Вопросы

1. Что понимают под коллизией переменных?
2. К каким последствиям может привести перенос квантора в начало формулы? Приведите примеры корректного и некорректного переноса кванторов в начало формулы.
3. Объясните суть замены связанной переменной.
4. Сформулируйте коммутативные свойства кванторов.
5. При соблюдении какого условия правомерно использование дистрибутивных свойств кванторов?
6. Каким образом можно обойти ограничения дистрибутивных свойств кванторов?
7. Запишите формулы закона де Моргана для кванторов.

Задания

1. Опустить знаки отрицания непосредственно на предикаты или булевы переменные:
   а) $\neg(\exists x)\,((\neg(\forall y)\,(B(y) \vee (\exists z)C(z)) \wedge \neg A(x)))$;
   б) $\neg(\forall y)\,(\neg(\exists x)\,(A(y, z) \vee B(x)) \vee C(z))$;
   в) $\neg(\exists x)\,((\neg(\forall y)\,(A(x) \to B)) \vee C(x, y))$;
   г) $\neg(\forall u)\,\neg(\forall v)(P(u) \to Q(v) \to A(z))$.
2. Установить, эквивалентны ли заданные предикаты:
   а) $(\exists x)(A(x) \wedge \neg B(y))$ и $\neg(\forall z)(A(z) \to B(y))$;
   б) $(\forall x)((A(x) \to B(x)) \wedge (A(x) \to \neg B(x)))$ и $\neg(\exists y)\,\mathrm{A}(y)$.
3. Вынести за скобки кванторы:
   а) $(\exists v)\,C(v, y) \wedge ((\exists x)\,A(x) \vee B)$;
   б) $(\exists x)(\exists y)\,A(x, y) \wedge (\exists x)(\exists y)\,B(x, y)$;
   в) $(\exists x)\,A(x, y) \vee ((\forall x)\,B(x) \vee (\forall y)\,C(y))$;
   г) $(\exists x)(\forall y)A(x, y) \wedge (\exists x)(\exists z)\,(B(x, z) \wedge (\exists y)\,A(x, y))$.
4. Доказать общезначимость следующих формул:
   а) $(\forall x)P(x) \to (\exists y)P(y)$;
   б) $(\forall x)P(x) \vee ((\exists y)\neg P(y))$.
5. Доказать, что формула $(\forall x)P(x) \wedge (\exists y)\neg P(y)$ противоречива.
6. Доказать, что формула $P(a) \to \neg((\exists x)P(x))$ непротиворечива.

5.9. Предваренные нормальные формы и логический вывод в логике предикатов

Предваренная нормальная форма, алгоритм приведения к предваренной нормальной форме, правила удаления/введения квантора всеобщности/существования

В логике высказываний были введены две нормальные формы: конъюнктивная и дизъюнктивная. В логике первого порядка

вводится третья нормальная форма, называемая предваренной нормальной формой.

Определение

Формула F в логике первого порядка находится в ***предваренной нормальной форме*** (***ПНФ***) тогда и только тогда, когда она может быть представлена в виде $(Q_1x_1)\ \ldots\ (Q_nx_n)(M)$, где каждое (Q_ix_i), $i = 1, \ldots, n$, есть или $(\forall x)$, или $(\exists x)$, а M — формула, не содержащая кванторов. Причем $(Q_1x_1)\ \ldots\ (Q_nx_n)$ называется ***префиксом***, а M — ***матрицей формулы*** F.

Для преобразования выражений произвольной формы в ПНФ необходимо поочередно выполнить следующие этапы преобразования.

1. Исключить логические связки эквиваленции (~) и импликации ($\rightarrow$), выразив их через операции дизъюнкции, конъюнкции и отрицания с помощью следующих законов:

$$F \rightarrow G = \neg F \vee G; \quad F \sim G = (\neg F \vee G) \wedge (\neg G \vee F).$$

2. Опустить знаки операций отрицания непосредственно на предикаты, используя закон двойного отрицания $\neg(\neg F) = F$ и законы де Моргана $\neg(F \vee G) = \neg F \wedge \neg G$, $\neg(F \wedge G) = \neg F \vee \neg G$, в том числе для кванторов:

$$\neg((\forall x)\ F(x)) = (\exists x)\ (\neg F(x)); \quad \neg((\exists x)\ F(x)) = (\forall x)\ (\neg F(x)).$$

3. Если необходимо, переименовать связанные переменные.

4. Вынести кванторы в начало формулы, используя соответствующие законы, для получения предваренной нормальной формы.

Пример. Привести формулу $(\forall x)F(x) \rightarrow (\exists x)Q(x)$ к ПНФ.

Решение. Сначала исключим импликацию, затем опустим знак операции отрицания непосредственно на предикат и вынесем квантор в начало:

$$(\forall x)F(x) \rightarrow (\exists x)H(x) = \neg((\forall x)F(x)) \vee (\exists x)H(x) =$$
$$= (\exists x)(\neg F(x)) \vee (\exists x)H(x) = (\exists x)(\neg F(x) \vee H(x)).$$

Пример. Получить ПНФ для формулы

$$G \equiv (\forall x)(\forall y)((\forall z)P(x, y) \wedge P(y, z) \rightarrow (\exists z)R(x, y, z)).$$

Решение. Воспользуемся приведенным выше алгоритмом.

$$G = (\forall x)(\forall y)(\neg((\forall z)P(x, z) \wedge P(y, z)) \vee (\exists z)\, R(x, y, z)) =$$
$$= (\forall x)(\forall y)((\exists z)(\neg P(x, y) \vee \neg P(y, z)) \vee (\exists u)\, R(x, y, u)) =$$
$$= (\forall x)(\forall y)(\exists z)(\exists u)(\neg P(x, y) \vee \neg P(y, z) \vee R(x, y, u)).$$

Рассмотрим правила вывода, которые можно использовать для проведения дедуктивных умозаключений с высказываниями логики предикатов, содержащими кванторы. Эти правила часто используются в ходе математических доказательств без дополнительного объяснения.

Правило удаления квантора всеобщности

$$\frac{\forall x\, F(x)}{F(c) \text{ для произвольного } c \in D}$$

используется для доказательства истинности $F(c)$, где c — произвольно выбранный элемент предметной области D, в которой справедливо $\forall x\, F(x)$. Например, из посылки «*Все студенты любят получать хорошие оценки*» делаем вывод: «*Студент Петров любит получать хорошие оценки*».

Правило введения квантора всеобщности

$$\frac{F(c) \text{ для произвольного } c \in D}{\forall x\, F(x)}$$

утверждает истинность $\forall x\, F(x)$, если доказана истинность $F(c)$ для любого c, то есть для всех элементов c из рассматриваемой предметной области D.

Правило удаления квантора существования в истинной формуле $\exists x\, F(x)$ заключается в указании имени элемента c (конкретного или гипотетического), для которого $F(c)$ истинно:

$$\frac{\exists x\, F(x)}{F(c) \text{ для некоторого } c \in D}.$$

Правило введения квантора существования

$$\frac{F(c) \text{ для некоторого } c \in D}{\exists x\, F(x)}$$

позволяет заключить, что $\exists x\, F(x)$ является истинным, когда известен некоторый элемент c, для которого истинно $F(c)$.

Кроме перечисленных правил, в логике предикатов в ходе дедуктивного вывода можно использовать все правила, которые применяются для дедуктивных выводов в логике высказываний.

Пример. Показать, что из утверждений «*Все в первой группе изучают математику*» и «*Маша — студентка первой группы*» следует вывод: «*Маша изучает математику*».

Решение. Обозначим через $F(x)$ предикат «x есть студент первой группы», а через $M(x)$ — «x изучает математику». Тогда посылки можно записать в виде: $\forall x\,(F(x) \to M(x))$ и F(Маша) соответственно, а требуемый вывод — M(Маша). Для получения данного вывода необходимо осуществить следующую последовательность действий:

1. $\forall x\,(F(x) \to M(x))$ — первая посылка.
2. F(Маша) $\to M$(Маша) — шаг 1, правило удаления квантора $\forall$.
3. F(Маша) — вторая посылка.
4. M(Маша) — шаги 2 и 3, правило отделения.

Таким образом, получен искомый результат.

Вопросы

1. Дайте определение предваренной нормальной формы.
2. С помощью каких законов можно опустить знаки операций отрицания непосредственно на предикаты?
3. Сформулируйте алгоритм преобразования выражения произвольной формы в ПНФ.
4. Перечислите правила вывода, которые можно использовать для проведения дедуктивных умозаключений с высказываниями логики предикатов.
5. В чем заключается правило удаления квантора всеобщности?
6. Как и зачем используется правило введения квантора всеобщности?
7. Объясните отличие в трактовке элемента предметной области в правиле удаления квантора существования от принятой в правиле введения квантора всеобщности.
8. Каково правило введения квантора существования?

Задания

1. Привести к ПНФ следующие выражения:
   а) $(\forall y)\,(F_1(y) \vee \neg(\exists x)\,P(x, y))$;
   б) $\neg(\exists x)\,((\forall y)\,A(x, y) \wedge (\exists y)\,(\forall z)\,(C(z) \to B(x, y))$;
   в) $(\exists x)\,(\forall y)\,B(x, y) \sim (\exists x)\,A(x)$;
   г) $\neg(\forall y)\,(\exists x)\,(A(x) \to B(y))$;
   д) $(\forall x)\,F_1(x) \to \neg(\forall x)\,(F_2(y) \vee (\forall y)\,P(x, y))$.
2. Определите, является ли формула $(\exists x)\,(P(x) \wedge Q(x))$ логическим следствием формул $(\exists x)\,P(x)$ и $(\exists x)\,Q(x)$.
3. Применяя дедуктивные правила логики предикатов, выведите заключение из следующих посылок:

   3.1. Если кто-то из тех людей — автор этого слуха, то он глуп и беспринципен. Но никто из тех людей не глуп и не лишен принципов.

3.2. Если все эти люди не храбры или на них нельзя положиться, то они не принадлежат к нашей компании. Но они принадлежат к нашей компании.

3.3. Если кто-то из подозреваемых совершил все эти нераскрытые кражи, то он был тщательно подготовлен и имел сообщника. Если бы все кражи были подготовлены тщательно, то, если бы был сообщник, украдено было бы гораздо больше. Но последнее не имеет места.

3.4. Если один из нас пойдет завтра на первое занятие, то он должен будет встать рано, а если мы пойдем сегодня вечером в кино, то он ляжет поздно спать. Если любой из нас ляжет поздно спать, а встанет рано, то будет довольствоваться пятью часами сна. Но мы не можем довольствоваться пятью часами сна.

3.5. В бюджете возникает дефицит, если и только если не повысят некоторые пошлины. Государственные расходы на все социальные нужды сократятся, если и только если в бюджете будет дефицит. Некоторые пошлины повысят.

3.6. Если все цены одновременно повышаются, то повышается и заработная плата. Все цены высоки или применяется регулирование цен. Если применяется регулирование цен, то нет инфляции. Наблюдается инфляция.

5.10. Исчисление предикатов

Структура исчисления предикатов, правила отделения и обобщения, правила $\forall$- и $\exists$-введения, переименование свободных и связанных переменных

Аналогично исчислению высказываний, в логике предикатов существует формальная система — ***исчисление предикатов***, которая занимается конструированием формул и доказательством их общезначимости. Исчисление предикатов имеет идентичную исчислению высказываний структуру, а именно: язык, систему аксиом и правила вывода.

Аксиомы исчисления предикатов можно разделить на две группы:

1) Аксиомы исчисления высказываний (можно выбрать любую из формальных систем S_1, S_2).
2) Предикатные аксиомы, где переменная x в формуле $F(x)$ является свободной и ни разу не подвергается действию квантора по y:

P1) $\forall x\, F(x) \to F(y)$;

P2) $F(y) \to \exists x\, F(x)$.

Формула $F(y)$ получена из $F(x)$ заменой x на y.

Для уяснения смысла требования к вхождениям x в $F(x)$ рассмотрим в качестве $F(x)$ формулу $\exists y\, P(y, x)$, в которой свободное вхождение x находиться в области действия квантора $\exists y$, т.е. указанное требование не удовлетворяется. Подстановка данной формулы в аксиому P1 дает следующую формулу: $\forall x\, \exists y\, P(y, x) \to \to \exists y\, P(y, y)$.

Если полученную формулу проинтерпретировать на множестве натуральных чисел N с предикатом P «быть больше», то получим высказывание: «если для всякого x найдется y, который больше его, то найдется и y, больший самого себя». Посылка этой импликации истинна на N, а ее заключение ложно, поэтому все высказывание является ложным.

В исчислении предикатов используются следующие правила вывода:

1) Правило отделения (Modus Ponens), сформулированное в п. 5.3 (см. таблицу 5.5), полностью переносится из исчисления высказываний.

2) Правило обобщения ($\forall$-введения):

$$\frac{F \to G(x)}{F \to \forall x G(x)},$$

где $G(x)$ содержит свободные вхождения x, а F их не содержит.

3) Правило $\exists$-введения:

$$\frac{G(x) \to F}{\exists x G(x) \to F}$$

при тех же требованиях к F и G, что и в предыдущем правиле.

Правило переименования свободных переменнных

В исчислении предикатов из выводимости формулы $F(x)$, содержащей свободные вхождения x, ни одно из которых не находится в области действия квантора по y, следует выводимость $F(y)$.

Пример. Доказать справедливость правила переименования свободных переменных.

Решение.

1. $F(x)$ (по условию).

2. $F(x) \to (G \to F(x))$ (аксиома 1 формальной системы S_2; здесь в качестве G можно выбрать любую доказуемую формулу, не содержащую свободных вхождений переменных x; доказуемость данной формулы понадобится на шаге 5, а ограничение на x — на шаге 4).

3. $G \to F(x)$ (по правилу отделения, шаги 1, 2).

4. $G \to \forall x\, F(x)$ (по правилу обобщения, шаг 3).

5. $\forall x\, F(x)$ (следствие из шага 4, т. к. G — истинная формула).

6. $F(y)$ (шаг 5, аксиома P1)

Правило доказано.

Правило переименования связанных переменных

В исчислении предикатов из выводимости $\forall x\, F(x)$ следует выводимость $\forall y\, F(y)$, а из выводимости $\exists x\, F(x)$ — выводимость $\exists y\, F(y)$ при условии, что $F(x)$ не содержит свободных вхождений y и содержит свободные вхождения x, ни одно из которых не входит в область действия квантора по y.

Пример. Доказать правило переименования связанных переменных для квантора общности.

Решение.

1. $\forall x\, F(x)$ (по условию).

2. $\forall x\, F(x) \to F(y)$ (аксиома P1).

3. $\forall x\, \mathrm{F}(x) \to \forall y\, F(y)$ (правило обобщения, шаг 2).

4. $\forall y\, F(y)$ (шаги 1, 3).

Необходимо отметить, что доказательство для квантора $\exists$ осуществляется аналогично, но использует аксиому P2 и правило $\exists$-введения.

Теорема 1

Всякая доказуемая формула исчисления предикатов тождественно истинна.

Данная теорема аналогична соответствующей теореме исчисления высказываний и показывает, что правила вывода в исчислении предикатов сохраняют общезначимость, т.е. их применение к общезначимым формулам снова дает общезначимые формулы.

Вопросы

1. Сформулируйте назначение исчисления предикатов.
2. Запишите формулы аксиом исчисления предикатов.
3. Объясните ограничение аксиом исчисления предикатов.
4. Перечислите правила вывода исчисления предикатов.
5. Постройте схему правила обобщения.
6. Определите правило ∃-введения с помощью схемы.
7. Какие ограничения необходимы для правила обобщения и ∃-введения? К каким последствиям может привести несоблюдение указанных ограничений?
8. Какую особенность приобретает правило подстановки в исчислении предикатов?
9. Сформулируйте правило переименования свободных переменных.
10. В чем заключается сущность правила переименования связанных переменных?
11. Сформулируйте утверждение теоремы об общезначимости правил вывода исчисления предикатов.

Задания

1. Докажите нелогичность следующих рассуждений:

   1.1. Все студенты нашей группы — члены клуба «Динамо». **А** некоторые члены клуба «Динамо» занимаются спортом. Следовательно, некоторые студенты нашей группы занимаются спортом.

   1.2. Некоторые студенты нашей группы — болельщики «Динамо». **А** некоторые болельщики «Динамо» занимаются спортом. Следовательно, некоторые студенты нашей группы занимаются спортом.

   1.3. Каждый первокурсник знаком с кем-либо из студентов второго курса. **А** некоторые второкурсники — спортсмены. Следовательно, каждый первокурсник знаком с кем-либо из спортсменов.

2. Показать, что формула G не является логическим следствием множества формул K:

   а) $G = (\forall x)\neg R(x), \quad K = \{(\exists x)R(x) \to (\exists x)Q(x), \quad \neg Q(a)\}$;

   б) $G = (\forall x)R(x, x), \quad K = \{(\forall x)(\forall y)(R(x, y) \to R(y, x)),$
   $(\forall x)(\forall y)(\forall z)(R(x, y) \wedge R(y, z) \to R(x, z))\}$;

   в) $G = (\exists x)(P(x) \wedge \neg R(x)), \quad K = \{(\forall x)[P(x) \to (\exists y)(Q(y) \wedge S(x, y))],$
   $(\exists x)[R(x) \wedge (\forall y)(Q(y) \to \neg S(x, y))], (\exists x)P(x)\}$.

5.11. Многозначная логика

Возникновение многозначных логик, значение истинности высказывания, алфавит многозначной логики, унарные и бинарные функции, полная система функций многозначной логики

Впервые многозначная логика появилась в связи с отрицанием аристотелева закона исключенного третьего. В соответствии с этим законом дизъюнктивное высказывание $p \vee \neg p$ есть тавтология, а атомарное высказывание p в аристотелевой логике всегда либо истинно, либо ложно. Поскольку в аристотелевой логике любое высказывание может принимать только одно из двух значений истинности (истину или ложь), она получила название ***двузначной логики***. В 1921 году Я. Лукашевич в маленькой двухстраничной статье рассматривает трехзначную логику, т.е. такую логику, в которой всякое высказывание p может принимать одно из трех возможных значений истинности. Независимо от Лукашевича Э. Пост анализирует m-значную логику, в которой высказывание p может принимать одно из m возможных значений истинности, где m — любое целое число, большее 1. В случае, когда m больше 2, логику называют ***многозначной***. В 1930 г. Лукашевич и А. Тарский предпринимают дальнейшее изучение m-значной логики. В 1932 г. понятие m-значной логики обобщается Г. Рейхенбахом, рассматривающим бесконечнозначную логику, в которой для высказывания p существует бесконечное множество значений истинности.

Выдающийся ученый А. Гейтинг примерно в то же время построил двузначную символическую логику, исходя из потребностей интуиционистской математической школы. Данная логика, в отличие от аристотелевой, не принимает безоговорочно законов исключенного третьего и двойного отрицания. Вследствие этого законы созданной со специальными целями логики Гейтинга, так же как и законы многозначных логик, отличаются от законов Аристотеля. Поэтому такие логики называют неаристотелевыми. Символическая двузначная логика, построенная Гейтингом в работе «Принципы математики», принадлежит к числу неаристотелевых логик, отличаясь от аристотелевой иной интерпретацией импликации.

Подобно неэвклидовым геометриям неаристотелевы логики также нашли себе применение. Бесконечнозначная логика была задумана Г. Рейхенбахом в качестве фундамента математической

теории вероятности. А в 1933 г. Т. Швицкий обнаружил, что многозначные логики могут быть использованы в современной квантовой физике. Многие аспекты такого использования были исследованы Г. Биркгофом и Г. Рейхенбахом. Использование интуиционистами логики Гейтинга также свидетельствует о математической ценности новых логик.

Для простоты рассмотрения основных положений теории многозначных логик ограничимся трехзначной логикой и воспользуемся методом таблиц истинности. Прежде всего, воспроизведем таблицу истинности для операции конъюнкции (табица 5.11).

Таблица 5.11. Таблица истинности конъюнкции

$\wedge$		q	
		И	*Л*
p	*И*	*И*	*Л*
	Л	*Л*	*Л*

Таблица 5.11 построена следующим образом: в левом столбце приводятся возможные значения истинности для высказывания p, а в верхней строке — возможные значения истинности для высказывания q. Зная значения истинности указанных высказываний, можно найти значение их конъюнкции в ячейке, стоящей на пересечении строчки, соответствующей значению истинности p, и столбца, соответствующего значению истинности q. Поскольку, по определению, конъюнкция истинна в том и только в том случае, когда оба высказывания p и q истинны, значение *И* стоит в левой верхней ячейке таблицы, а во всех остальных — *Л*. Отметим, что таблица заполняется на основании одного лишь определения операции конъюнкции.

Теперь перейдем к трехзначной логике и обозначим три возможные значения истинности высказывания через *И*, «?» и *Л*. Снова составим таблицу истинности операции конъюнкции (таблица 5.12). Поскольку конъюнкция истинна, когда оба высказывания p и q истинны, левая верхняя ячейка таблицы должна содержать значение *И*, кроме того, *И* не может находиться ни в какой другой ячейке таблицы. Таким образом остается восемь ячеек, в каждой из которых может быть записано либо значение *Л*, либо значение «?». Всего получается $2^8 = 256$ способов заполнения таблицы. Отсюда следует, что в трехзначной логике существует 256 различных определений конъюнкции.

Таблица 5.12. Шаблон таблицы истинности конъюнкции в трехзначной логике

∧		q		
		И	?	*Л*
p	*И*	*И*		
	?			
	Л			

В таблицах 5.13 и 5.14 приведены две из 256 возможных таблиц истинности операции конъюнкции в трехзначной логике.

Таблица 5.13. Таблица истинности конъюнкции в трехзначной логике (Лукашевич и др.)

∧		q		
		И	?	*Л*
p	*И*	*И*	?	*Л*
	?	?	?	*Л*
	Л	*Л*	*Л*	*Л*

Таблица 5.14. Таблица истинности конъюнкции в трехзначной логике (Бочар)

∧		q		
		И	?	*Л*
p	*И*	*И*	?	*Л*
	?	?	?	?
	Л	*Л*	?	*Л*

Таблица истинности 5.13 была предложена Лукашевичем, Постом и Россером. Данная таблица строится на основании соглашения, по которому значение «?» более ложно, чем *И*, но менее ложно, чем *Л*, а значение конъюнкции совпадает со значением истинности более ложного из составляющих высказываний.

Отличное от описанного определение конъюнкции дает Бочар (таблица 5.14). По Бочару символ «?» означает неразрешимость, а конъюнкция высказываний p и q считается неразрешимой в случае неразрешимости хотя бы одного из составляющих высказываний.

Рассмотрим таблицу истинности операции отрицания. В случае отрицания единственное ограничение заключается в том, что $\neg p$ не может быть истинным при истинности высказывания p, и $\neg p$ не может быть ложным в случае ложности p. Указанное ограничение полностью определяет таблицу истинности для отрицания в двузначной логике и допускает 12 возможных способов определения

отрицания в трехзначной. Ниже приведены две из 12 возможных для отрицания таблиц истинности операции отрицания. Таблица истинности 5.15, предложенная Постом, основана на соглашении, согласно которому операции отрицания присваивается следующее по порядку значение аргумента. Таблица истинности 5.16 была предложена Бочаром, Лукашевичем и Россером, которые исходили из того, что $\neg(\neg p)$ должно быть эквивалентно p.

Таблицы истинности других логических операций могут быть построены на основании их определения с помощью операций конъюнкции и отрицания. Поскольку конъюнкция и отрицание независимы, а остальные операции могут быть через них выражены, то существует в общей сложности $256 \times 12 = 3072$ различных трехзначных логик. Таким образом, количество различных возможных структур многозначной логики чрезвычайно велико.

Таблица 5.15. Таблица истинности отрицания в трехзначной логике (Пост)

p	$\neg p$
И	?
?	*Л*
Л	*И*

Таблица 5.16. Таблица истинности отрицания в трехзначной логике (Бочар и др.)

p	$\neg p$
И	*Л*
?	?
Л	*И*

В некоторых случаях при построении многозначных логик используется следующее понятие значения истинности.

Определение

Действительное число $T(p)$ из интервала $[0, 1]$, которое ставится в соответствие высказыванию p, называют ***значением истинности*** высказывания p.

Значение истинности $T(p)$ можно понимать как вероятность того, что высказывание p истинно, причем два высказывания, имеющие одно и то же значение истинности, можно считать логически эквивалентными. Значение истинности $T(p) = 1$ означает истинность, а значение $T(p) = 0$ — ложность высказывания p. Отрицание $\neg p$ определяется посредством значения истинности следующим

образом: $T(\neg p) = 1 - T(p)$. В трехзначной логике, например, в качестве значений истинности можно принять числа 0, 1/2 и 1. Если значение истинности $T(p) = 1/2$, то значение истинности $\neg p$ также равно 1/2, и p оказывается логически эквивалентным своему отрицанию. Такое высказывание может быть названо сомнительным, причем отрицание этого высказывания также оказывается сомнительным. Указанный подход свидетельствует о наличии тесной связи между многозначными логиками и теорией вероятностей.

Многозначная логика рассматривает однородные логические функции, определяемые на множестве $(0, 1, ..., k - 1)$, состоящем из k элементов. В силу однородности сама функция k-значной логики от n переменных принимает значения из того же конечного множества.

> **Определение**
> Функции k-значной логики определены и принимают значения, входящие в некоторое множество $B_k = \{0, 1, ..., k - 1\}$, называемое ***алфавитом*** данной ***логики***.

Функцию k-значной логики однозначно определяет ее таблица значений (истинности). Множество всех функций k-значной логики обозначают P_k. Количество функций P_k, зависящих от n переменных, равно k^{k^n}. Также как и в двузначной логике, высказывания представляются в виде формул k-значной логики. Элементарные функции представляют собой обобщение аналогичных функций двузначной логики. Рассмотрим основные функции k-значной логики.

Унарные функции

1. Циклическое отрицание:

$$\neg x = x + 1 (\mathrm{mod}\ k),$$

где $y \bmod k$ — остаток от деления y на k.

Таким образом, данная элементарная функция представляет собой обобщение отрицания в смысле «циклического» сдвига значений:

$$\neg x = \begin{cases} x+1, & \text{при } x \neq k-1 \\ 0, & \text{при } x = k-1 \end{cases}.$$

2. Отрицание Лукашевича:

$$N_x = k - 1 - x.$$

Приведенная элементарная функция N_x является другим обобщением операции отрицания в смысле «зеркального» отображения значений.

3. Обобщенное отрицание:

$$I^{\sigma}(x) = \begin{cases} k-1, \text{ при } x = \sigma \\ 0, \text{ при } x \neq \sigma, \text{ где } x, \sigma \in \{0, 1, \ldots, k-1\} \end{cases}.$$

Элементарная функция $I^{\sigma}(x)$ при $\sigma \neq k-1$ является обобщением некоторых свойств отрицания.

4. Характеристическая функция:

$$J^{\sigma}(x) = \begin{cases} 1, \text{ при } x = \sigma \\ 0, \text{ при } x \neq \sigma, \text{ где } x, \sigma \in \{0, 1, \ldots, k-1\} \end{cases}.$$

Функция $J^{\sigma}(x)$ — характеристическая функция значения σ при $\sigma \neq k-1$ представляет собой обобщение операции отрицания.

Бинарные функции

1. Обобщение конъюнкции:

$$\min(x_i, x_j).$$

2. Другое обобщение конъюнкции:

$$x_i\, x_j \pmod{k}.$$

3. Обобщение дизъюнкции:

$$\max(x_i, x_j).$$

Определение

Система функций $f_1 \ldots f_n$ называется ***полной***, если любая функция из P_k может быть представлена в виде формулы, состоящей из этих функций.

В k-значной логике остаются справедливыми некоторые законы двузначной, а именно: ассоциативности, коммутативности, дистрибутивности и т.д. Подобно функциям двузначной логики k-значные логические функции могут быть заданы в виде таблицы. Количество столбцов в таблице равно k^n, где n — количество переменных, входящих в функцию, а количество функций определяется числом k^{k^n}, которое быстро возрастает с ростом k.

В k-значной логике существует k констант $f_0 = 0$, $f_1 = 1$, ..., $f_{k-1} = k-1$. Среди функций одной переменной наиболее часто используемыми являются следующие:

1) Характеристические функции i-го порядка — $f_0(x)$, $f_1(x)$, $f_2(x)$, определенные в таблице 5.17.

2) Отрицание Лукашевича $N_x = k - 1 - x$.

3) Функция циклического отрицания $\neg x = x + 1 \pmod{}$.

Таблицы истинности указанных функций в трехзначной логики будут иметь вид, представленный в таблице 5.17. Среди функций двух переменных наиболее важное значение имеют следующие:

1) k-значная дизъюнкция — $x_1 \vee x_2 = \max(x_1, x_2)$.

2) k-значная конъюнкция — $x_1 \wedge x_2 = \min(x_1, x_2)$.

3) Функция Шеффера-Веббах — $x_1 | x_2 = x_1 \vee x_2 + 1 \pmod{k}$.

4) Сложение по модулю k — $x_1 + x_2 \pmod{k}$.

5) Умножение по модулю k — $x_1 * x_2 \pmod{k}$.

Значения данных функций при $k = 4$ представлены в таблице 5.18.

Таблица 5.17. Функции одной переменной в трехзначной логике

	X			
$F(x)$	x	0	1	2
	$f_0(x)$	2	0	0
	$f_1(x)$	0	2	0
	$f_2(x)$	0	0	2
	N_x	2	1	0
	$\neg x$	1	2	0

Воспользовавшись понятием характеристических функций для двузначного случая, несложно записать СДНФ и СКНФ в многозначной логике. Напомним, что основную роль в СДНФ играют элементарные конъюнкции $x_1^{\sigma_1} \wedge \ldots \wedge x_n^{\sigma_n}$, которые отличны от нуля лишь на одном наборе $(\delta_1, \ldots, \delta_n)$. При этом все они получены из конъюнкции, отвечающей единичному набору, подстановкой функции от одной переменной x^{δ}.

Таблица 5.18. Функции двух переменных в четырехзначной логике

x_1	0	0	0	0	1	1	1	1	2	2	2	2	3	3	3	3
x_2	0	1	2	3	0	1	2	3	0	1	2	3	0	1	2	3
$x_1 \vee x_2$	0	1	2	3	1	1	2	3	2	2	2	3	3	3	3	3
$x_1 \wedge x_2$	0	0	0	0	0	1	1	1	0	1	2	2	0	1	2	3
$x_1 \vee x_2 + 1 \pmod{k}$	1	2	3	0	2	2	3	0	3	3	3	0	0	0	0	0
$x_1 + x_2 \pmod{k}$	0	1	2	3	1	2	3	0	2	3	0	1	3	0	1	2
$x_1 * x_2 \pmod{k}$	0	0	0	0	0	1	2	3	0	2	0	2	0	3	2	1

Вопросы

1. Что понимают под многозначной логикой?
2. Какие существуют разновидности многозначных логик?
3. Сколько различных определений конъюнкции существует в четырехзначной логике?
4. Составьте таблицу истинности конъюнкции в трехзначной логике.
5. В чем состоит единственное ограничение операции отрицания в многозначной логике?
6. Дайте определение понятию значения истинности высказывания.
7. Сформулируйте определение алфавита k-значной логики.
8. Сколько существует различных функций k-значной логики от n переменных?
9. Запишите формулы основных унарных функций k-значной логики.
10. Перечислите важнейшие бинарные функции k-значной логики.
11. Какая система функций k-значной логики называется полной?
12. Перечислите наиболее часто используемые функции одной переменной.
13. Составьте таблицу основных функций двух переменных в четырехзначной логике.

Задания

1. Составьте таблицу истинности функции обобщенного отрицания в трехзначной логике.
2. Постройте таблицу истинности импликации в трехзначной логике, исходя из предположения, что истина не может имплицировать ложь. Возможно ли в данной логике единственным образом выразить импликацию через отрицание и конъюнкцию?
3. По аналогии с двухзначной логикой докажите справедливость дистрибутивного закона в k-значной логике.
4. Постройте таблицы истинности функций одной переменной в четырехзначной логике.
5. Запишите формулы СДНФ и СКНФ в k-значной логике.

раздел 6

Теория графов

6.1. Исторические замечания. Типичные задачи

Принято связывать зарождение теории графов как математической дисциплины с работой Леонарда Эйлера 1736 г., в которой найдено условие существования в связном графе цикла, содержащего все ребра графа (без повторений). Такой цикл теперь называется ***эйлеровым***. Как показывает простой пример (рис. 6.1), есть графы, не являющиеся эйлеровыми циклами.

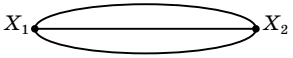


Рис. 6.1. Не эйлеров граф

Первоисточником задачи об эйлеровом цикле сам Эйлер называет известную головоломку о кенигсбергских мостах. В городе Кенигсберге (в 1736 г.) два речных острова A, B соединялись между собой и с берегами C, $Д$ реки Прегель семью мостами (рис. 6.2).

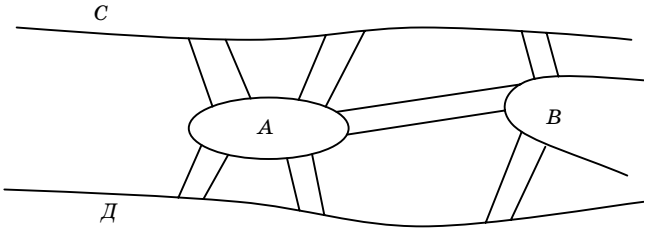


Рис. 6.2. Кенигсбергские мосты

Можно ли совершить прогулку по городу, пройдя по каждому мосту в точности один раз, и вернуться в исходную точку? Изобразив участки суши вершинами, мосты — ребрами (рис. 6.3), получим граф, который не является эйлеровым циклом (см. п. 6.3). Поэтому и задача о кенигсбергских мостах не имеет положительного решения.

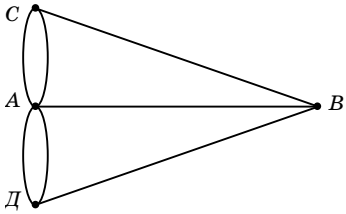


Рис. 6.3. Граф переходов по мостам

Следующий шаг сделал в 1847 г. Кирхгоф. Изучая электрические цепи и абстрагируясь от физической природы устройств, он выделил отдельно комбинаторно-топологическую структуру, называемую сейчас ***графом цепи***, разработал алгоритм нахождения максимального подграфа без циклов (называемого деревом) и с его помощью записал наименьшую независимую систему уравнений цепи. Исследования по электротехнике, электронике, релейно-контактным схемам до настоящего времени индуцируют различные исследования по теории ***графов***, и наоборот.

В 1857 г. в связи с проблемами органической химии Кэли рассматривал задачу перечисления всех деревьев со степенями вершин 1 и 4. Она связана с описанием изомеров предельных (насыщенных) углеводородом C_nH_{2n+2} с данным числом (n) атомов углерода. Задача оказалась непростой, ее смысл понятен из рис. 6.4, на котором изображены 2 изомера при $n = 4$ (бутан и изобутан).

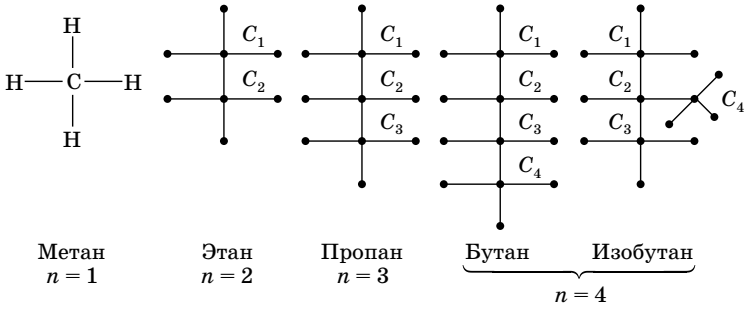


Рис. 6.4. Изомеры C_nH_{2n+2}

Хорошо известна задача коммивояжера, имеющая многочисленные приложения (в том числе в экономике, проектировании):

для заданной системы пунктов (городов) и дорог между ними, схематически представленных плоским[1] связным графом, выбрать кратчайший маршрут коммивояжера, выходящего и возвращающегося в исходный город так, чтобы посетить все остальные пункты и только один раз. Задача не всегда имеет решение, а если имеет, то известные алгоритмы работают эффективно лишь для сравнительно небольших графов — и это, несмотря на обширнейшую библиографию по задаче коммивояжера.

Частный случай этой задачи, не учитывающий длину переходов между пунктами, но сохраняющий все ее принципиальные трудности, сформулировал в 1859 г. В. Гамильтон на примере графа додекаэдра (см. рис. 6.11): обойти замкнутым маршрутом по ребрам многогранника все его вершины, пройдя через каждую вершину ровно один раз (кроме начала и конца пути). Именно эту задачу Гамильтон продал за 25 гиней одному мастеру игрушек в виде игры «Вокруг света». Для произвольного графа задача Гамильтона не обязательно имеет решение и формулируется так: Найти ***простой цикл***, содержащий все вершины графа (такой цикл называется ***гамильтоновым***). Для графа додекаэдра гамильтонов цикл существует. При определенных условиях современная задача коммивояжера, допускающая неоднократное посещение вершин, сводится к отысканию в графе с заданными «длинами» ребер гамильтонового цикла наименьшей длины. Упомянутые условия таковы: каждое ребро (a, b) графа имеет длину, равную длине кратчайшей цепи между вершинами a и b.

Одна из самых знаменитых задач математики — проблема четырех красок — также относится к графам. Еще в прошлом веке было замечено, что каждую конкретную географическую карту можно раскрасить четырьмя цветами так, чтобы всякие две соседние[2] страны окрашивались в разные цвета. Однако доказательство для произвольной карты (то есть для двусвязного плоского графа) удалось дать с использованием ЭВМ только в последние годы. Начиная с середины прошлого века, было дано множество ошибочных доказательств проблемы четырех красок как профессионалами, так и не математиками — всех привлекала простота, эстетичность и громкая известность задачи. Следует отметить результат Хивуда, доказавшего в 1890 г. достаточность пяти красок для раскраски карты.

---

[1] То есть реализуемым как геометрический граф в R^2.

[2] Имеющие общий линейный (а не точечный) участок границы.

Для неплоских графов задача о раскрасках ставится иначе — для вершин; здесь имеется много результатов и нерешенных задач, интересных для теории и практики (см. п. 6.7).

Граф называется ***плоским*** (или ***планарным***), если его можно уложить на плоскость без скрещивания ребер (путем их непрерывной деформации). Топологический критерий планарности графа установил Л. С. Понтрягин (1927 г.) без опубликования этого результата и независимо — К. Куратовский (1930 г.). В связи с важностью планарной реализации электронных схем (печатные платы и пр.) последнее время получены другие критерии планарности и алгоритмы укладки графов (Вагнер, Уитни, Мак-Лейн, Фари, Штейн и др.)

Начиная с 30-х годов нашего столетия, популярность графов и количество работ по чистой теории графов и ее применению неуклонно возрастает. С помощью графа моделируются любые схемы, в которых вычленяются более простые части (вершины) и связи между ними (ребра). Неудивительно, что графы встречаются в исследованиях по социологии, психологии, экономике, теории игр, логике, программированию, теории вероятностей (цепи Маркова), квантовой механике (диаграммы Фейнмана), химии (структура молекул), статистической механике, кристаллофизике, медицине (нервные, сосудистые и другие сети), электро- и радиотехнике, лингвистике, теории расписаний, по транспортным сетям и потокам, вентиляционным сетям и т.д.

6.2. Неориентированные графы и терминология

В дальнейшем будет выяснено, что всякий конечный граф (при самом общем его определении) можно для наглядности представлять себе как совокупность линий, соединяющих заданные точки в трехмерном пространстве. Поэтому полезно иметь определение так называемого ***геометрического графа*** в пространстве R^n: это совокупность точек X и простых[1] кривых Y, концы которых есть точки из X, а сами кривые попарно не имеют общих внутренних

[1] Непрерывных, самонепересекающихся.

точек. Точки из X называются ***вершинам***, кривые из Y — ***ребрами*** графа. На рис. 6.5 изображен граф в пространстве R^2 (или, если угодно, — в R^3), где $X = \{x_1, x_2, x_3, x_4, x_5\}$; $Y = \{y_1, y_2, y_3, y_4, y_5, y_6, y_7\}$. Обозначение для графа: $G = (X, Y)$, $n = n_G$ — число вершин, $m = m_G$ — число ребер графа.

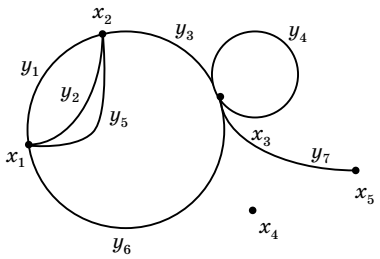


Рис. 6.5. Пример графа в R^2

Основные термины[1]

Смежные вершины (x_2, x_3) — это вершины, соединенные ребром, ***смежные*** ребра (y_2, y_3) — это ребра, имеющие общую вершину.

Вершина x_2 и ребро y_3 ***инцидентны друг другу***, если точка x_2 есть конец кривой y_3. ***Петля*** (y_4) — замкнутое ребро. Изолированная вершина (x_4) неинцидентна ни одному ребру.

Параллельные (***кратные***) ***ребра*** (y_1, y_2, y_5) — это ребра, инцидентные одной паре вершин (x_2, x_1).

Маршрут (соединяющий вершины x_a, x_b) — конечная последовательность ребер и инцидентных им вершин, составляющих непрерывную кривую с концами x_a, x_b. Число ребер в маршруте называется его ***длиной*** (включая вклад повторяющихся ребер). Так, на рис. 4. последовательность ребер y_1, y_2, y_3, y_4, y_5, y_7 (в более подробных обозначениях — чередующаяся последовательность вершин и ребер x_1, y_1, x_2, y_5, x_1, y_2, x_2, y_3, x_3, y_4, x_3, y_7, x_5) есть маршрут длины 6 с концами x_1, x_5.

Маршрут ***замкнут***, если концы его совпадают ($x_a = x_b$).

Маршрут называется ***цепью***, если все его ребра различны, и ***простой цепью***, если все его вершины (кроме, может быть, концов цепи) различны.

Цикл — это замкнутая цепь (***простой цикл***, если цепь простая).

Граф называется ***связным***, если любая пара его вершин соединяется цепью.

Подграф — любая часть графа, сама являющаяся графом.

Компонента (связности) — максимальный связный подграф в графе (например, граф рис. 6.5 имеет 2 компоненты: $\{x_4\}$ и остальная

[1] Для ориентированных графов см. п. 6.4.

часть). **Степень** $\deg x_i = \delta(x_i)$ ***вершины*** x_i — число инцидентных ей ребер ($\delta(x_1) = 4$, $\delta(x_5) = 1$, $\delta(x_4) = 0$), причем петля учитывается как два ребра ($\delta(x_3) = 5$).

Граф называется ***n-связным***, если между любыми его двумя вершинами найдется n цепей, попарно не имеющих общих неконцевых вершин.

Граф $G = (X, Y)$ называется:

- ***пустым*** (или ***ноль***), если множество его ребер пусто;
- ***простым***, если он не содержит петель и параллельных ребер;
- ***полным***, если он простой и каждая пара вершин смежна;
- ***регулярным*** или ***однородным*** (степени r), если степени всех его вершин одинаковы (равны $r = \deg x_i$, $\forall x_i \in X$);
- ***деревом***, если он не содержит циклов и связен;
- ***мультиграфом***, если он содержит параллельные ребра;
- ***псевдографом***, если он содержит петли и кратные ребра.

Граф $G' = (X, Y')$ называется ***дополнением*** простого графа $G = (X, Y)$ с тем же множеством вершин X, если $Y \cap Y' = \varnothing$ и граф $G_0 = (X, Y \cup Y')$ является ***полным***. Иначе говоря, в дополнительном графе G' вершины (x_i, x_j) смежны (соединены ребром $y'_k \in Y'$), если они несмежны в исходном графе G.

Точка сочленения графа — вершина, удаление которой вместе с инцидентными ей ребрами приводит к увеличению числа компонент графа.

Расстояние $d(\upsilon, \vartheta)$ между вершинами υ, ϑ графа G — длина кратчайшей цепи между υ, ϑ. Если от вершины υ до вершины ϑ не ведет ни одна цепь, то $d(\upsilon, \vartheta) = \infty$.

Диаметром $d(G)$ графа G называется максимальное расстояние $d(\upsilon, \vartheta)$ в графе G.

Гранью (***ячейкой***) геометрического графа в R^2 (т.е. ***плоского графа***, в котором ребра пересекаются только в вершинах) называется такая непустая замкнутая подобласть плоскости, что всякие две точки области можно соединить простой (Жордановой) кривой, внутренние (не концевые) точки которой лежат внутри области, не пересекаясь с ребрами графа.

Границей грани считается множество ребер и вершин графа, принадлежащих грани. Всякий ***конечный плоский граф*** имеет в точности одну ***неограниченную*** грань, которая называется ***внешней гранью***. Все остальные (ограниченные) грани называются ***внутренними***.

Эксцентриситет $e(\upsilon)$ вершины $\upsilon \in X$ в связном графе G есть расстояние от υ до наиболее удаленной от нее вершины, а ***радиус*** $r(G)$ графа G — наименьший из эксцентриситетов вершин:

$$e(\upsilon) = \max_{\forall x \in X} d(x, \upsilon), \quad r(\upsilon) = \min_{\forall \upsilon \in X} e(\upsilon).$$

Ясно, что наибольший эксцентриситет равен диаметру графа:

$$d(G) = \max_{\forall \upsilon \in X} e(\upsilon) = \max_{\upsilon} \max_{x} d(x, \upsilon).$$

Вершина υ называется ***центральной вершиной графа G***, если на ней достигается минимум эксцентриситетов, то есть $e(\upsilon) = r(G)$.

Центром графа G называется множество всех его центральных вершин; центр может состоять из единственной вершины, может, — из двух и более вершин. Например, центр простого цикла G_n содержит все n вершин. На рис. 6.6 приведено дерево G с числом вершин $n = 21$, числом ребер $m = 20$, где для каждой вершины указан ее эксцентриситет. Дерево имеет радиус $r(G) = 4$, диаметр $d(G) = 7$, центр дерева состоит из пары вершин $\{u, v\}$, каждая из которых имеет минимальный эксцентриситет 4.

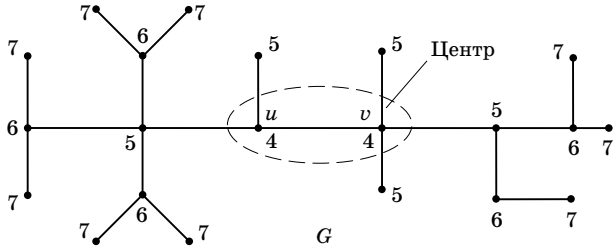


Рис. 6.6. Эксцентриситеты вершин и центр графа G

Задания

1. Верно ли, что замкнутый маршрут нечетной длины содержит простой цикл? А четной длины?
2. Доказать или опровергнуть:
   а) объединение любых двух различных цепей, соединяющих две вершины, содержит простой цикл;
   б) объединение любых двух простых цепей, соединяющих две вершины, содержит простой цикл;
   в) граф связен тогда и только тогда, когда множество X его вершин нельзя разбить на два непересекающихся подмножества X_1, X_2

так, чтобы каждое ребро было инцидентно либо двум вершинам из X_1, либо двум вершинам из X_2;
г) в связном графе любые две длиннейшие простые цепи имеют общую вершину;
д) неверно, что в каждом связном графе все длиннейшие простые цепи имеют общую вершину.

3. Если граф связен и n — число его вершин, то какова нижняя оценка для числа его ребер?
4. Доказать, что связный граф остается связным при удалении ребра тогда и только тогда, когда это ребро содержится в некотором цикле.
5. Доказать, что множество всех ребер конечного связного графа образует простой цикл тогда и только тогда, когда степени всех вершин кратны 2.
6. Доказать, что все ребра конечного связного графа можно включить в некоторый маршрут.
7. Доказать, что цепь z не может быть простой, если из множества всех ее ребер можно образовать другую цепь $z_1(\neq z)$.
8. Пусть граф G имеет n_k вершин степени k, а остальные вершины имеют степень $k + 1$. Тогда $n_k = (k + 1)n - 2m$, где n — число вершин, m — число ребер графа.

6.3. Эйлеровы циклы

Отрицательный ответ в головоломке о кенигсберских мостах (см. п. 6.1) вытекает из следующей теоремы Эйлера.

Теорема 6.1

Граф содержит эйлеров цикл тогда и только тогда, когда он связен и степени всех его вершин — четные.

□ Если граф эйлеров, то он связен и при обходе эйлерова цикла заход и выход в очередную вершину вносит две единицы в ее степень (так как ребра не повторяются). Поскольку проходятся все ребра графа, то степени всех вершин — четные.

Пусть теперь связный граф G имеет четные степени вершин. Выберем вершину x_0 и перейдем от нее к вершине x_1 по некоторому ребру y_0, окрасив его мысленно в какой-либо цвет. Из вершины x_1 двинемся дальше в x_2 по неокрашенному ребру (это возможно в силу $\deg x_1 = 2k$, $k \geq 1$), окрасив его после прохождения. Продвигаясь таким образом по неокрашенным ребрам, мы должны вернуться в исходную вершину x_0 (встречая, возможно, другие

вершины несколько раз), ибо в противном случае вершина $x_i \neq x_0$, из которой мы не можем двигаться дальше, имеет нечетную степень. Вернувшись первый раз в вершину x_0, мы окрасим граф G_0, являющийся по построению эйлеровым циклом. Если $G_0 \neq G$, то в силу связности G существует хотя бы одно неокрашенное ребро $\bar{y}$, соединяющееся с подграфом G_0 неокрашенной цепью z, возможно, нулевой длины (рис. 6.7). Удалим из графа G окрашенные ребра и получим граф с четными степенями вершин. Найдем вершину $\tilde{x}$ прикосновения цепи $z \cup \bar{y}$ к подграфу G_0. Начиная с вершины $\tilde{x}$, можно выделить из неокрашенных ребер новый цикл $G_1 (\supset z \cup \bar{y})$, который в соединении с G_0 также образует эйлеровый цикл в графе $G_1 \cup G_0$. Наращивая таким образом циклы, мы исчерпаем весь конечный граф G эйлеровым циклом. ■

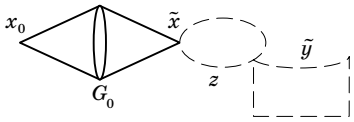


Рис. 6.7. Наращивание цикла

Задания

1. Как нарисовать весь эйлеров цикл одним росчерком, не прибегая в процессе построения к последовательному наращиванию циклов?
2. Доказать, что граф эйлеров тогда и только тогда, когда множество его ребер можно разбить на простые циклы (попарно не пересекающиеся по ребрам).
3. Эйлеровой цепью в графе G называется цепь, содержащая все ребра из G. Доказать следующее обобщение теоремы Эйлера: граф G обладает эйлеровой цепью тогда и только тогда, когда он связен и число вершин нечетной степени равно 0 или 2.

Дополнения

1. Задача китайского почтальона тесно связана с эйлеровыми циклами: в графе G, ребрам которого приписаны положительные веса C_j, найти содержащий все ребра $\{y_j\}_1^m$ замкнутый маршрут Q с минимальным суммарным весом $\sum_{d=1}^{m} n_j c_j$, где n_j — число прохождений ребра y_j в маршруте Q.
   Для всякого связного графа G задача китайского почтальона разрешима, так как число прохождений ребра y_j не ограничивается единицей. Если G содержит эйлеров цикл, то любой такой цикл дает решение задачи почтальона: каждое ребро имеет минимум прохождений и вес цикла $\sum c_j$ — минимальный. Алгоритмы решения задачи китайского почтальона предлагали Беллман и Кук — метод динамического программирования в частном

случае, когда $c_j = 1(\forall_j)$; в общем случае — Эдмонде, Джонсон, Басакер, Саати, Кристофидес.

Приложения задачи китайского почтальона достаточно многочисленны. Например, это минимизируемые по длине пробега транспортные задачи обслуживания потребителей в каждом ребре на заданной сети дорог (доставка почты, продуктов и других грузов, поливка, патрулирование улиц, уборка мусора) или инспектирование каждого отрезка в сети распределенных систем (электрических, телефонных, железнодорожных, вентиляционных) и даже определение и рекомендация экономного маршрута осмотра музея.

6.4. Абстрактные графы и геометрические реализации. Ориентированные графы

Геометрический граф в R^n, определенный в п. 6.2, обычно называется неориентированным. Если на каждом его ребре выбрать определенное направление, то такой граф называется ***ориентированным геометрическим графом*** в R^n, а его ребра с направлениями — ***дугами***. Например, снабдив ребра графа на рис. 6.5 направлениями, получим ориентированный граф (рис. 6.8).

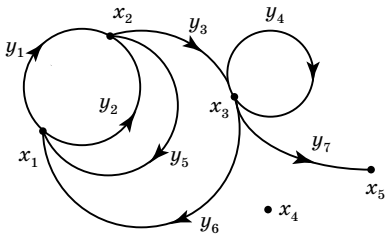


Рис. 6.8. Ориентированный граф

Наоборот, игнорируя направления дуг в ориентированном графе и рассматривая их как ребра, получаем соответствующий ***неориентированный граф***. В связи с этим для ориентированного графа сохраняются все понятия и характеристики, относящиеся, по сути, к соответствующему неориентированному: смежность двух вершин (например, x_2 и x_3), инцидентность дуги и вершины (например, x_2 и y_3), маршрут, цепь, цикл, связность, степень вершины, n-связность и т.д. Естественно, возникают аналогичные понятия и характеристики, учитывающие ориентацию дуг, чаще всего с добавлением эпитета «ориентированный».

Ориентированный маршрут (длины n) — последовательность из n дуг (не обязательно различных), проходимая непрерывным росчерком вдоль направления дуг от начальной вершины до конечной (например, ориентированный маршрут y_1, y_5, y_2, y_3 имеет началом вершину x_1, концом — x_3). Всякий ориентированный

маршрут является маршрутом (неориентированным), обратное может не выполняться (например, маршрут y_6, y_7 между вершинами x_1, x_5 не является ориентированным). Ориентированный маршрут без повторяющихся дуг называется ***путем*** (y_1, y_3, y_7); замкнутый путь — ***контуром***. Путь является цепью, контур — циклом; обратное, вообще говоря, неверно. ***Путь*** (контур) без повторяющихся внутренних вершин называется ***простым***. Орграф называется ***сильно связным***, если для любой пары различных вершин v, w существует путь из v в w и путь из w в v.

Хотя геометрические графы (в R^n) имеют в качестве вершин X и ребер (дуг) Y реальные геометрические объекты в R^n (точки и кривые), метрические характеристики этих объектов (формы кривых, их длины и расстояния между точками) не играют роли. Принимаются в расчет лишь свойства инциденции вершин x_i и ребер (дуг) y_k. В связи с этим определение графа как геометрического объекта в метрическом пространстве R^n является избыточным с логической точки зрения, хотя и наглядным, опирающимся на геометрическую интуицию.

Определение абстрактного графа. Графом G (ориентированным) называется тройка $G = (X, Y, f)$, где X, Y — произвольные множества, f: $Y \to X \times X$ — отображение множества Y в декартово произведение множества X на себя. Элементы множества X называются вершинами, множества Y — дугами, отображение f — инцидентором. Элементами $X \times X$ являются упорядоченные пары (x_i, x_j) и равенство $f(y_k) = (x_i, x_j)$ можно интерпретировать следующим образом: «дуга» y_k выходит из «вершины» x_i и заходит в «вершину» x_j; такая дуга y_k называется инцидентной вершинам x_i, x_j. Например, для геометрического графа на рис. 6.8 можно построить соответствующий ему абстрактный граф так: ввести абстрактные множества $X = \{x_i\}_{i=1}^{5}$, $Y = \{y_k\}_{k=1}^{7}$ и инцидентор f:

$$f(y_1) = f(y_2) = (x_1, x_2);\ f(y_3) = (x_2, x_3);\ f(y_4) = (x_3, x_3);$$

$$f(y_5) = (x_2, x_1);\ f(y_6) = (x_3, x_1);\ f(y_7) = (x_3, x_5).$$

Неориентированный абстрактный граф определяется с помощью тройки $G = (X, Y, \varphi)$, где множество X называется множеством вершин, Y — множество ребер, а φ: $Y \to <X \times X>$ — отображение ребер в симметризованное декартово произведение вершин на себя, в котором пары (x_i, x_j) и (x_j, x_i) отождествлены. Если через Θ: $X \times X \to <X \times X>$ обозначить соответствующее отображение симметризации, то для ориентированного графа $G = (X, Y, f)$,

$f\colon Y \to X \times X$ соответствующий ему неориентированный граф можно определить как $G_0 = (X, Y, \Theta f)$.

Часто графы обозначаются в виде первых двух множеств тройки:

$$G = (X, Y),$$

инцидентор же f подразумевается неявно.

Два графа G и G' называются ***изоморфными*** ($G \sim G'$), если существуют взаимно однозначные отображения вершин и ребер $\varphi\colon X \to X'$, $\Psi\colon Y \to Y'$ соответственно, сохраняющие отношения инциденции: $f'\psi = (\varphi \times \varphi)f$. Иначе говоря, ребро (дуга) y инцидентно (заходит, выходит) вершине x графа G тогда и только тогда, когда в графе G' соответствующее ребро (дуга) $y' = \Psi(y)$ инцидентно (заходит, выходит) вершине $x' = \varphi(x)$. Итак, ***изоморфизм*** $\Phi\colon G \to G'$ есть преобразование $\Phi(G) = \Phi(X, Y, f) = (\varphi(X), \Psi(Y); (\varphi \times \varphi)f\Psi^{-1})$, но для простоты мы будем писать иногда $X' = \Phi(X)$, $Y' = \Phi(Y)$, $f' = \Phi(f)$. Изоморфизмы графа в себя называются ***автоморфизмами***; очевидно, они образуют группу. Обратно можно показать, что всякая абстрактная конечная группа изоморфна группе автоморфизмов некоторого графа G (с конечным числом вершин и дуг). *Если граф G изоморфен геометрическому графу G' в R^n, то G' называется* ***геометрической реализацией*** *графа G в* ***пространстве*** $\boldsymbol{R^n}$. Например, геометрический граф в R^3 на рис. 6.9 изоморфен геометрическому графу в R^2 на рис. 6.10.

Граф (абстрактный или геометрический), имеющий геометрическую реализацию в R^2 (иначе — плоскую реализацию), называется ***планарным***. Геометрический граф в R^3 на рис. 6.9 — планарный, пятивершинный граф, на рис. 6.11 — непланарный (подробнее см. задания и дополнения 9, 10).

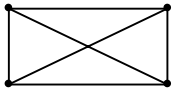


Рис. 6.9. Граф K_4

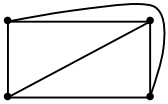


Рис. 6.10. Плоское изображение K_4

Рис. 6.11. Граф K_5

Изоморфизм графов является *отношением эквивалентности*, поэтому все множество графов разбивается на классы изоморфных графов. Часто мы не будем различать изоморфные графы (то есть изучать классы) и, по возможности, иметь дело с геометрическими представителями (реализациями).

Следует отметить алгоритмическую трудность проверки изоморфизма двух графов — это комбинаторная задача, которую можно свести к выяснению подобия двух матриц смежности (см. ниже п. 6.5). Геометрические представления графов на чертежах здесь мало эффективны. Например, не сразу видно, что небольшие шестивершинные графы на рис. 6.12 изоморфны друг другу.

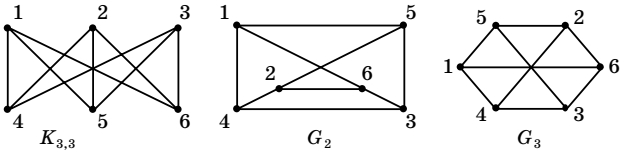


Рис. 6.12. Изоморфные графы

Граф G называется ***конечным***, если множества вершин X и ребер Y конечны, и ***локально конечным***, если степени всех вершин конечны. ***Бесконечные дискретные графы*** (*со счетными множествами* X, Y) встречаются в приложениях.

Связь с отношениями. Простейшие свойства

Если R — бинарное отношение на множестве X, то факт $x_1 R x_2$ можно изобразить наличием дуги из элемента x_1 в элемент x_2. Таким образом, *между бинарными отношениями и ориентированными графами без параллельных дуг устанавливается взаимно однозначное соответствие*. Поскольку нас будут интересовать графы, допускающие параллельные дуги (например y_1, y_2 на рис. 6.8), то мы имеем дело с объектом более общим, чем бинарные отношения.

В заключение коснемся простейших комбинаторных свойств конечных графов. Поскольку появление каждого нового ребра добавляет по единице к степеням двух инцидентных ему вершин (или двойку к степени одной вершины в случае петли), сумма степеней всех вершин графа в два раза превышает число ребер $m = |Y|$ (***«лемма о рукопожатиях»***):

$$\sum_{\forall x \in X} \delta(x) = 2|Y|.$$

Следствие. *Число вершин нечетной степени четно.*

Действительно, сумма четных степеней — четна, поэтому и сумма нечетных степеней должна быть четной.

В орграфе, кроме степени вершины $\delta(x)$, вводится ***положительная степень*** $\delta^+(x)$ — число дуг, начинающихся в вершине x, и ***отрицательная степень*** $\delta^-(x)$ — число дуг, заканчивающихся в вершине X. По другой терминологии $\delta^+(x)$, $\delta^-(x)$ — ***полустепени «исхода и захода»*** в вершине x. Очевидно,

$$\delta^+(x) + \delta^-(x) = \delta(x),$$

$$\sum_{\forall x \in X} \delta^+(x) = \sum_{\forall x \in X} \delta^-(x) = 1/2 \sum_{\forall x \in X} \delta(x) = |Y| = m.$$

Задания и дополнения

1. Эйлеров контур в орграфе, по определению, содержит все дуги и только по одному разу. Скорректировать доказательство теоремы Эйлера 6.1 (для неорграфов) так, чтобы получилась следующая теорема. Орграф обладает эйлеровым контуром тогда и только тогда, когда он является связным и псевдосимметричным: $\delta^+(x) = \delta^-(x)$, $\forall x \in X$.
2. Граф $G = (X, Y, f)$ называется ***симметрическим***, если для каждой пары вершин (x_i, x_j) число дуг, идущих из x_i в x_j, равно числу дуг, идущих из x_j в x_i: $|f^{-1}(x_i, x_j)| = |f^{-1}(x_j, x_i)|$. Проверить, что симметрический граф является псевдосимметрическим, а обратное — неверно.
3. Доказать, что любой контур может быть разделен на несколько простых контуров.
4. Доказать, что любой непростой путь из v в w можно разделить на простой путь и один или несколько простых контуров.
5. Если $\delta^+(x) \geq 1$ и $\delta^-(x) \geq 1$ $\forall x \in X$, то верно ли, что любая вершина орграфа принадлежит, по крайней мере, одному контуру?
6. Доказать, что если $\delta^+(x) > 0$ или $\delta^-(x) > 0$ для $\forall x \in X$, то в орграфе существует, по крайней мере, один контур.
7. Доказать, что ориентированный граф сильно связан тогда и только тогда, когда существует замкнутый ориентированный маршрут, в который каждая дуга входит, по крайней мере, один раз.
8. Пусть вершины соответствуют целым числам от 0 до 12. Построить графы, характеризующие бинарные отношения R, где vRw означает: а) $V = W^2$; б) v и w сравнимы по модулю 4 (то есть $v - w$ кратно 4); в) v и w взаимно простые.
9. Известная головоломка — «Задача о трех колодцах» — состоит в соединении дорогами трех враждующих домов с тремя колодцами (источниками) так, чтобы каждый дом соединялся с каждым колодцем, но дороги не пересекались (за исключением начальных и конечных точек). Убедитесь, что задача не имеет решения: ее решение означало бы, что граф $K_{3,3}$ (рис. 6.12) имеет плоскую реализацию, что неверно.

10. Графы K_5 (рис. 6.11) и $K_{3,3}$ (рис. 6.12) дают примеры неплоских графов. Оказывается, что эти две конфигурации являются характерными для неплоских графов. Этот непростой результат был установлен в 1927 г. Л. С. Понтрягиным (без опубликования), и в 1930 г. — независимо К. Куратовским.

Теорема Понтрягина — Куратовского

Граф G является планарным тогда и только тогда, когда он не содержит подграфов вида G_1, G_2 (рис. 6.13). G_1 и G_2 называются графами Понтрягина — Куратовского; имеют простые цепи между помеченными вершинами и могут быть получены заменой ребер в графах $K_{3,3}$ и K_5 произвольными простыми цепями.

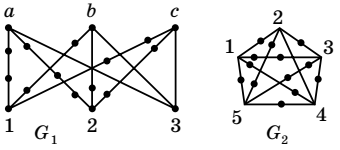


Рис. 6.13. Графы Понтрягина — Куратовского

11. Доказать строго, что графы Понтрягина — Куратовского не планарны.

Указание. Достаточно рассмотреть графы K_5 (рис. 6.11) и $K_{3,3}$ (рис. 6.12). Предположить, что они планарны, по теореме Эйлера вычислить число граней f; учитывая оценки снизу 4 (в $K_{3,3}$) и 3 (в K_5) для числа ребер в грани, просуммировать числа ребер по всем граням и сравнить с истинным числом ребер.

6.5. Матрица смежности, изоморфизмы и операции над графами

1. Квадратная матрица A размера $n = |X|$, элемент которой $a_{i,j}$ равен числу дуг, идущих от вершины x_i к вершине x_j, называется ***матрицей смежности орграфа*** $G = (X, Y)$. Для неориентированного графа элемент $a_{i,j}$ матрицы смежности A_0 определяется как число ребер, соединяющих вершины x_i и x_j (очевидно, A_0 всегда симметрична относительно главной диагонали: $A_0 = A_0^T$). Ненулевое значение элемента $a_{i,j}$ характеризует смежность вершин x_i и x_j в графе, отсюда и название — матрица смежности. Ясно, что граф с точностью до изоморфизма восстанавливается по матрице $A(A_0)$. Для орграфа на рис. 6.8 и соответствующего неорграфа на рис. 6.5 матрицы смежности имеют вид:

$$A = \begin{array}{c} \\ x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{array} \begin{array}{c} \begin{array}{ccccc} x_1 & x_2 & x_3 & x_4 & x_5 \end{array} \\ \begin{bmatrix} 0 & 2 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \end{array}; \quad A_0 = \begin{array}{c} \\ x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{array} \begin{array}{c} \begin{array}{ccccc} x_1 & x_2 & x_3 & x_4 & x_5 \end{array} \\ \begin{bmatrix} 0 & 3 & 1 & 0 & 0 \\ 3 & 0 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{bmatrix} \end{array}.$$

Очевидно, для произвольного орграфа G и соответствующего неориентированного графа G_0 («потерявшего» ориентацию дуг) матрицы смежности A и A_0 связаны соотношением $A + A^T = A_0$.

2. Уже отмечалось, что практически (и визуально для геометрических графов) трудно установить изоморфизм двух графов (см. рис. 6.12). Матрицы смежности позволяют сформулировать простой алгебраический критерий изоморфизма.

Теорема 6.2

Графы $G = (X, Y)$, $G' = (X', Y')$ изоморфны тогда и только тогда, когда они имеют одинаковое число вершин, и матрица смежности $A(G')$ получается из матрицы $A(G)$ последовательными перестановками строк с одновременной перестановкой одноименных столбцов.

В неориентированных графах то же верно для матриц A_0. Иначе говоря, существует такая подстановка δ в группе подстановок множества $\{1, \ldots, n\}$, $n = |X| = |X'|$, что для всех элементов матриц верно $a_{ij}(G') = a_{\delta(i)\cdot\delta(j)}(G)$.

□ Из определения матрицы смежности следует, что взаимная замена номеров вершин x_1, x_2 в графе G приводит к перестановке местами первых двух строк и одновременно первых едвух столбцов в матрице $A(G)$, и наоборот. Такое преобразование является простейшим изоморфизмом над графом G. В то же время изоморфизм Ф над G однозначно определяется (восстанавливается) по своему действию на вершины: $\Phi(X) = X'$, а значит — по некоторой подстановке δ на множестве номеров вершин. Но подстановка δ раскладывается в суперпозицию простейших подстановок $(i \leftrightarrow j)$, поэтому изоморфизм Ф: $G \to G'$ эквивалентен суперпозиции простейших изоморфизмов и указанному в условии теоремы преобразованию над матрицей $A(G)$. ■

Количество всех подстановок равно $n!$, поэтому непосредственное применение теоремы 6.2 для распознавания изоморфизма графов, состоящее в переборе $n!$ вариантов, неэффективно при большом числе вершин $n = |X|$. Существуют различные приемы и алгоритмы, которые для графов того или иного специфического вида существенно уменьшают объем вычислений при выяснении изоморфизма.

3. Для двух орграфов $G = (X; Y; f)$, $G_1 = (X; Y_1; f_1)$ с одинаковыми множествами вершин определим *операции сложения и умножения*.

Сумма $G + G_1 = (X; Y \cup Y_1; f + f_1)$ получается объединением дуг Y и Y_1. При умножении $GG_1 = (X; Y_0; f_0)$ из вершины x_i в вершину x_j идет столько дуг, сколько существует путей длины 2 в графе $G + G_1$ из вершины x_i в вершину x_j таких, что первая дуга принадлежит Y, вторая — Y_1.

Теорема 6.3

Для суммы и произведения орграфов (с одинаковыми множествами вершин) матрицы смежности соответственно складываются и умножаются, и наоборот:

$$A(G + G_1) = A(G) + A(G_1); A(GG_1) = A(G) \cdot A(G_1).$$

□ Утверждение для суммы очевидно: в графе $G + G_1$ число дуг из x_i в x_j равно $a_{ij}(G) + a_{ij}(G_1)$. В графе $G + G_1$ число различных путей с последовательностью вершин вида $\{x_i, x_k, x_j\}$, таких, что дуги из x_i в x_k принадлежат графу G и дуги из x_k в x_j принадлежат графу G_1, равно $a_{ik}(G) \cdot a_{kj}(G_1)$ (рис. 6.14). В этой последовательности вершин (x_i, x_k, x_j) промежуточная вершина x_k фиксирована. Общее число путей длины 2 с первой дугой из Y, второй — из Y_1 равно $\sum_{k=1}^{m} a_{ik}(G) \cdot a_{kj}(G_1)$ — общему элементу матрицы $A(G) \cdot A(G_1)$. ■

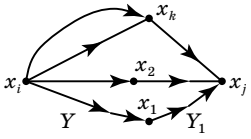


Рис. 6.14. Граф $G + G_1$

Следствие 6.1. *Если A — матрица смежности орграфа G, то элемент a_{ij}^n матрицы A^n (натуральной степени n) равен числу*

различных ориентированных маршрутов длины n, идущих из вершины x_i в x_j в графе G.

□ Утверждение верно при $n = 1$ по определению матрицы смежности. Пусть оно верно для $k = n - 1$: элемент a_{ij}^{n-1} матрицы A^{n-1} равен числу ориентированных маршрутов длины $n - 1$ из x_i в x_j в графе G и одновременно числу дуг из x_i в x_j в графе G^{n-1}, определенном последовательным умножением двух сомножителей. Поскольку $A^n = A^{n-1}A$, то, применяя теорему 6.3 к произведению графов $G^{n-1}G$, получаем утверждение для матрицы A^n (n-го шага индукции). ■

Следствие 6.2. *В орграфе G существует ормаршрут длины n тогда и только тогда, когда $A^n \neq 0$; не существует контуров тогда и только тогда, когда матрица смежности нильпотентна: $A^r = 0$ при некотором r.*

Замечание. В связи со второй частью следствия 2 полезно отметить, что в случае контура длины r с вершинами $\alpha, \beta, \ldots, \gamma$ все степени A^{nr} ($n = 1, 2, \ldots$) матрицы смежности A имеют отличные от нуля диагональные элементы с номерами $(\alpha, \alpha), (\beta, \beta), \ldots, (\gamma, \gamma)$. Если контуров в графе нет (в частности, если более того, нет циклов), то длины всех путей ограничены числом дуг m и $A^{m+1} = 0$, элемент a_{ij}^n матрицы A^n равен числу различных простых путей длины n из вершины i в вершину j.

4. Вопросы ***достижимости*** (одной вершины из другой путем определенной длины) связаны, конечно, со свойствами степеней матрицы смежности A графа. Оказывается, свойства степеней A, $A^2, A^3, \ldots$ произвольной матрицы A с неотрицательными элементами $a_{ij}^n \geq 0$ (например, матрицы стоимостей в экономике, вероятностей переходов и т.п.) также дают ценную информацию о связях между локальными «узлами» объекта в данный момент или различные моменты времени.

Практически полезные задачи на подсчет числа контуров заданной длины n естественно связаны с матрицей смежности A (достаточно проанализировать диагональные элементы матрицы A^n). Задачи такого сорта встречаются при кодировании, в криптографии, лингвистике, при моделировании графом дискретных физических процессов — в связи с «зацикливающимися» подсистемами.

Задания

1. Установить изоморфизм графов G_1, G_2 на рис. 6.15.

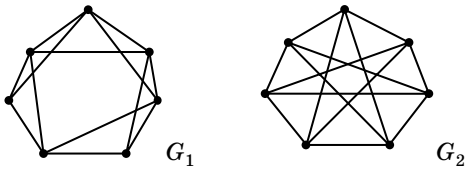


Рис. 6.15. Графы степени 4

2. Докажите, что:

   а) граф G не является связным тогда и только тогда, когда некоторой перестановкой строк с одноименной перестановкой столбцов матрица смежности $A(G)$ приводится к распадающейся:

$$\begin{bmatrix} A_1 & 0 \\ 0 & A_2 \end{bmatrix}, \text{ где } A_i \text{ — квадратные матрицы;}$$

   б) максимальное число диагональных квадратных блоков, на которые может распадаться матрица смежности при некоторой нумерации вершин, равно числу связных компонент графа.

3. Дать определение сильно связного графа в терминах поведения элементов a_{ij}^n последовательных степеней A^n ($n = 1, 2, \ldots$) матрицы смежности A.

4. Как выразить через элементы матриц A^n расстояние $d(x_i, x_j)$ между вершинами x_i, x_j (длину кратчайшей цепи между этими вершинами)?

5. Докажите, что если G — регулярный неориентированный граф степени r, то все корни λ_k «характеристического» полинома $\det(A_0 - \lambda I) = X_0(\lambda)$ располагаются в круге радиуса r: $|\lambda_k| \le r$, причем один из корней равен $r(= \lambda_{k0})$.

6.6. Матрица инциденций

1. Коль скоро вершины и дуги орграфа $G = (X, Y, f)$ без петель занумерованы: $X = \{x_1, \ldots, x_n\}$, $Y = \{y_1, \ldots, y_m\}$, действие инцидентора f можно охарактеризовать $(n \times m)$ — матрицей инциденций $B = \{b_{ij}\}$, где по определению:

$$b_{ij} = \begin{cases} +1, \text{если дуга } y_j \text{ исходит из вершины } x_i; \\ -1, \text{если дуга } y_j \text{ заходит в вершину } x_i; \\ 0, \text{если дуга } y_j \text{ не инцидентна вершине } x_i. \end{cases}$$

Для неорграфа G без петель матрица инциденций $R = \{r_{ij}\}$ определяется так: $r_{ij} = 1$, если ребро y_j инцидентно вершине x_i, и $r_{ij} = 0$ — в противном случае.

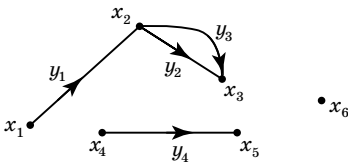


Рис. 6.16. Граф с тремя компонентами

Для графа на рис. 6.16 и для соответствующего неорграфа с потерей ориентации дуг имеем:

$$B = \begin{array}{c} \\ x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \end{array} \begin{array}{c} \begin{array}{cccc} y_1 & y_2 & y_3 & y_4 \end{array} \\ \begin{bmatrix} 1 & 0 & 0 & 0 \\ -1 & 1 & 1 & 0 \\ 0 & -1 & -1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & -1 \\ 0 & 0 & 0 & 0 \end{bmatrix} \end{array}; \quad R = \begin{array}{c} \\ x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \end{array} \begin{array}{c} \begin{array}{cccc} y_1 & y_2 & y_3 & y_4 \end{array} \\ \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix} \end{array}.$$

Матрицы ориентированного и соответствующего симметризованного неориентированного графов связаны соотношением $|b_{ij}| = r_{ij}$. Очевидно также, что в матрице B каждый столбец содержит в точности одну $+1$ и одну -1, в R — две $+1$, остальные элементы — нули. По этой причине сумма всех строк в B равна нулевому вектору, в R — равна нулю по модулю 2.

Если граф G имеет n компонент связности, то, нумеруя вершины и ребра отдельными группами в каждой компоненте (или переставляя столбцы и строки в B и R), получим для матрицы инциденций $B(R)$ блочно-диагональную структуру:

$$B = \begin{bmatrix} B_1 & 0 & \dots & 0 \\ 0 & B_2 & \dots & 0 \\ & \dots & & \\ 0 & 0 & \dots & B_n \end{bmatrix}.$$

Верно и обратное.

2. **Теорема 6.4**

Ранг матрицы инциденций B связного орграфа с n вершинами равен $n - 1$ (числу строк без одной).

□ Вследствие связности графа каждой вершине инцидентна хотя бы одна дуга, так что в матрице B нет нулевых строк. Вычеркнем последнюю строку $\vec{b}_n$ и предположим, что для оставшейся матрицы B_0 линейная комбинация ее вектор-строк равна нулю:

$$\sum_{i=1}^{n-1}\alpha_i\vec{b}_i = 0.$$

Вершина x_n связана дугой y_{j1} с некоторой вершиной x_{i1}, поэтому в вычеркнутой строке $\vec{b}_n$ элемент b_{nj1} отличен от нуля, а в строке $\vec{b}_{i_1}$ — элемент $b_{i1j1} \neq 0$. Поскольку в столбце $j1$ матрицы B_0 только один элемент b_{i1j1} отличен от нуля, то $\alpha_{i1} = 0$. В графе G система вершин $\{x_n, \ldots, x_{i1}\}$ связана некоторой дугой y_{j2} с новой вершиной x_{i2}, поэтому $b_{i2j2} \neq 0$. Второй ненулевой элемент столбца j_2 матрицы B принадлежит одной из строк $\vec{b}_n$, $\vec{b}_{i_1}$, следовательно, $\alpha_{i2} = 0$. Продолжая процесс присоединения вершин (строк): $\{x_n\}$, $\{x_n, x_{i1}\}$, $\{x_n, x_{i1}, x_{i2}\}$, ..., $\{X\}$, мы последовательно получим нулевые значения для всех коэффициентов α_i в линейной комбинации вектор-строк матрицы B_0 так, что

$$\text{rang } B_0 = \text{rang } B = n - 1.$$ ■

Матрицы инциденций и смежности можно связать интересным соотношением, если ввести диагональную матрицу степеней вершин $\Delta = \{\delta_{ij}\}$ графа G, $\delta_{ij} = 0$ $(i \neq j)$, $\delta_{ii} = i(x_i)$ — степень вершины x_i.

Теорема 6.5

Если G — орграф без петель, то его матрицы смежности A, инциденций B и степеней вершин Δ связаны соотношением:

$$(A + A^T) + BB^T = \Delta. \qquad (6.1)$$

□ Элемент $\tilde{a}_{ij}$ матрицы $A + A^T$ равен числу Π_{ij} ребер, соединяющих вершины x_i, x_j; $\tilde{a}_{ii} = 2a_{ii} = 0$. Для матрицы BB^T элемент с индексами i, j равен $\sum_{k=1}^{m} b_{ik}b_{jk}$, что дает $(-\Pi_{ij})$ при $i \neq j$ и степень $\delta(x_i)$ вершины x_i при $i = j$. ■

Следствие 6.2

1. $\det(A + BB^T + A^T) = \prod_{i=1}^{n}\delta(x_i)$.

2. *Граф регулярен тогда и только тогда, когда матрица $(A + A^T + BB^T)$ скалярна, причем $A + A^T + BB^T = rI$, где $r = \delta(x_i)$ — степень графа, I — единичная матрица.*

3. *Граф без петель является полным, если*

$$\det(A + A^T + BB^T) = (n-1)^n,$$

где $n = |X|$, *и наоборот*.

Задания

1. Покажите, что матрица инциденций $B(R)$ определяет граф G с точностью до изоморфизма.
2. Проверить формулу (6.1) для конкретных значений матриц графа рис. 6.2:
   2.1. Для полного графа без петель и параллельных ребер
   $$(A + A^T)^2 = (n-1)(A + A^T) + BB^T.$$
   2.2. Для регулярного графа степени r
   $$\det(A + A^T + BB^T) = r^n.$$
   Верно ли обратное (сравни с п. 3 следствия 6.2)?
   2.3. Ранг матрицы инциденций p-компонентного графа с n вершинами равен $n - p$.

6.7. Раскраски

1. О проблеме четырех красок для плоских карт уже говорилось в п. 6.1. Настолько многих она вводила в искушение, что Фрэнк Харари писал в 1969 г.: «Гипотезу четырех красок можно с полным основанием назвать еще «болезнью четырех красок», так как она во многом похожа на заболевание. Она в высшей степени заразна. Иногда она протекает сравнительно легко, но в некоторых случаях приобретает затяжной или даже угрожающий характер. Никаких прививок против нее не существует; правда, люди с достаточно здоровым организмом после короткой вспышки приобретают пожизненный иммунитет. Этой болезнью человек может болеть несколько раз, и она подчас сопровождается острой болью, но ни одного летального исхода зарегистрировано не было. Известен, по крайней мере, один случай передачи болезни от отца к сыну, так что, может быть, она наследственна».

Сравнительно недавно, в 1976 г., «болезнь четырех красок» научились лечить — гипотеза подтвердилась, наконец. Однако многоступенчатое доказательство этого факта является «вещью в себе», его настолько трудно проверить, что некоторые специалисты сомневаются в том, что гипотеза решена. Ступеньки этого доказательства таковы. Сначала, используя идею Г. Биркгофа об описании свойств специальных «неприводимых» (не раскрашиваемых

четырьмя цветами) графов, математики довели число вершин 4-раскрашиваемых графов до 96. Затем в 1969 г. Х. Хееш свел проблему четырех красок к вопросу 4-раскраски большого, но конечного числа графов. Позже число таких графов было доведено до 1482. Наконец, в 1976 г. К. Аппель и В. Хейкен (с коллективом математиков и программистов) правильно раскрасили все 1482 графа с помощью мощного компьютера, затратив на раскраску около 2000 часов машинного времени.

И все же эта столь долго простоявшая проблема — лишь частный случай целого направления раскрасок графов. Перефразируем задачу раскраски стран связного графа G в задачу раскраски вершин другого плоского графа $\overline{G}$ (не требуя для общности, чтобы G был двусвязным). Внутри каждой области Q_i (грани, страны) графа G, включая внешнюю, пометим одну точку $\overline{x}_i$ — вершину искомого графа $\overline{G}$.

Если в G страны Q_i и Q_j имеют общее ребро y, то соединим в $\overline{G}$ вершины $\overline{x}_i$ и $\overline{x}_j$ ребром $\overline{y}$, пересекающим только y. Граф $\overline{G}$ называется ***геометрически двойственным*** к плоскому графу G.

Следствие. *Задача раскраски стран географической карты (областей плоского графа G) эквивалентна задаче раскраски вершин геометрически двойственного графа $\overline{G}$, также плоского.*

При этом условие правильной раскраски вершин состоит в том, что смежные (в $\overline{G}$) вершины $\overline{x}_i$, $\overline{x}_j$ приобретают различные цвета. Поскольку операция $G \to \overline{G}$ отображает весь класс связных плоских графов на себя взаимно однозначно (это вытекает из равенства $\overline{\overline{G}} = G$), то проблема четырех красок переформулируется так:

всякий плоский граф допускает правильную раскраску вершин не более, чем четырьмя красками.

Замечание. Эквивалентность проблемы раскраски стран и проблемы раскраски вершин в классе всех плоских графов можно установить, не пользуясь точным равенством $\overline{\overline{G}} = G$. Важно, что при переходе к двойственному графу $G \to \overline{G}$ раскраска стран G переходит в раскраску вершин $\overline{G}$, раскраска вершин G — в раскраску стран $\overline{G}$.

Произвольный (не обязательно плоский и связный) граф $\overline{G}$ называется ***r-раскрашиваемым*** (*или* ***r-хроматическим***), если он допускает правильную раскраску вершин r красками.

Не уменьшая общности, можно считать, что при раскраске вершин граф не содержит параллельных ребер и петель, то есть является простым.

Граф G_1 на рис. 6.17 3-раскрашиваем, G_2 — 4, 3, 2-раскрашиваем.

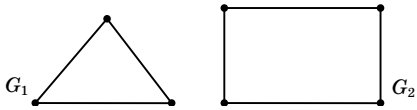


Рис. 6.17. 3-раскрашиваемый и 2-раскрашиваемый графы

Минимальное число цветов, необходимое для правильной раскраски вершин графа $G(\min r)$ называется ***хроматическим числом*** $\gamma(G)$.

Некоторые практические задачи связаны с раскраской ребер графа G, которая считается правильной, если каждые два смежных ребра (инцидентных одной вершине) окрашиваются в различные цвета. Наименьшее число цветов, допускающих правильную раскраску ребер, называется ***хроматическим классом*** $\gamma'(G)$ графа или ***индексом***. В принципе, раскраска ребер графа G сводится к раскраске вершин некоторого графа G^1, называемого ***дуальным***[1] по отношению к G, так что $\gamma'(G) = \gamma(G^1)$ — хроматический класс G равен хроматическому числу дуального графа. Это свойство дуального графа G^1 очевидным образом индуцирует его определение: вершины дуального графа G^1 находятся во взаимно однозначном соответствии с ребрами G, и две вершины в G^1 соединены ребром, если (и только если) соответствующие два ребра в G смежны.

2. Если $\gamma(G) = 1$, то граф G не имеет ребер (состоит из изолированных вершин; петли в вопросах раскраски вершин не учитываются). Всякое дерево имеет хроматическое число 2; граф с хроматическим числом $\gamma(G) = 2$ называется ***бихроматическим***. Всякая раскраска G двумя красками (рис. 6.18) разбивает вершины на два одноцветных множества X_1, X_2, внутри каждого из которых вершины *попарно несмежны* (*независимы*, как иногда говорят). Это обстоятельство находит многочисленные приложения на практике и при различных доказательствах (см., например, п. 6.9).

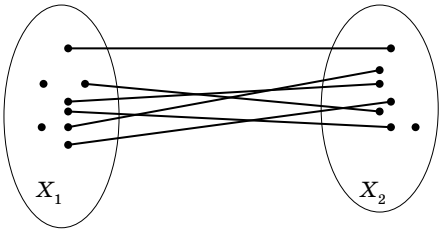


Рис. 6.18. Бихроматичность

Нам понадобится более-менее очевидная лемма.

[1] По другой терминологии, граф G^1 называется реберным графом для G и обозначается $L(G)$.

Лемма 6.1

Всякий замкнутый маршрут

$$Q = \{x_0, y_1, x_1, y_2, x_2, \ldots, y_n, x_0\}$$

можно исчерпать (получить пустую цепь) путем многократного применения следующих операций.

Ф. Удаление непрерывно проходимого подмаршрута, являющегося простым циклом.

Ψ. Удаление тривиального замкнутого подмаршрута длины 2 вида $\{x_k, y_k, x_{k+1}, y_k, x_k\}$.

□ Действительно, применим к Q последовательно операцию Ф до тех пор, пока это возможно. Поскольку оставшиеся подмаршруты Q_i ($\subset Q$) не входят ни в один цикл, то каждое ребро $q \in Q_i$ встречается в объединении подмаршрутов $\bigcup_i Q_i$ одинаковое число раз в прямом и обратном направлениях. Применяя теперь последовательно операцию Ψ, получим пустую цепь. ■

Теорема 6.6 *(Кениг)*

Граф G является бихроматическим тогда и только тогда, когда все его циклы имеют четную длину.

Замечание. Следующие три свойства графа G эквивалентны.

1. Все замкнутые маршруты имеют четную длину.
2. Все циклы имеют четную длину.
3. Все простые циклы имеют четную длину.

Действительно, импликация $1 \Rightarrow 2 \Rightarrow 3$ очевидна, а $3 \Rightarrow 1$ вытекает из леммы, так как удаляемые в доказательстве леммы подмаршруты имеют четную длину.

□ Пусть граф G бихроматичен; реализуем какую-нибудь раскраску двумя цветами (в каждой компоненте связности). Концы всякой цепи четной длины имеют одинаковый цвет, нечетной — разные цвета, поэтому замкнутые маршруты (циклы) нечетной длины отсутствуют. Обратно, пусть все замкнутые маршруты графа G имеют четную длину. Окрасим в цвет α одну вершину x_0, затем в цвет β — ее окружение (множество смежных вершин), в цвет α — окружение вершин цвета β и так далее. За конечное число шагов все вершины графа окрасятся в цвета α и β, причем окрашивание вершины x_i дважды означает, что между первым и вторым окрашиванием алгоритм окраски захватил замкнутый маршрут с началом и концом x_i. В силу четности его длины оба раза x_i окрашена в один цвет (α или β). ■

3. Охарактеризовать структуру n-хроматических графов при $n \geq 3$ подобно тому, как это делает теорема Кенига для бихроматического графа, пока не удается. Имеется много частных результатов, например, для некоторых классов регулярных графов, плоских графов с треугольными ячейками (плоских триангуляций). Отметим, что хотя плоский (иначе — планарный) граф имеет хроматическое число $\gamma \leq 4$, планарность не является характеристическим признаком 4-хроматичности графа: $K_{3,3}$ (рис. 6.12) бихроматичен ($\gamma = 2$), но не планарен.

Для получения общих оценок хроматического числа удобно ввести несколько определений. ***Клика*** — любой полный подграф графа G; ***кликовое число*** (или ***плотность***) $\rho(G)$ — число вершин самой мощной клики. ***Независимое множество*** — всякая совокупность попарно несмежных вершин; ***число независимости*** $\alpha(G)$ — число вершин самого мощного множества независимости. ***Дополнительным к графу*** $G = (X, Y)$ называется граф $\tilde{G} = (X, \tilde{Y})$ с тем же множеством вершин, дополняющий G до полного графа

$$G_0 = G + \tilde{G} = (X, Y \cup \tilde{Y}).$$

Примеры клик графа G (рис. 6.19) изображены на рис. 6.20, дополнительный граф $\tilde{G}$ — на рис. 6.21.

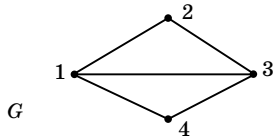


Рис. 6.19. Связный граф G

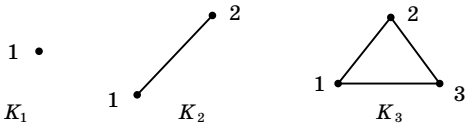


Рис. 6.20. Клики графа G

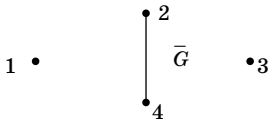


Рис. 6.21. Дополнительный граф $\bar{G}$

Поскольку всякое множество попарно смежных вершин при раскраске требует красок по числу вершин, кликовое число является нижней оценкой для хроматического числа (6.2,*а*). Вторая оценка (6.2,*б*) следует из того, что α совпадает с мощностью p_{i_0} наибольшего множества вершин, допускающего окрашивание одним цветом:

$$а)\ \gamma(G) \ge \rho(G); \qquad б)\ \gamma(\mathrm{G}) \ge \left\lceil \frac{|X|}{\alpha(G)} \right\rceil. \tag{6.2}$$

Действительно, пусть фиксирована некоторая оптимальная раскраска, V_i — одноцветные множества вершин, $|V_i| = p_i$. По построению $\bigcup\limits_{i=1}^{\gamma} V_i = X$, $\sum\limits_{i=1}^{\gamma} p_i = |X|$, и каждое V_i есть независимое множество. Деление последнего равенства на α с учетом неравенств $p_i \le \alpha$ приводит к нужной оценке:

$$\gamma \ge \sum_{i=1}^{\gamma} \frac{p_i}{\alpha} = \frac{|X|}{\alpha} \ge \left\lceil \frac{|X|}{\alpha} \right\rceil.$$

Отметим без доказательства более грубую, чем (6.2,*а*), но зато легче вычислимую оценку (Геллер Д. П., 1970 г.):

$$\gamma(G) \ge \frac{n^2}{n^2 - 2m}, \quad n = |X|,\ m = |Y|.$$

Интересны оценки суммы и произведения хроматических чисел γ и $\tilde{\gamma}$ графа G и его дополнения $\bar{G}$, доказанные Нордхаузом Е. А. и Гаддумом Ж. В. в 1956 г.:

$$2\sqrt{n} \le \gamma + \tilde{\gamma} \le n+1; \quad n \le \gamma\,\tilde{\gamma} \le \left(\frac{n+1}{2}\right)^2. \tag{6.3}$$

Очевидна также оценка сверху: $\gamma \le n + 1 - \alpha$. Пусть $\delta = \delta(G)$ наименьшая из степеней вершин графа G, $\Delta = \Delta(G)$ — наибольшая. Справедливы следующие изящные утверждения.

1°. **Теорема Брукса** (1941 г.).

Если G — неполный связный граф и $\Delta(G) \ge 3$, то $\gamma(G) \le \Delta(G)$.

2°. *Для любых натуральных чисел n, α, γ, удовлетворяющих неравенствам $n/2 \le \gamma \le n + 1 - \alpha$, существует n-вершинный граф с числом независимости α и хроматическим числом γ.*

4. Задачу раскраски вершин графа G можно переформулировать в виде некоторой задачи булевого программирования. Пусть $r \ge \gamma(G)$ — какая-нибудь верхняя оценка хроматического числа так, что граф G r-раскрашиваем. Введем матрицу булевых переменных раскраски $T = \{t_{ij}\}$, $i = 1, \ldots, r$; $j = 1, \ldots, n$; $n = |X|$, где для каждой

раскраски $t_{ij} = 1$, если вершина x_j окрашена в i-й цвет, и $t_{ij} = 0$ — в противном случае. Если $R = \{r_{ij}\}$ — матрица инциденций неориентированного графа G, то для произведения матриц TR элемент $(T \cdot R)_{ij} = \sum_{k=1}^{n} t_{ik} r_{kj}$ равен: 0, если ни один из концов ребра j не окрашен в i-й цвет; 1, если точно один конец ребра j окрашен в i-й цвет; 2, если оба конца ребра j окрашены в i-й цвет. Последняя ситуация возможна только при неправильной раскраске, поэтому условие правильности раскраски выражается системой $r \cdot m$ неравенств:

$$(T \cdot R)_{ij} \le 1; \quad 1 \le i \le r; \quad 1 \le j \le m = |Y|. \tag{6.4}$$

Условием окрашенности каждой вершины x_j в один цвет есть

$$\sum_{i=1}^{r} t_{ij} = 1, \quad 1 \le j \le n. \tag{6.5}$$

Любая булевская матрица T, удовлетворяющая ограничениям (6.4, 6.5), обеспечивает r-раскраску G. Чтобы получить стандартную оптимизационную задачу, необходимо добавить функцию цели, например, минимизировать частоту использования последнего (r-го) цвета:

$$\rho_r(T) = \sum_{j=1}^{n} t_{rj} \to \min. \tag{6.6}$$

Всякое решение T_0 задачи линейного булевского программирования (6.4, 6.5, 6.6) дает правильную раскраску G, притом с помощью $(r-1)$ красок, если $\gamma(G) < r$. В этом случае $\rho_r(T_0) = 0$ и можно уменьшить число строк матрицы раскраски T на одну, заменить в (6.4, 6.5, 6.6) r на $r-1$ и решить задачу заново. Продолжая такой процесс, мы придем к моменту, когда для решения T очередной уменьшенной задачи впервые получится ненулевое значение целевой формы: $\rho_S(T) \ne 0$. Это будет означать, что $\gamma(G) = S$.

Существует способ задания целевой формы, сразу приводящий к раскраске с наименьшим числом красок $\gamma(G)$. Сопоставим каждому цвету i штраф p_i так, чтобы $p_{i+1} > np_i$ ($n = |X|$). Тогда целевая форма

$$Z(T) = \sum_{i=1}^{r} \sum_{j=1}^{n} p_i t_{ij} \to \min \tag{6.7}$$

с ограничениями (6.4, 6.5) обеспечивает раскраску T_0 с помощью $\gamma(G)$ красок. Иными словами, булевское решение T_0 задачи (6.4, 6.5, 6.6) имеет нулевые строки $\gamma + 1$, ..., r.

Действительно, нулевая i-я строка у T_0 означает, что i-й цвет не используется, и если используется цвет $i + m$, то перестановка

двух строк i, $i + m$ в T_0 приведет к раскраске $\overline{T}_0$ с меньшей формой (2.14): $Z(\overline{T}_0) < Z(T_0)$, что невозможно. Поэтому в T_0 нулевые строки стоят последними. Далее пусть r_0 — наибольший номер ненулевых строк. Тогда $r_0 \geq \gamma = \gamma(G)$, так как для всякой правильной раскраски требуется не менее γ красок. Предположим, $r_0 \geq \gamma + 1$ и $\overline{T}$ — матрица правильной раскраски с γ красками (хотя бы одна такая раскраска существует и удовлетворяет условиям (6.4, 6.5)). Не уменьшая общности, можно считать, что все нулевые строки $\overline{T}$ имеют номера $\gamma + 1$, ..., r. Очевидно,

$$Z(\overline{T}) = \sum_{i}^{\gamma}\sum_{j}^{n} p_i \bar{t}_{ij} \leq p_\gamma \sum_{j}^{n}\sum_{i}^{\gamma} \bar{t}_{ij} = p_\gamma \sum_{j}^{n} 1 = np_\gamma,$$

$$Z(T_0) = \sum_{i=1}^{r_0}\sum_{j=1}^{n} p_i t_{ij}^0 \geq p_{r_0} \geq p_{\gamma+1}.$$

По условию оптимальности решения, T_0 должно быть $Z(T_0) \leq Z(T)$, откуда $p\gamma_{+1} \leq np\gamma$. Последнее неравенство противоречит условию выбора штрафных коэффициентов, поэтому $r_0 = \gamma$.

В зависимости от ситуации, вместо $r \cdot m$ условий (6.4) можно использовать $r \cdot n$ условий

$$L(1 - t_{ij}) - \sum_{k=1}^{n} t_{ik} a_{kj} \geq 0; \quad 1 \leq i \leq r; \quad 1 \leq j \leq n; \qquad (6.8)$$

где L — любое число, большее n; $a_{k,j}$ — элементы матрицы смежности неорграфа G. При этом, не уменьшая общности, можно считать, что $0 \leq a_{k,j} \leq 1$, ибо кратные ребра можно убрать из G, не меняя задачи раскраски вершин.

Заметим, что при больших n и r в целевой форме штрафные коэффициенты имеют слишком большой разброс по порядку значений; это создает неудобства для вычислений и требует особых приемов для сохранения надлежащей точности. В сочетании с принципиальными трудностями точного решения задачи булевского программирования высокой размерности мы приходим к неутешительному выводу: для больших графов раскраска и нахождение хроматического числа методом сведения к задаче булевского программирования практически неэффективны. Следует использовать специфические алгоритмы, учитывающие особенности геометрии графа. Более того, если поставленная перед вычислителем задача булевского программирования допускает интерпретацию как задача раскраски некоторого графа G, то при высоких размерностях эффективнее искать именно оптимальную раскраску графа, а уже с ее помощью записывать решение исходной задачи программирования.

Для раскраски не очень больших графов применяются алгоритмы перебора.

Для раскраски больших графов, когда точные алгоритмы оказываются недопустимо трудоемкими, используются алгоритмы приближенной раскраски, семейство которых весьма многочисленно. Среди них имеются эвристические и случайно-поисковые алгоритмы. При их использовании следует проявлять известную осторожность (например, сравнивать результаты решения по нескольким принципиально различным алгоритмам), ибо для каждого алгоритма, как правило, можно построить граф, для которого алгоритм дает произвольно плохую оценку хроматического числа.

5. Прикладные задачи, связанные с раскрасками, могут быть в точности эквивалентными задаче раскраски, но чаще содержат еще дополнительные ограничения.

5.1. *Задача загрузки* (*размещения*) *n продуктов* (предметов) по ящикам (хранилищам). Модельный граф G имеет n вершин (отвечающих продуктам), а наличие ребра (x_i, x_j) означает, что продукт x_i несовместим с x_j. Если вместимости Q_i ящиков велики ($Q_i \geq n$), то задача размещения в наименьшее число ящиков эквивалентна задаче оптимальной раскраски вершин графа G. При ограниченных вместимостях Q_i к обычным условиям раскраски в обозначениях п. 6.4 добавятся ограничения

$$\sum_{j=1}^{n} t_{ij} \leq Q_i. \qquad (6.9)$$

5.2. В задачах *теории расписаний* (иначе, *календарного планирования*) операциям (осмотрам) сопоставляются временные интервалы. Отнесем им вершины x_i модельного графа G, у которого ребро (x_i, x_j) есть тогда и только тогда, когда операции x_i, x_j несовместимы во времени. При одинаковой длительности и произвольной очередности операций оптимальное расписание (минимизирующее суммарное время выполнения всех работ) эквивалентно оптимальной раскраске модельного графа G. Хроматическое число равно оптимальному времени выполнения всех работ: $\gamma(G) = T_{\min}$. Кроме того, различные цвета могут отвечать, например, различным участкам (помещениям), и если на i-м участке нельзя выполнять более Q_i операций одновременно, то в модельной раскраске появляется дополнительное ограничение (6.9).

Типичной задачей этого типа является задача составления расписания занятий. Пусть, например, требуется прочесть несколько лекций (каждая по одному часу) за кратчайшее время. Число

групп слушателей равно числу лекций, но все же некоторые лекции нельзя читать одновременно (например, их читает один и тот же лектор). Оптимальное расписание эквивалентно минимальной раскраске такого графа G, у которого вершины отвечают лекциям, а смежность вершин означает, что соответствующие лекции нельзя читать одновременно.

Нетрудно скорректировать любой из алгоритмов оптимальной раскраски так, чтобы учитывались ограничения (6.9).

Задания

1. Доказать, что число вершин n регулярного бихроматического графа четно, а множества одноцветности X_1, X_2 (рис. 6.18) имеют одинаковые числа вершин:
   $|X_1| = |X_2| = \alpha(G) = n/2$.
2. Доказать, что:
   а) для правильной раскраски карты, получающейся при пересечении окружностей на плоскости, достаточно двух цветов;
   б) всякая карта, не имеющая вершин степени 2, имеет грань, граница которой содержит не более пяти ребер.

6.8. Деревья

Определение

Связный граф T без циклов называется ***деревом.*** Ориентация может не учитываться, и тогда говорят о ***ребрах дерева*** y_j. Если ориентация учитывается, то говорят о ***дугах дерева,*** а само дерево называется ***ориентированным***, например, дерево логических возможностей, генеалогическое дерево.

Помимо приведенного определения можно дать еще несколько эквивалентных определений дерева.

Теорема 6.7

Для графа G с n вершинами и m ребрами равносильны следующие свойства:

1. *G связен и не имеет циклов.*
2. *G связен, и число его вершин на единицу превосходит число ребер: $n = m + 1$.*
3. *G не содержит циклов и $n = m + 1$.*
4. *G не содержит циклов, но добавление ребра между любыми двумя несмежными вершинами приводит к появлению цикла.*
5. *G связен, но теряет это свойство при удалении любого ребра.*
6. *Любые две вершины $x_i \neq x_k$ графа G соединяет единственная цепь.*

Если одно из свойств 2–6 выполнено, то граф G есть дерево.

Доказательство теоремы 6.7 или ее частей содержится во многих книгах по теории графов. Поскольку оно не требует специальной техники и достаточно элементарно, предоставим его читателю в качестве упражнения.

Следствие 6.3. *Всякое дерево с числом вершин $n \geq 2$ имеет, по меньшей мере, две висячие (концевые) вершины.*

Действительно, по лемме о рукопожатиях (п. 6.4) сумма степеней вершин есть $\sum_{i=1}^{n} \delta(x_i) = 2(n-1)$. Поскольку $\delta(x_i) \geq 1$ (изолированные вершины отсутствуют), то хотя бы две вершины x_1, x_2 имеют степень $\delta(x_1) = \delta(x_2) = 1$, в противном случае $\delta(x_i) \geq 2$ для всех i, так что

$$\sum_{i=1}^{n} \delta(x_i) \geq 2n. \qquad \blacksquare$$

В разделе 6.2 на рис. 6.6 был приведен пример дерева G с числом вершин $n = 21$, в котором центр состоял из двух вершин $\{u, v\}$. Для простой цепи P_n центр состоит из одной вершины, если число вершин n нечетно, и соответственно — из двух вершин, если n четно (рис. 6.22).

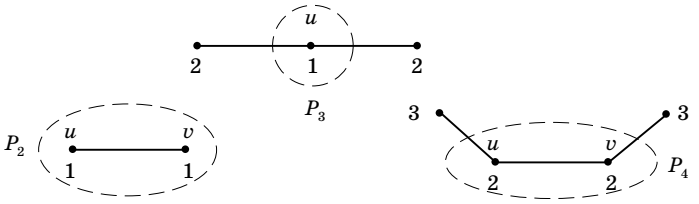


Рис. 6.22. Эксцентриситеты вершин и центры цепей P_2, P_3, P_4

Оказывается, для любого дерева справедлива.

Теорема 6.8

Центр любого дерева состоит из одной или двух смежных вершин.

□ Для деревьев P_2, P_3 с числом вершин $n = 2$, $n = 3$ утверждение справедливо (см. рис. 6.22). Пусть T — дерево с n вершинами, $n > 3$. Если удалить из T все висячие ребра с концевыми вершинами, то полученное дерево $T_0 (\subset T)$ имеет тот же центр, что и дерево T. Теперь теорема выводится по индукции. ■

Перечисление графов и деревьев: неизоморфных, корневых и помеченных

Задачи перечисления всех «различных» (в том или ином смысле) графов с заданным числом вершин n являются трудными. Они возникали и решались в чистом виде или с дополнительными ограничениями в математике, химии, физике, проектировании и различных инженерных проблемах. Понятно, что перечисление деревьев с n вершинами образует, с одной стороны, более простой, с другой стороны, — базовый класс задач.

Граф $G = (X, Y)$ называется ***помеченным***, если его вершинам присвоены фиксированные *метки*, например, номера 1, 2, ..., n. Два помеченных графа ***одинаковы (не различаются)***, если их вершины помечены одной системой меток и существует изоморфизм (см. п. 10.4) одного графа на другой, при котором сохраняются метки всех вершин. На рис. 6.23 изображены все помеченные графы с числом вершин $n = 3$; их количество равно 8.

Количество G_{nm} (неориентированных) помеченных (n, m) — графов (простых с n вершинами и m ребрами) равно, очевидно, числу сочетаний из множества различных неориентированных пар вершин $\{(x_i, x_j)\}$ по числу ребер m. Так как число указанных пар вершин равно числу сочетаний C_n^2 (из n по 2), то (см. п. 10.3)

$$G_{nm} = C^m_{C_n^2} = C_s^m, \quad S = C_n^2 = \frac{n(n-1)}{2}. \qquad (6.11)$$

Суммируя числа G_{nm} по всем возможным количествам ребер $m = 1, 2, ..., \frac{n(n-1)}{2}$ от случая безреберного графа до случая полного графа с n вершинами, получаем *число G_n всех помеченных графов с n вершинами*:

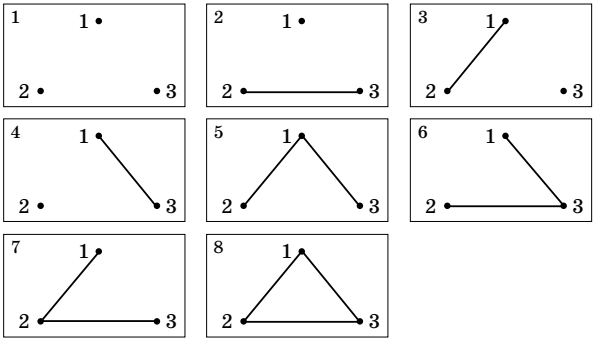


Рис. 6.23. Помеченные графы с тремя вершинами

$$G_n = \sum_{m=0}^{s} C_s^m = 2^s, \quad S = C_n^2 = \frac{n(n-1)}{2}\ldots \qquad (6.12)$$

Представление суммы в (6.12) в виде $2^{C_n^2}$ следует из формулы (10.14). В частности, $G_3 = 2^{C_3^2} = 2^3 = 8$ (см. рис. 6.23), $G_4 = 2^{C_4^2} = 2^{\frac{4\cdot3}{1\cdot2}} = 2^6 = 64$.

Число g_n неизоморфных графов без пометок (простых, неориентированных) найти значительно труднее. Среди восьми графов рис. 6.23 без учета пометок можно указать только четыре попарно неизоморфных друг другу, например, в квадратах 1, 2, 7, 8. Следовательно, $g_3 = 4$, что вдвое меньше числа G_3 помеченных графов. Число G_4 помеченных четырехвершинных графов равно 64, в то время, как различных неизоморфных четырехвершинных графов без пометок существует всего $11 = g_4$ (см. рис. 6.24).

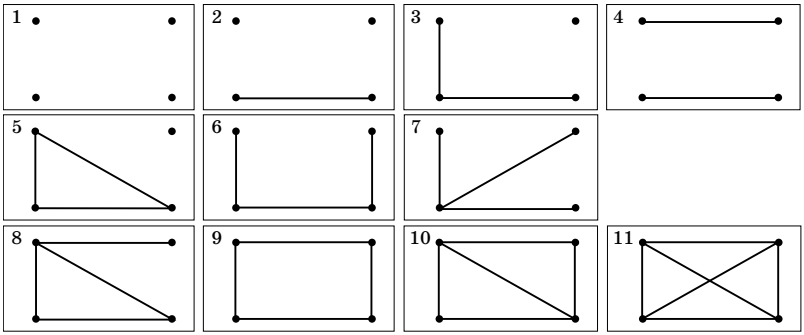


Рис. 6.24. Четырехвершинные неизоморфные графы

Если через g_{nm} обозначить *число неизоморфных n-вершинных графов с m ребрами*, то

$$g_n = \sum_{m=0}^{s} g_{nm}, \quad s = C_n^2 = \frac{n(n-1)}{2}. \qquad (6.13)$$

Согласно рис. 6.24, для $n = 4$ имеем: $s = 6$; $g_{4,0} = 1$ (квадрат 1), $g_{4,1} = 1$ (квадрат 2), $g_{4,2} = 2$ (квадраты 3, 4), $g_{4,3} = 3$ (квадраты 5, 6, 7), $g_{4,4} = 4$ (квадраты 8, 9), $g_{4,5} = 1$ (квадрат 10), $g_{4,6} = 1$ (квадрат 11). По формуле (6.13) $g_4 = 1 + 1 + 2 + 3 + 2 + 1 + 1 = 11$.

В общем случае количество неизоморфных графов g_{nm}, g_n находятся с помощью теории перечисления конфигураций, созданной Д. Пойа. С ее помощью подсчитано, например, что $g_{9,5} = 25$, $g_{9,7} = 148$, $g_{9,9} = 771$, $g_{9,15} = 21933$, $g_{9,18} = 34040$; и наконец, число всех графов с 9-ю вершинами равно $G_9 = 308168$.

Перейдем к *перечислению деревьев*, обозначив через τ_n *число помеченных деревьев* и через t_n *число обычных*[1] *(неизоморфных) деревьев* с n вершинами. На рис. 6.23 (квадраты 5, 6, 7) видно, что $\tau_3 = 3$, $t_3 = 1$. Далее $t_4 = 2$ (рис. 6.24, квадраты 6, 7) и как видно из рис. 6.25, число помеченных четырехвершинных деревьев равно $16 = \tau_4$.

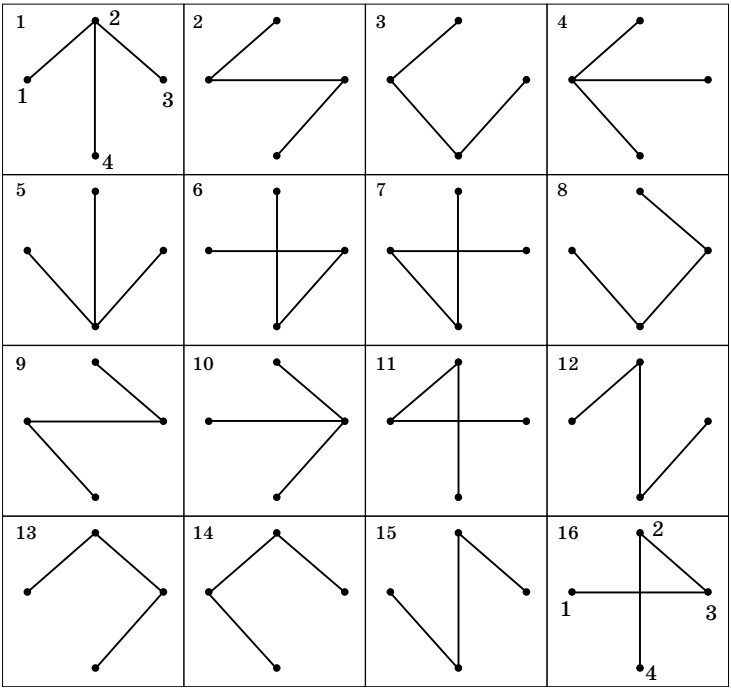


Рис. 6.25. Помеченные четырехвершинные

Формула А. Кэли (1897 г.). *Число* τ_n *помеченных деревьев с* n *вершинами равно* n^{n-2}

$$\tau_n = n^{n-2}. \qquad (6.14)$$

Доказательство приводится ниже (см. следствие 6.4 из теоремы 6.9).

Например, $\tau_2 = 2^{2-2} = 1$, $\tau_3 = 3^{3-2} = 3$, $\tau_4 = 4^{4-2} = 16$ (рис. 6.25), $\tau_5 = 5^{5-2} = 125$, $\tau_6 = 6^4 = 1296$, $\tau_7 = 7^5 = 16807$.

[1] Деревья без каких-либо пометок иногда называются *свободными деревьями*.

Числа t_n обычных (неизоморфных) деревьев являются значительно меньшими, однако вычислить их существенно труднее. Современные алгоритмы нахождения значений t_n и получения конфигураций неизоморфных деревьев с n вершинами основаны на теории перечисления Д. Пойа.

Важный класс графов образуют деревья с одной помеченной вершиной, которая называется ***корнем***. Само дерево с одной помеченной («выделенной») вершиной называется ***корневым деревом***. Число корневых деревьев с n вершинами обозначается через T_n; ясно, что

$$t_n \le T_n \le \tau_n. \qquad (6.15)$$

Все корневые деревья с числом вершин $n = 3$ и $n = 4$ изображены на рис. 6.26.

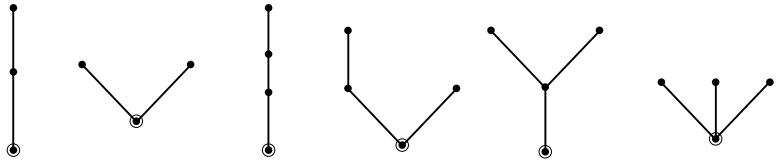


Рис. 6.26. Корневые деревья с тремя и четырьмя вершинами

В таблице 6.1 приведены значения чисел t_n неизоморфных деревьев, T_n корневых деревьев и τ_n помеченных деревьев с числами вершин n от 1 до 10.

Таблица 6.1. Значения t_n, T_n, τ_n

Число вершин n	1	2	3	4	5	6	7	8	9	10
Число деревьев t_n	1	1	1	2	3	6	11	23	47	106
Число корневых деревьев T_n	1	1	2	4	9	20	48	115	286	719
Число помеченных деревьев τ_n	1	1	3	16	125	1296	16897	262144	4782969	10^8

На рис. 6.27 изображены все неизоморфные деревья с пятью вершинами ($t_5 = 3$).

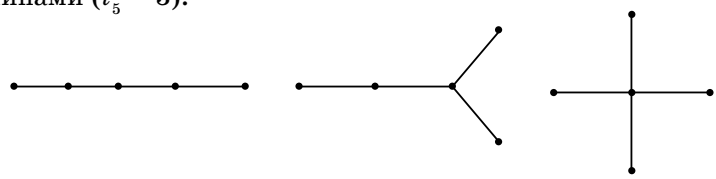


Рис. 6.27. Неизоморфные деревья с пятью вершинами

Остовы (каркасы) графа

Остовным подграфом (кратко — ***остовом***) или ***каркасом графа*** $G = (X, Y)$ называется подграф $H = (X, Y_H)$ *без циклов*, содержащий *все вершины* X и *максимально большое число ребер*. Если граф G — связный, то остов H есть *дерево*, если G содержит p компонент связности, то остов (каркас) H состоит из p *деревьев*. Два остова в G считаются различными, если они отличаются хотя бы одним ребром, то есть как графы. Для того чтобы G имел более одного остова, необходимо и достаточно существование хотя бы одного цикла в G.

Задача определения *числа всех остовов графа* G и их фактического конструирования является важной для приложений. Понятно, ее достаточно решить для связного G. Для *полного* (*простого*) *графа* G *перечисление остовов есть перечисление всех помеченных деревьев* с $n = |X|$ вершинами, так что число остовов равно n^{n-2} (см. (6.13), теорема Кэли). В общем случае число остовов получил Кирхгоф в 1847 г. ***Матрицей Кирхгофа*** $K(G)$ простого графа G называется $n \times n$ — матрица с элементами

$$k_{i,j} = \begin{cases} -1, \text{если вершины } x_i,\ x_j \text{ смежны;} \\ 0, \text{если вершины } x_i,\ x_j \text{ не смежны;} \\ \delta_i = \deg x_i, \text{ если } i = j. \end{cases}$$

Сумма элементов в каждой строке (и каждом столбце) матрицы $K(G)$ *равна нулю. У любой квадратной матрицы с таким свойством строк и столбцов, а не только у матрицы Кирхгофа* $K(G)$ *алгебраические дополнения* Δ_{ij} *всех элементов* κ_{ij} *равны между собой:* $\Delta_{ij} = \Delta$.

У матрицы же $K(G)$ величина Δ равна числу остовных деревьев.

Теорема 6.9 *(Кирхгоф)*

Число остовных деревьев в простом связном графе G *с* n *вершинами* ($n \geq 2$) *равно алгебраическому дополнению любого элемента матрицы Кирхгофа* $K(G)$.

Доказательство опирается на выражение определителя по формуле Бинэ — Коши.

Следствие 6.4. *Справедлива формула Кэли (6.14) для числа помеченных деревьев с* n *вершинами* $\tau_n = n^{n-2}$:

□ Все помеченные деревья с n вершинами образуют в точности множество остовных деревьев *полного* графа K_n. По теореме 6.9

их число равно алгебраическому дополнению Δ_{11} элемента κ_{11} матрицы Кирхгофа

$$K(K_n) = \begin{pmatrix} n-1 & -1 & -1 & \dots & -1 \\ -1 & n-1 & -1 & \dots & -1 \\ . & . & . & \dots & . \\ -1 & -1 & -1 & \dots & n-1 \end{pmatrix}$$

размерности $n \times n$. Определитель порядка $(n-1)$

$$\Delta_{11} = \begin{vmatrix} n-1 & -1 & -1 & \dots & -1 \\ -1 & n-1 & -1 & \dots & -1 \\ \dots & \dots & \dots & \dots & \dots \\ -1 & -1 & -1 & \dots & n-1 \end{vmatrix}$$

удобно преобразовать, заменив сначала первую строку суммой всех строк, а затем, прибавляя полученную строку из единиц к строкам с номерами 2, 3, ..., $n-1$:

$$\Delta_{11} = \begin{vmatrix} 1 & 1 & 1 & \dots & 1 \\ . & . & . & \dots & . \\ -1 & n-1 & -1 & \dots & -1 \\ . & . & . & \dots & . \\ -1 & -1 & -1 & \dots & n-1 \end{vmatrix} = \begin{vmatrix} 1 & . & 1 & 1 & \dots & 1 \\ . & . & . & . & \dots & . \\ 0 & . & n & 0 & \dots & 0 \\ . & . & . & . & \dots & . \\ 0 & . & 0 & 0 & \dots & n \end{vmatrix} = n^{n-2}. \quad \blacksquare$$

Алгоритм построения остовного дерева. Поиск в глубину

Задача построения какого-либо (одного!) остовного дерева в графе $G = (X, Y)$ относится к числу важнейших для практики и, к счастью, является простейшей для алгоритмизации и компьютерной реализации. Понятно, что основная идея построения должна состоять в последовательном отборе ребер, не образующих циклы с предыдущими. Предположим, построение остовного дерева $T = (X, Y_T)$ закончено. Тогда в его процессе были просмотрены все n вершин и отобраны $(n-1)$ ребро графа G, $n-1 = |Y_T|$. Неясно, сколько всего ребер потребуется просмотреть, отбирая ребра Y_T без циклов; возможно, в худшем случае все ребра множества Y.

Учитывая последнее замечание, мы сначала приведем так называемый алгоритм «поиска в глубину», просматривающий по одному разу все ребра графа G, и, разумеется, все его вершины. Заметим,

что поиск в глубину (сокращенно ПГ) — не единственный метод просмотра всех вершин и ребер графа G. Например, часто используется просмотр графа «поиском в ширину», при котором в каждой очередной вершине просматриваются все инцидентные ей ребра без исключения и все их концевые вершины (то есть «окружение» вершины X). Метод поиска в ширину фактически использовался нами в разделе 6.7 при раскраске бихроматического графа в доказательстве теоремы Кенига.

Качественное описание алгоритма поиска в глубину, обозначения

В процессе поиска в глубину вершинам графа G последовательно присваиваются новые номера (ПГ-номера) от 1 до n, а ребра получают метки двух классов: «прямое ребро» и «обратное ребро». Поиск начинается с произвольной вершины $\nu_0 \in X$, которой присваивается ПГ-номер1: ПГ $(\nu_0) = 1$. Далее выбирается любое инцидентное к ν_0 ребро (ν_0, ν), помечается меткой «прямое ребро» и вершине ν присваивается ПГ-номер 2. Следующий (третий) шаг поиска начинается в вершине ν, в которой ПГ $(\nu) = 2$, и подчиняется реккурентной процедуре, которую мы опишем для произвольного шага $(k + 1)$. Предположим, k-ый шаг выполнен и некоторая вершина x получила ПГ-номер ПГ $(x) = k$.

Возможны две ситуации:

1. В вершине x существует инцидентное непомеченное ребро (x, w). Если вершина w уже имеет ПГ-номер, то ребро (x, w) получает метку «обратное», и продолжается поиск непомеченного ребра, инцидентного вершине x. Если же вершина w не имеет ранее присвоенного ПГ-номера, то ей присваивается очередной ПГ-номер ПГ $(w) = k + 1$, ребро (x, w) получает метку «прямое», и поиск перемещается в вершину w.
2. Все ребра, инцидентные вершине x, помечены на предыдущих шагах. Тогда поиск возвращается в вершину $\overline{x}$, имеющую предшествующий ПГ-номер ПГ $(\overline{x}) = k - 1$.

Поиск в глубину заканчивается, когда все ребра графа G окажутся помеченными.

Обозначим Y_+ — множество прямых ребер, Y_- — множество обратных ребер. Из процедуры поиска в глубину следует утверждение.

Утверждение. Если $G = (X, Y)$ — связный граф, то подграф $T = (X, Y_+)$ есть остовное дерево в G, а подграф (X, Y_-) является

дополнительным к дереву T. Дерево T является ориентированным корневым с корнем v_0.

Перейдем к формализованному описанию алгоритма поиска в глубину. Предварительно информацию о графе G удобно представить в виде списков смежности $\{N_x\}_{x \in X}$, где N_x — список вершин w, инцидентных вершине x. В процессе поиска формируется динамический список S вершин, в который каждая вершина x включается в точности один раз в момент присвоения ей ПГ-номера, и, возможно, исключается из S в момент возврата из x в вершину с предыдущим ПГ-номером. Список S организуется как стек, в котором включение и исключение вершин производится справа. Используются следующие переменные:

k — последний присвоенный ПГ-номер;

p — указатель конца стека S, так что $S(p)$ — имя последней вершины стека S. Алгоритм формирует три расширяющихся списка: список вершин SПГ, получивших к данному моменту ПГ-номера; список «прямых» ребер SY_+; список SY_- «обратных» ребер.

Алгоритм поиска в глубину в неориентированном связном графе $G = (X, Y)$

1. Выбрать произвольную вершину $v_0 \in X$. Положить ПГ(v_0):= 1, $S(1) := v_0$, SПГ:= $\{v_0\}$, $SY_+ := \{\varnothing\}$, $SY_- := \{\varnothing\}$.
2. Выбрать исследуемую вершину $x := S(p)$.
3. Просмотр списка смежности N_x. Найти такую вершину $w \in N_x$, что ребро (x, w) не помечено, и перейти к шагу 4. Если такой вершины в списке N_x нет, то перейти к шагу 5.
4. 4.1. Если вершина w уже имеет ПГ-номер, то пометить ребро (x, w) меткой «обратное» и занести в конец списка SY_-. Перейти к шагу 3 и продолжить просмотр списка смежности N_x.
   4.2. Если вершина w еще не имеет ПГ-номера, то присвоить ей очередной ПГ-номер: $k := k + 1$, ПГ $(w) = k + 1$. Ребро (x, w) снабдить меткой «прямое» и занести в конец списка SY_+; положить $p := p + 1$, $S(p) := w$. Перейти к шагу 2.
5. Если для просматриваемого на шаге 3 списка смежности N_x каждая вершина $w \in N_x$ соединяется с вершиной x уже помеченным ребром (x, w), то вернуться от вершины x к вершине с предыдущим ПГ-номером и положить:

   $p := p - 1$ (удаление вершины x из стека S).

   Если $p = 0$, поиск в глубину заканчивается. Если $p \neq 0$, перейти к шагу 2.

6. Поиск в глубину заканчивается, когда все ребра Y получили метки, то есть когда $SY_+ \cup SY_- = Y$. Тогда

$$Y_+ = SY_+,\ Y_- = SY_-,\ S\Pi\Gamma = X,\ T = (X, Y_+) —$$

остовное дерево графа G.

Замечание. Остовное дерево T могло быть сформировано, вообще говоря, до окончания просмотра всех ребер Y, именно, в тот момент, когда число прямых ребер достигло значения $n - 1$: $|SY_+| = n - 1$.

Какова *трудоемкость алгоритма поиска в глубину*? Поскольку каждая из n вершин графа G не более чем два раза обрабатывается в стеке S (один раз включается и, возможно, один раз исключается), то работа со стеком занимает $\theta(n)$ времени[1].

Суммарное выполнение шага 4 алгоритма требует $\theta(m)$ времени, где m — число ребер графа G (см. раздел 8). Суммарное выполнение шага 3 также можно ограничить временем $\theta(m)$: достаточно организовать дополнительный массив с переменной $q = q(x)$, указывающей номер первой непросмотренной вершины в списке смежности N_x, и начинать очередной просмотр на шаге 3 с вершины $q(x)$. Следовательно, трудоемкость всего алгоритма составляет $\theta(n + m)$. Это наилучший возможный результат, так как все $n + m$ вершин и ребер должны быть просмотрены. О таких алгоритмах говорят, что они имеют *линейную сложность*: в оценке времени для наихудшего варианта вычислений число всех «входных данных» алгоритма (здесь это $n + m$) умножается на постоянную, не зависящую от количества данных.

Проиллюстрируем работу алгоритма на графе (рис. 6.28). Для удобства первоначальные имена вершин не приводятся, а указываются уже их ПГ-номера, которые получаются последовательно в процессе работы алгоритма. Прямые ребра помечаются индексом «n» и одной стрелкой на сплошной линии, обратные ребра — индексом «o» и стрелками на пунктирной линии. Номера ребер присваиваются в порядке их прохождения. В таблице 6.2

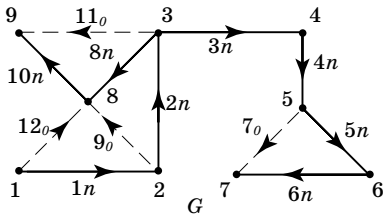


Рис. 6.28. Алгоритм поиска в глубину

[1] То есть при больших n затраты времени на стек s не более чем cn, где постоянная c не зависит от n.

приведены 15 шагов поиска в глубину, на каждом шаге указывается полностью весь стек вершин S, весь массив данных SY_+ весь массив SY_- и пометка нового ребра — если таковое имеется. Остановка — после шага 15. В строке 15 массив SY_+ дает все множество Y_+ ребер остовного дерева T, массив SY_- — множество Y_- ребер кодерева, хотя фактически Y_+ было получено еще на шаге 13. Граф задан списками смежности N_k — множествами вершин, смежными k-ой вершине: ($k = 1, 2, ..., 9$): $N_1 = \{2, 8\}$, $N_2 = \{1, 3, 8\}$, $N_3 = \{2, 4, 8, 9\}$, $N_4 = \{3, 5\}$, $N_5 = \{4, 6, 7\}$, $N_6 = \{5, 7\}$, $N_7 = \{5, 6\}$, $N_8 = \{1, 2, 3, 9\}$, $N_9 = \{3, 8\}$.

Таблица 6.2. Поиск в глубину для графа G рис. 6.28

№	S — стек вершин	Метка ребра	Массив SY_+	Массив SY_-
1	1	$1n$	$1n$	–
2	1, 2	$2n$	$1, 2 - n$	–
3	1, 2, 3	$3n$	$1, 2, 3 - n$	–
4	1, 2, 3, 4	$4n$	$1, 2, 3, 4 - n$	–
5	1, 2, 3, 4, 5	$5n$	$1, 2, 3, 4, 5 - n$	–
6	1, 2, 3, 4, 5, 6	$6n$	$1, 2, 3, 4, 5, 6 - n$	–
7	1, 2, 3, 4, 5, 6, 7	7_0	$1, 2, 3, 4, 5, 6 - n$	7_0
8	1, 2, 3, 4, 5, 6	–	$1, 2, 3, 4, 5, 6 - n$	7_0
9	1, 2, 3, 4, 5	–	$1, 2, 3, 4, 5, 6 - n$	7_0
10	1, 2, 3, 4	–	$1, 2, 3, 4, 5, 6 - n$	7_0
11	1, 2, 3	$8n$	$1, 2, 3, 4, 5, 6, 8 - n$	7_0
12	1, 2, 3, 8	9_0	$1, 2, 3, 4, 5, 6, 8 - n$	$7_0, 9_0$
13	1, 2, 3, 8	$10n$	$1, 2, 3, 4, 5, 6, 8, 10 - n$	$7_0, 9_0$
14	1, 2, 3, 8, 9	11_0	$1, 2, 3, 4, 5, 6, 8, 10 - n$	$7_0, 9_0, 11_0$
15	1, 2, 3, 8	12_0	$1, 2, 3, 4, 5, 6, 8, 10 - n$	$7_0, 9_0, 11_0, 12_0$
	Останов			

Замечание. Если исходный граф G не является связным, то алгоритм поиска в глубину просмотрит только одну из компонент графа и построит остовное дерево этой компоненты. Понятно, как модифицировать алгоритм, чтобы поиск перешел к новой компоненте связности и далее просмотрел последовательно все компоненты, построив, в частности, остовные деревья во всех компонентах, то есть *остовный лес графа G*.

Алгоритм построения остовного дерева путем произвольного просмотра ребер

Рассмотрим еще один алгоритм построения остовного дерева, в котором ребра графа G просматриваются в произвольном порядке. В каждом шаге алгоритма множество ребер Y разбивается на три подмножества: Y_α — окрашенные α-цветом, Y_β — окрашенные β-цветом, Y_0 — неокрашенные. До начала работы алгоритма $Y_\alpha = \varnothing$, $Y_\beta = \varnothing$, $Y_0 = Y$; после остановки — $|Y_\alpha| = n - 1$, и достижение этого равенства является одним из двух возможных условий остановки. Искомое остовное дерево T строится из α-ребер постепенно, путем пополнения связных поддеревьев, которые иногда называются «*букетами*». Пополнение α-букетов производится до тех пор, пока это возможно без появления α-циклов или достигается условие $|Y_\alpha| = n - 1$. Если $|Y_\alpha| < n - 1$, а окраска α-цветом любого ребра из Y_0 приводит к появлению α-цикла, то алгоритм останавливается, граф G не связен, остовное дерево не существует, и можно скорректировать алгоритм для построения остовного леса.

Формальное описание алгоритма

Шаг 1. Выбрать произвольное ребро y_1, пометить его α-цветом и сформировать первый α-букет, присоединив к ребру y_1 пару инцидентных ему вершин.

Шаг 2. Выбрать любое неокрашенное ребро y_2 и остановиться, если такового нет (в последней ситуации не существует остовного дерева в графе G с числом вершин $n > 2$). Для выбранного ребра y_2 возможны четыре ситуации.

2.1. Обе концевые вершины ребра y_2 принадлежат одному α-букету. Тогда окрасить ребра y_2 цветом β и вернуться к началу шага 2.

2.2. Одна из концевых вершин ребра y_2 принадлежит некоторому α-букету B, другая — не принадлежит ни одному из α-букетов. Тогда y_2 окрасить α-цветом и пополнить букет B ребром y_2 вместе с инцидентной концевой вершиной.

2.3. Обе концевые вершины ребра y_2 не принадлежат ни одному из сформированных букетов. Тогда окрасить y_2 цветом α и сформировать новый букет из y_2 и двух его концевых вершин.

2.4. Обе концевые вершины ребра y_2 принадлежат двум различным букетам. В этом случае окрасить y_2 цветом α и слить два упомянутых букета в один.

Шаг 3. Если все вершины графа G вошли в один букет (это эквивалентно условию $|Y_\alpha| = n - 1$), закончить процедуру, в этом случае подграф $T = (X, Y_\alpha)$, состоящий из всех вершин и всех α-ребер, является остовным деревом графа G. В противном случае — вернуться к началу шага 2.

Остов минимального веса

Взвешенным графом называется граф $G = (X, Y)$, каждому ребру которого приписан *вес* — положительное число $c_i = C(y_i) > 0$ (иногда $c_i \geq 0$). ***Весом подграфа*** из G называется сумма весов ребер подграфа. Задача отыскания *остова наименьшего веса* в графе G часто встречается в приложениях: при проектировании компьютерных и кабельных сетей, линий электропередачи, трубопроводов, дорог и т.д. При математическом моделировании этих реальных ситуаций роль вершин графа G играют соответственно компьютеры, компьютерные центры, абоненты и передатчики (в кабельных сетях), источники и потребители электроэнергии, воды, газа и т.д. Роль ребер y_i и весов c_i взвешенного графа G играют допустимые пути прокладки кабелей, труб, дорог и их стоимости. Если связный граф G «допустимых трассировок» близок к полному графу K_n, то число всех остовных деревьев равно n^{n-2}, и прямой перебор всех деревьев и вычислений их весов может иметь удручающий объем вычислений. Например, число остовов τ_n для K_n при $n = 22$ имеет нижнюю оценку $\tau_{22} = 22^{20} > 20^{20} = 2^{45} \cdot 10^{20} > 10^{25}$. Тем не менее, для нахождения остова наименьшего веса имеется целый ряд эффективных алгоритмов. Опишем два из них, ставших уже классическими.

Алгоритм Дж. Краскала нахождения остова минимального веса (1956 г.)

Идея алгоритма состоит в том, что *в каждом шаге алгоритма выбирается кратчайшее ребро, не образующее цикл с выбранными ранее ребрами*.

1 *шаг*. В исходном графе $G = (X, Y)$ строится подграф $T_1 = (X, Y_1)$ с одним ребром $u_1 = Y_1$ минимального веса $c(u_1) = \min\limits_{\forall y_i \in Y} \{c(y_i)\}$.

Шаг k. Если граф $T_{k-1} = (X, Y_{k-1})$ без циклов уже построен, выбираем ребро $u_k \in Y \backslash Y_{k-1}$ так, чтобы оно не образовывало цикла с ребрами Y_{k-1} и имело наименьший вес среди всех таких ребер. Строим граф T_k, присоединяя к графу T_{k-1} ребро u_k; $T_k = (X, Y_k)$, $Y_k = Y_{k-1} \cup \{u_k\}$ $k = 2, 3, \ldots, n - 1$.

Алгоритм Р. Прима
построения остова минимального веса (1957 г.)

В каждом шаге алгоритма Прима строится дерево — связный подграф на подмножестве X_k вершин в G, в то время, как в алгоритме Краскала последовательные подграфы T_k без циклов содержали все множество вершин X графа G, и поэтому при $\kappa < n-1$ графы T_k были несвязными. В остальном алгоритмы похожи друг на друга.

Шаг 1. Выбираем в $G = (X, Y)$ ребро e_1 с минимальным весом $c(e_1) = \min\limits_{\forall y \in Y}\{c(y)\}$ и отмечаем вершины v_1, $v_2 \in X$, которые инцидентны ребру e_1. Строим дерево $T_1 = (X_1, Y_1) \subset G$, где $X_1 = \{v_1, v_2\}$, $Y_1 = \{e_1\}$.

Шаг $k+1$. Пусть дерево $T_k = (X_k, Y_k) \subset G$ уже построено, где $Y_k = \{e_i\}_{i=1}^{k}$, $X_k = \{v_j\}_{j=1}^{k+1}$. Среди множества ребер $Y(X_k, X'_k)$, соединяющих вершины из X_k с вершинами из множества $X'_k = X \backslash X_k$, выбираем ребро e_{k+1} с наименьшим весом: $c(e_{k+1}) = \min\{c(y), \forall y \in Y(X_k, X'_k)\}$. Дерево T_{k+1} строится присоединением к T_k одного ребра e_{k+1} и одной вершины $v_{k+2} \in X'_k$, инцидентной ребру e_{k+1}: $T_{k+1} = (X_{k+1}, Y_{k+1})$, $X_{k+1} = X_k \cup v_{k+2}$, $Y_{k+1} = Y_k \cup e_{k+1}$.

Замечание. Иногда требуется построить остов максимального веса в графе G. Алгоритмы Краскала и Прима полностью решают эту задачу, если в них всюду минимальный вес заменить на максимальный.

Ориентированные и бинарные деревья

Деревья с ориентированными *ребрами (дугами)* используются давно. Достаточно вспомнить генеалогические деревья, ветвящиеся физические и информационные процессы. В настоящее время они существенно используются в вопросах кодирования, теории языков и грамматик, теории автоматов, в логическом и синтаксическом анализе, в теории алгоритмов.

Определение

Ориентированным деревом *(иначе —* ***корневым ориентированным деревом****)* называется ориентированный граф без циклов со свойствами:

а) существует в точности одна вершина r, называемая *корнем*, в которую не заходит ни одна дуга;

б) в каждую вершину, кроме корня, заходит в точности одна дуга;

в) существует путь из корня в любую другую вершину.

Замечание. Выбрав ориентацию на всех ребрах неориентированного графа, мы получаем ориентированный граф. Однако, выбор произвольной ориентации на ребрах неориентированного дерева не обязательно приводит к ориентированному дереву в смысле приведенного выше определения. Например, граф с четырьмя дугами на рис. 6.29 не является ориентированным деревом, граф на рис. 6.30 — ориентированное дерево с корнем $r = 1$.

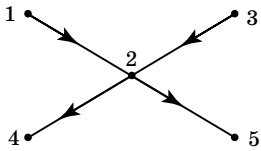


Рис. 6.29.
Неориентированное дерево

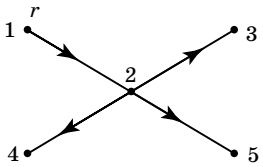


Рис. 6.30.
Ориентированное дерево с корнем r

Относительно элементов ориентированных деревьев установилась следующая терминология. ***Потомок*** v вершины u — это вершина v, в которую ведет *путь* из вершины u; при этом u называется ***предком*** для v. Если длина этого пути равна 1, то есть из u в v ведет дуга, то u — «***отец***» для v, а v — «***сын***» для u. Вершина, не имеющая потомков, называется ***листом***. ***Глубина вершины*** v в дереве — это длина пути из корня в v. ***Высотой вершины*** v в дереве называется длина самого длинного пути из v в какой-нибудь лист. ***Высота дерева*** — это число дуг самого длинного пути, то есть высота корня. ***Уровень вершины*** v — это разность между высотой всего дерева и глубиной вершины v. Высота дерева на рис. 6.31 равна 3, вершина 8 имеет глубину 2, высоту 1, уровень 1.

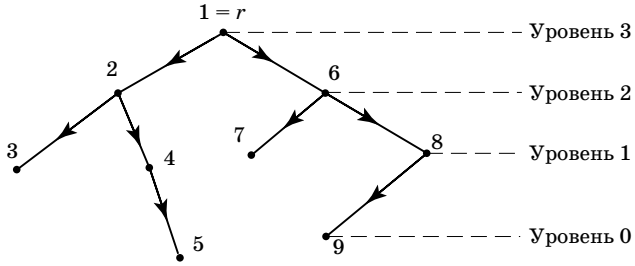


Рис. 6.31. Корневое дерево, упорядоченное по уровням

Листами являются вершины 3, 5, 7, 9; их уровни и глубина, в принципе, различны: листы 3, 7 имеют уровень 1, глубину 2, листы 5, 9 имеют уровень 0, глубину 3. Высота же любого листа равна 0.

Упорядоченным деревом (ориентированным, разумеется) называется дерево, у которого множество сыновей каждой вершины упорядочено, обычно слева направо, как на рис. 6.31: $2 < 6$, $3 < 4$, $7 < 8$.

Определение

Бинарным (двоичным) деревом *T называется упорядоченное дерево, из каждой вершины которого может исходить не более двух дуг.*

Напомним, что упорядоченное дерево T является ориентированным, с корнем, и потому в каждую вершину, отличную от корня, заходит ровно одна дуга. Каждая вершина бинарного дерева может иметь либо двух сыновей — *левого* и *правого* (так вершина 2 рис. 6.31 имеет левого сына 3 и правого сына 4), либо иметь только левого сына (левый сын 9 вершины 8), либо только правого сына (вершина 5 — единственный правый сын вершины 4), либо не иметь ни одного сына (листья 3, 5, 7, 9).

Левым поддеревом вершины u называется максимальное поддерево $T_1(u_1) \subset T$, корнем которого является левый сын u_1 вершины u. Соответственно ***правое поддерево*** $T_r(u_r)$ вершины u определяется как максимальное поддерево с корнем u_r — правым сыном вершины u. На рис. 6.31 левое поддерево вершины 6 состоит из одной вершины 7, правое поддерево — из вершин 8, 9 и дуги (8, 9).

Компьютерное задание двоичного дерева T обычно представляет собой два массива: ЛЕВЫЙ СЫН и ПРАВЫЙ СЫН. Если вершины T занумерованы числами от 1 до n, то в массиве ЛЕВЫЙ СЫН $[i] = k$ тогда и только тогда, когда вершина «k» является левым сыном вершины 'i', и $[i] = 0$ тогда и только тогда, когда вершина 'i' не имеет левого сына. Для дерева рис. 6.31 представление указанными двумя массивами имеет вид таблицы 6.3.

Таблица 6.3. Представление дерева рис. 6.31

$[i]$	1	2	3	4	5	6	7	8	9	
ЛЕВЫЙ СЫН	2	3	0	0	0	7	0	9	0	
ПРАВЫЙ СЫН	6	4	0	5	0	8	0	0	0	

Задания

1. Проверить, что путь из корня в любую другую вершину является единственным.
2. Перечислить все попарно не изоморфные друг другу деревья четвертого порядка (то есть с четырьмя вершинами); аналогично — пятого порядка.
3. Если два дерева изоморфны, то последовательности степеней их вершин $\delta_1 \le \delta_2 \le \ldots \le \delta_n$ одинаковы. Верно ли обратное?
4. Верно ли, что любой маршрут в дереве является простой цепью? Доказать или опровергнуть.
5. Перечислить неизоморфные деревья с шестью вершинами (t_6).
6. Перечислить корневые деревья с пятью вершинами ($T_5 = 9$).
7. Скорректировать алгоритм построения остовного дерева поиском в глубину для случая несвязного графа и получить соответствующий алгоритм построения остовного леса.
8. Модифицировать алгоритм построения остовного дерева произвольным перебором ребер в алгоритм построения остовного леса несвязного графа.

6.9. Теорема Пуанкаре. Фундаментальные матрицы сечений и циклов

1. В ориентированном графе $G = (X, Y, f)$ каждому *циклу* q (то есть замкнутой цепи, п. 6.2) ставится в соответствие вектор-цикл $\vec{q}$ по следующему правилу: компонента q_i строки $\vec{q}$ равна нулю, если дуга y_i не входит в цикл q; если же $y_i \in q$, то $q_i = 1$ или $q_i = -1$ в зависимости от того, совпадает или нет направление дуги y_i с выбранным положительным направлением обхода цикла. Циклу $q = \{y_1, y_2, y_4, y_6\}$ в графе на рис. 6.32 отвечает вектор-цикл

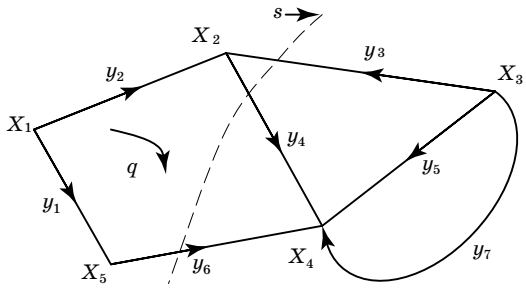


Рис. 6.32. Цикл в графе

$$\begin{array}{c} y_1 \quad y_2 \quad y_3 \quad y_4 \quad y_5 \quad y_6 \quad y_7 \\ \cdots\cdots\cdots\ \cdots\cdots\cdots\ \cdots\cdots\cdots\ \cdots\cdots\cdots\ \cdots\cdots\cdots\ \cdots\cdots\cdots\ \cdots\cdots\cdots . \\ \vec{q} = (-1 \quad 1 \quad 0 \quad 1 \quad 0 \quad -1 \quad 0) \end{array}$$

Любое *максимальное линейно независимое множество векторциклов, объединенных по строкам в матрицу* Q*, называется* ***матрицей циклов графа***[1].

Сечением (разрезом) s графа G называется определенное подмножество дуг, удаление которого увеличивает число компонент связности, так что подграф $G \backslash s$ состоит из двух дизъюнктных частей G_1 и G_2, связь между которыми в графе G осуществляется только дугами сечения s (рис. 6.33).

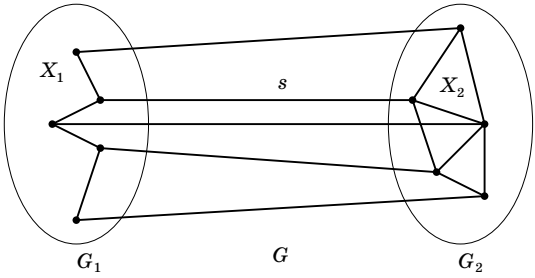


Рис. 6.33. Сечение графа

Вершины X_1 из G_1, которые инцидентны дугам s, называются первой стороной сечения; вершины X_2 из G_2, инцидентные s — второй стороной сечения s. Таким образом, непересекающиеся подмножества вершин X_1, X_2 образуют ***разбиение*** множества всех вершин $X = X_1 \cup X_2$ ***графа*** G. Для сечения (как и для цикла) произвольно вводится ориентация — направление от одной стороны сечения к другой. Вектор-сечение $\vec{s}$ — это строка $\vec{s}$, у которой компонента $s_i = 0$, когда дуга y_i, не входит в сечение s. Если дуга y_i принадлежит сечению согласованно с выбранной ориентацией сечения, то $s_i = 1$; если не согласованно, то $s_i = -1$. Для сечения $s = \{y_3, y_4, y_6\}$ в графе (рис. 6.32) вектор-сечение равно:

$$\begin{array}{c} y_1 \quad y_2 \quad y_3 \quad y_4 \quad y_5 \quad y_6 \quad y_7 \\ \cdots\cdots\cdots\ \cdots\cdots\cdots\ \cdots\cdots\cdots\ \cdots\cdots\cdots\ \cdots\cdots\cdots\ \cdots\cdots\cdots\ \cdots\cdots\cdots . \\ \vec{s} = (0 \quad 0 \quad -1 \quad 1 \quad 0 \quad 1 \quad 0) \end{array}$$

[1] Иногда Q называется «базисной матрицей циклов», а матрицей циклов — совокупность всех векторов циклов, вообще говоря, линейно-зависимых.

Матрица S, строки которой образуют максимальный линейно-независимый набор вектор-сечений, называется ***матрицей сечений***[1] графа G. Столбцы матриц сечений и циклов нумеруются одинаково — дугами графа, а число их строк в общем случае различно.

Задания

1. Построить матрицы сечений S и циклов Q графа на рис. 6.34 и убедится, что их объединение вида $\begin{bmatrix} S \\ Q \end{bmatrix}$ есть квадратная невырожденная матрица.
2. Каждая строка $\vec{b}_i$ матрицы инциденций B (п. 10.6) является вектор-сечением, отвечающим сечению s_i вокруг вершины x_i (рис. 6.34).

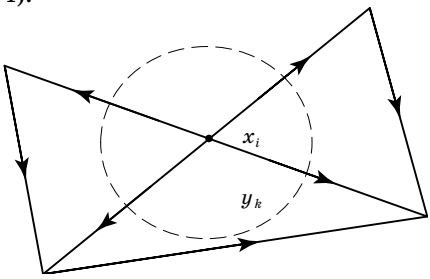


Рис. 6.34. Сечение вокруг вершины

2. В графе G с m дугами каждый вектор-цикл $\vec{q}$ ортогонален любому вектору-сечению $\vec{s}$ (теорема Пуанкаре):

$$\vec{q} \cdot \vec{s}^T = (\vec{q}, \vec{s})_{R^m} = 0. \qquad (6.16)$$

Более того, если ввести пространство вектор-циклов H_q как линейную оболочку вектор-циклов[2] и пространство вектор-сечений H_s как линейную оболочку вектор-сечений, то

$$H_q \oplus H_s = R^m. \qquad (6.17)$$

Таким образом, помимо свойства ортогональности (6.16) вектор-циклы и вектор-сечения графа с m дугами образуют ***тотальную систему*** элементов в пространстве R^m, то есть систему векторов, линейная оболочка которых есть все пространство R^m.

Докажем сначала соотношение (6.16).

---

[1] Или «базисной матрицей сечений», в отличие от матрицы всех сечений графа.

[2] Где линейные комбинации образуются с любыми вещественными коэффициентами, а не только с целочисленными.

Теорема 6.10 *(А. Пуанкаре)*

Произвольный вектор-сечение $\vec{s}$ *ортогонален всякому вектор-циклу* $\vec{q}$ *графа* G.

□ В сумме

$$\vec{s}\vec{q}^T = \sum_{j=1}^{m} s_j q_j = \vec{q}\vec{s}^T \qquad (6.18)$$

отличны от нуля только те слагаемые $(s_j\ q_j)$, для которых дуга y_j входит одновременно в рассматриваемые сечение s и цикл q. Число таких ребер y_j четно. Действительно, при обходе цикла q прохождение дуги $y_j \in s$ в направлении ориентации сечения должно компенсироваться прохождением некоторой дуги $y_k \in s$ в направлении, противоположном ориентации s. Предположим, дуги y_j, y_k сонаправлены с положительным направлением обхода цикла и $q_j = q_k = 1$. Тогда $sgn\ s_j = -sgn\ s_k$ и $s_j q_j + s_k q_k = 0$. Смена направления дуги y_j приводит к одновременной смене знаков чисел s_j и q_j, так что значение $s_j q_j$ сохраняется. В силу отмеченной четности сумма (6.18) равна нулю:

$$\sum_{\forall j} s_j q_j = 0. \qquad \blacksquare$$

3. Доказательство разложения (6.17) удобно провести с помощью так называемой ***фундаментальной системы циклов и сечений***, отвечающей фиксированному каркасу T графа G (***каркасом*** называется максимальный подграф без циклов).

Ясно, что достаточно рассматривать эти вопросы для связного графа, так как в несвязанном графе векторы $\vec{s}$, $\vec{q}$ являются прямыми суммами соответствующих векторов в компонентах связности.

Каркас T связанного графа $G = (X, Y, f)$ является, очевидно, деревом $T = (X, Y^T, f^T)$, содержащим все вершины графа. ***Кодеревом*** G_c называется дополнение

$$G_c = G \backslash T = (X, Y \backslash Y^T = Y_c, f_c = f|Y_c).$$

Например, один из каркасов графа G рис. 6.35 изображен жирным линиями (каркас «большой медведицы») и соответственно дуги кокаркаса $Y_c = \{y_7, y_8, y_9, y_{10}\}$ — тонкими линиями.

Сечение s называется ***фундаментальным*** (по дереву T), если оно содержит в точности одну дугу y_k дерева T, а ориентация сечения s согласована с ориентацией дуги y_k. Ясно, что задание дуги дерева полностью определяет соответствующее фундаментальное

сечение. На рис. 6.35 пунктирами показаны фундаментальные сечения $s_1, s_2, \ldots, s_6$ по дереву T «большая медведица».

Фундаментальной матрицей сечений S_T (по дереву T) называется матрица, строками которой являются вектор-сечения, фундаментальные по дереву T. Выделенному дереву T на рис. 6.35 отвечает фундаментальная матрица сечений, столбцы которой помечены номерами дуг всего графа, строки — номерами дуг дерева, точнее, номерами соответствующих фундаментальных сечений:

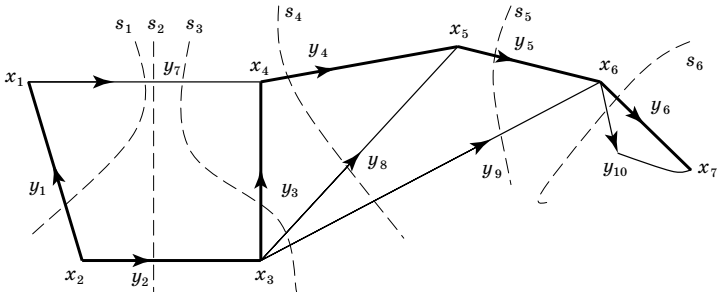


Рис. 6.35. Дерево «большой медведицы»

$$S_T = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \end{array} \begin{array}{c} \begin{array}{cccccc|cccc} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 \end{array} \\ \left[\begin{array}{cccccc|cccc} 1 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \end{array}\right] \end{array} \qquad (6.35)$$

Фундаментальным циклом графа G ***(по дереву*** T**)*** называется всякий цикл, содержащий в точности одну дугу y_i кодерева $G \backslash T$ и ориентированный согласованно с направлением дуги y_i. Ясно, что каждая дуга y кодерева однозначно задает фундаментальный цикл q_y, который дуга y замыкает на дереве T. ***Фундаментальной матрицей циклов*** Q_T (по дереву T) называется матрица, строки которой есть все фундаментальные вектор-циклы по дереву T. Если для графа на рис. 6.35 фундаментальные вектор-циклы пометить номерами задающих дуг кодерева y_7, y_8, y_9, y_{10}, то фундаментальная матрица циклов такова:

$$Q_T = \begin{array}{c} \\ 7 \\ 8 \\ 9 \\ 10 \end{array} \begin{array}{c} \begin{array}{cccccc|cccc} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 \end{array} \\ \left[\begin{array}{cccccc|cccc} 1 & -1 & -1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & -1 & -1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & -1 & -1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 1 \end{array}\right] \end{array} \qquad (6.20)$$

Замечание. Относительно разбиения множества всех дуг $Y = Y_T \cup Y_c$ на дуги дерева и кодерева фундаментальные матрицы сечений и циклов разбиваются на блоки вида:

$$\overset{\quad Y_T \quad Y_c}{S_T = Y_T[E; \quad S_0]}; \qquad \overset{\quad Y_T \quad Y_c}{Q_T = Y_c[Q_0; \quad E]}. \qquad (6.21)$$

Из теоремы Пуанкаре получаем:

Следствие. *Для произвольных матрицы сечений S и циклов Q данного графа G справедливо соотношение ортогональности*

$$SQ^T = 0; \qquad QS^T = 0. \qquad (6.22)$$

Информационные блоки S_0, Q_0 (6.21) фундаментальных матриц S_T, Q_T (по одному дереву) выражаются друг через друга по формуле:

$$S_0 = -Q_0^T; \qquad Q_0 = -S_0^T. \qquad (6.23)$$

Поэтому достаточно вычислить одну из фундаментальных матриц S_T, Q_T.

Теперь можно обосновать разложение (6.17) пространства R^m в ортогональную сумму пространства вектор-циклов H_q и вектор-сечений H_s. Пространство H_q (H_s) содержит линейную оболочку $Z_q(Z_s)$ строк матрицы Q_T (S_T). В силу (6.22) имеем $Z_q \perp Z_s$. В то же время из-за единичных блоков в (6.21) $\dim Z_s + \dim Z_q = m$, откуда $Z_q \oplus Z_s = R^m$.

Следствие. *Квадратная* $(m \times m)$ *матрица* $\begin{bmatrix} S \\ Q \end{bmatrix}$, *в частности* $\begin{bmatrix} S_T \\ Q_T \end{bmatrix}$ *имеет максимальный ранг* m.

Следствие (из формулы (6.21)). *Для связного графа с* m *дугами и* n *вершинами*

$$\operatorname{rang} S = \operatorname{rang} S_T = |Y_T| = n - 1; \quad \operatorname{rang} Q = m - n + 1 = |Y_c|.$$

Величина rang Q совпадает с цикломатическим числом ν — числом ячеек плоского графа. В случае неплоского связного графа число $\nu = m - n + 1$ также называется цикломатическим,

оно совпадает с размерностью пространства вектор-циклов $\dim H_q$ (или максимальным числом линейно независимых вектор-циклов графа).

Задания

1. Эквивалентны ли друг другу (и в каком смысле) матрица инциденций и сечений?
2. Полная линейно независимая система уравнений Кирхгофа электрической цепи на графе G может быть записана с помощью матриц сечений и циклов в векторной форме так:

   $S\vec{I} = 0$; $Q\vec{I} = 0$. Докажите это.

   Здесь $\vec{I}(\vec{U})$ — вектор всех токов (напряжений) ветвей.
3. Докажите, что:

   а) всякая невырожденная квадратная матрица, составленная из столбцов матрицы сечений, отвечает некоторому каркасу графа;

   б) всякая невырожденная квадратная матрица, составленная из столбцов матрицы циклов, отвечает некоторому кокаркасу в графе.
4. Убедитесь в том, что для плоского графа задача перечисления сечений графа G эквивалентна задаче перечисления циклов двойственного графа $\overline{G}$.
5. Построить фундаментальные матрицы сечений и циклов двух графов Понтрягина — Куратовского, изображенных на рис. 6.36. Убедиться в ортогональности матриц S и Q^T для каждого графа.

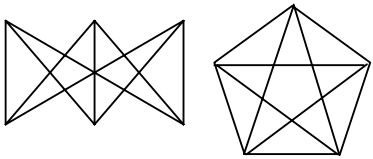


Рис. 6.36. Графы Понтрягина — Куратовского

6. Подмножество ребер Y_0 в планарном графе $G = (X, Y, f)$ образует простой цикл тогда и только тогда, когда соответствующее ему подмножество ребер $\overline{Y_0}$ в геометрически двойственном графе $\overline{G}$ образует сечение. Докажите это.

6.10. Кратчайшие расстояния и пути в сетях

Задачи отыскания кратчайших расстояний между вершинами графа (сети) с заданными весами ребер часто встречаются на практике и имеют очевидный экономический смысл. Это же

можно сказать и о задачах поиска оптимальных путей в ориентированных сетях. В данном разделе приводятся некоторые эффективные алгоритмы решения таких задач: алгоритмы Дейкстры, Форда, Флойда, Данцига.

Пусть $G = (X, Y)$ — взвешенный ориентированный связный граф с положительными длинами (весами) дуг $c(y) \geq 0$, требуется найти кратчайший путь (с минимальной суммой весов дуг) из вершины s в вершину t графа. Эффективный алгоритм был предложен Е. Дейкстрой в 1959 г. Его идея такова. Если уже найдены длины (веса) $l(x_i)$ самых коротких путей из вершины s и указаны («окрашены») те вершины $x_1 = s, x_2, \ldots, x_k$, к которым ведут эти пути, то «очередной» по длине путь из s в $(k + 1)$-ую вершину x_{k+1} в конце этого пути можно найти по следующему правилу. Обозначим через $X_k = \{x_i\}_{i=1}^{k}$ множество окрашенных вершин. Для каждой неокрашенной вершины w просмотреть длины $c(x_i, w)$ дуг (x_i, w), $x_i \in X_k$, $w \in W_k = X \backslash X_k$ и найти

$$\min\{c(x_i, w)\} = c(v_0, w_0),\ v_0 \in X_k,\ w_0 \in W_k. \qquad (6.24)$$

Когда вершины x_i, w не смежны, то $c(x_i, w) = \infty$. Если $c(x_i, w) = \infty$ для всех $x_i \in X_k$ и всех $w \in W_k$, то алгоритм останавливается: все вершины w множества W_k и, в частности, вершина t не достижимы из s. Если существует хотя бы одна дуга из множества вершин X_k во множество W_k, тогда минимум (6.24) конечен и вершина w_0 будет найдена. Искомый («очередной» по длине) путь ведет из вершины s в вершину w_0, длина этого пути равна $l(v_0) + c(v_0, w_0)$. Мы можем окрасить вершину w_0, то есть присвоить ей обозначение $x_{k+1} = w_0$, положить $l(x_{k+1}) = l(v_0) + c(v_0, w_0)$ и сформировать более широкое множество окрашенных вершин $X_{k+1} = \{x_i\}_{i=1}^{k+1}$. Процесс заканчивается, если $x_{k+1} = t$.

Формально в алгоритме окрашенной вершине $x \in X_k$ присваивается постоянная метка $l(x)$ — длина кратчайшего пути из s в x. Если вершина $w \in W_k$ еще не имеет постоянной метки, т.е. не окрашена, ей присваивается временная метка $l(w)$, равная длине кратчайшего (s, w)-пути, проходящего только по вершинам с постоянными метками (окрашенным). Временная метка вершины w, вообще говоря, изменяется от шага к шагу и перестает меняться, когда вершина w окрашивается (получает постоянную метку).

Практически в k-ом шаге алгоритма строится корневое дерево $T_k (\supset T_{k-1})$ от корня s до вершин множества X_k с постоянными пометками. На последнем шаге k_0 вершина t становится концевой вершиной дерева T_{k_0}.

Замечание. Алгоритм можно скорректировать в пункте «останов», если ставить задачу нахождения кратчайших путей из корня s до всех остальных вершин графа G при условии, что они достижимы из s. В этом случае алгоритм останавливается только тогда, когда будет построено дерево T_n, содержащее все вершины из X. *Таким образом, алгоритм Дейкстры построит остовное ориентированное дерево T_n графа G с корнем s.*

Замечание. Алгоритм Дейкстры может находить кратчайшие расстояния между парами вершин в *неориентированном связном взвешенном графе* G с *положительными весами ребер* $c(y)$.

Для применения описанной выше процедуры следует перейти к ориентированному симметричному графу, заменив каждое ребро парой противоположно направленных дуг одинакового веса. Остовное дерево T_n всегда может быть построено с помощью алгоритма Дейкстры, и T_n будет состоять из кратчайших простых цепей между вершиной s и остальными вершинами графа G.

Формальное описание алгоритма Дейкстры построения кратчайших путей из вершины s

Шаг 1. Присвоение начальных значений.

Положить $l(s) = 0$ и считать эту пометку постоянной (то есть окрасить вершину s). Положить $l(w) = \infty$ для всех вершин $w \neq s$ и считать эти пометки временными. Положить $p = s$ (p — имя последней вершины, получившей постоянную метку).

Шаг 2. Обновление пометок.

Для каждой неокрашенной вершины w, в которую ведет[1] дуга (p, w) из вершины p, метка $l(w)$ изменяется по правилу

$$l(w) \leftarrow \min\{ l(w), l(p) + c(p, w)\}. \qquad (6.25)$$

Шаг 3. Превращение пометки в постоянную.

На множестве всех неокрашенных вершин (с временными метками) W_p найти вершину w_0 с минимальной меткой: $l(w_0) = \min\limits_{w \in W_p}\{l(w)\}$. Сделать метку $l(w_0)$ вершины w_0 постоянной и положить $p = w_0$.

Шаг 4. (Если надо найти лишь кратчайший путь от s к t).

[1] Условие «ведет дуга (p, w)» можно не проверять, но тогда рассматриваются все вершины w с временными метками, и если дуга (p, w) отсутствует, то в (6.25) $c(p, w) = \infty$ и временная метка $l(w)$ не изменяется, см. пример рис. 6.38.

4.1. Если $p = t$, то $l(p)$ — длина кратчайшего (s, t)-пути. *Останов*.

4.2. Если $p \neq t$, то перейти к шагу 2.

Шаг 5. (Если требуется найти кратчайшие пути от s до всех вершин графа G).

Алгоритм выполняется так долго, чтобы все вершины получили постоянные метки и множество неокрашенных вершин исчерпалось: $W_p = 0$. *Останов*.

Сложность алгоритма Дейкстры

Заметим, что оценка сложности алгоритма не зависит от того, ищутся ли кратчайшие пути от вершины s до всех остальных вершин или только от s до фиксированной вершины t. Действительно, если кратчайший (s, t)-путь оказывается длиннейшим из всех кратчайших (s, w)-путей, то обе задачи решаются за одинаковое число шагов. В этом случае каждый из шагов 2, 3, 5 выполняется по $(n - 1)$ раз. Трудоемкость одного шага 2 равна $\theta(1)$, одного шага 5 — также $\theta(1)$; трудоемкость шага 3 имеет порядок средней мощности множества W_k — числа $\frac{(n-1)+1}{2} = \frac{n}{2}$. Следовательно, трудоемкость всего алгоритма имеет порядок числа $\frac{n(n-1)}{2} + 2(n-1)$, то есть $0(n^2)$, $n = |X|$.

Пример *(работа алгоритма Дейкстры)*

Найдем кратчайший путь между вершинами s и t во взвешенном графе G на рис. 6.37, одновременно указав кратчайшие пути меньшей длины из вершины s до соответствующих вершин (веса дуг, равные целым числам, указаны на рисунке). Множество вершин есть $X = \{s, a, b, e, d, t\}$, $|X| = n = 6$.

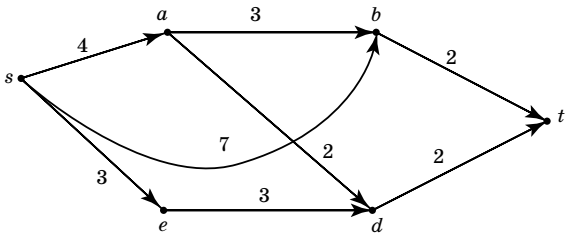


Рис. 6.37. Граф для иллюстрации алгоритма Дейкстры

Обозначения. $c(u, v)$ — вес (длина) дуги (u, v), $l(w)$ — длина (s, w)-пути или пометка вершины w на данном шаге, $l_(w)$ — длина (s, w)-пути или пометка вершины w на предыдущем шаге.

Шаг 1. Присвоение начальных значений.

$l(s) = 0$ — постоянная пометка; s окрашивается.

$l(a) = \infty$, $l(e) = \infty$, $l(b) = \infty$, $l(d) = \infty$, $l(t) = \infty$ — временные пометки вершин a, e, b, d, t. Полагаем $p = s$.

Шаг 2. s. $(p = s)$. Обновление пометок.

$l(a) = \min\{l_(a), l(s) + c(s, a)\} = \min\{\infty, 0 + 4\} = 4$

$l(b) = \min\{l_(b), l(s) + c(s, b)\} = \min\{\infty, 0 + 7\} = 7$

$l(e) = \min\{l_(e), l(s) + c(s, e)\} = \min\{\infty, 0 + 3\} = 3$

$l(d) = \min\{l_(d), l(s) + c(s, d)\} = \min\{\infty, 0 + \infty\} = \infty$

$l(t) = \min\{l_(t), l(s) + c(s, t)\} = \min\{\infty, 0 + \infty\}$.

Шаг 3. s. $(p = s)$. Превращение метки в постоянную.

$$\min\{l(w), w \in W_{\text{р}} = \{a, e, b, d, t\}\} =$$
$$= \min\{4, 7, 3, \infty, \infty\} = 3 = l(e).$$

Вершина e получает постоянную метку $l(e) = 3$, вершина e — окрашивается вместе с дугой (s, e). Текущее дерево T_1 кратчайших путей из вершины s содержит вершины s, e и дугу (s, e) (см. рис. 6.38).

Полагаем $p = e$.

Шаг 2. e. $(p = e)$. Поскольку $t \notin T_1$, переходим к шагу 2 при $p = e$, обновление пометок.

$l(a) = \min\{l_(a), l(e) + c(e, a)\} = \min\{4, 3 + \infty\} = 4$

$l(b) = \min\{l_(b), l(e) + c(e, b)\} = \min\{7, 3 + \infty\} = 7$

$l(d) = \min\{l_(d), l(e) + c(e, d)\} = \min\{\infty, 3 + 3\} = 6$

$l(t) = \min\{l_(t), l(e) + c(e, t)\} = \min\{\infty, 3 + \infty\} = \infty$.

Шаг 3. e. $(p = e)$. Превращение метки в постоянную.

$$\min\{l(w), w \in W_{\text{р}} = \{a, b, d, t\}\} = \min\{4, 7, 6, \infty\} = 4 = l(a).$$

Вершина a получает постоянную метку $l(a) = 4$. Вершина a и дуга (s, a) окрашиваются. Текущее дерево кратчайших путей T_2 состоит из вершин s, e, a и дуг (s, e), (s, a) (см. рис. 6.38).

Полагаем $p = a$.

Шаг 2. a. $(p = a)$. Обновление пометок, так как $t \notin T_2$.

$l(b) = \min\{l_(b), l(a) + c(a, b)\} = \min\{7, 4 + 3\} = 7$

$l(d) = \min\{l_(d), l(a) + c(a, d)\} = \min\{6, 4 + 2\} = 6$

$l(t) = \min\{l_(t), l(a) + c(a, t)\} = \min\{\infty, 4 + \infty\} = \infty$.

Шаг 3. a. $(p = a)$. Превращение метки в постоянную.

$$\min\{l(w), w \in W_{\text{р}} = \{a, d, t\}\} = \min\{7, 6, \infty\} = 6 = l(d).$$

Вершина d получает постоянную метку $l(d) = 6$. Вершина d и дуга (a, d) окрашиваются. Текущее дерево кратчайших путей T_3 состоит из вершин s, e, a, d и дуг (s, e), (s, a), (a, d) (см. рис. 6.38).

Полагаем $p = d$.

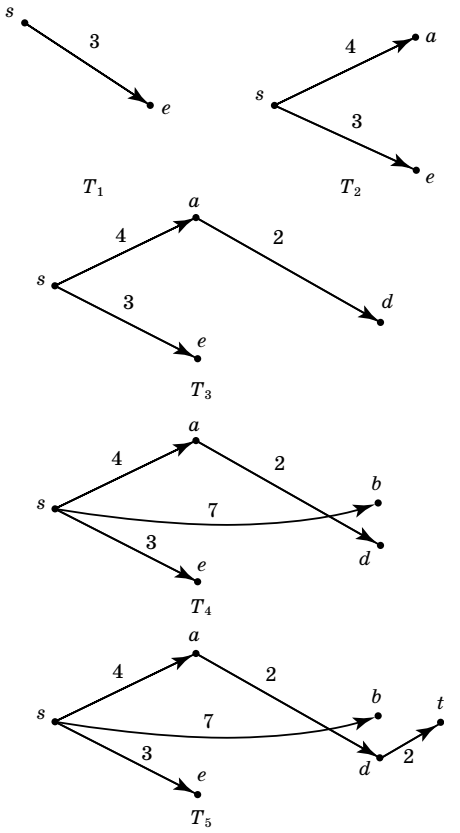


Рис. 6.38. Растущие ориентированные деревья T_p кратчайших путей

Шаг 2. d. $(p = d)$. Обновление пометок, так как $t \notin T_3$.

$l(b) = \min\{l_(b), l(d) + c(a, b)\} = \min\{7, 6 + \infty\} = 7$

$l(t) = \min\{l_(t), l(d) + c(a, t)\} = \min\{\infty, 6 + 2\} = 8$.

Шаг 3. d. $(p = d)$. Превращение метки в постоянную.

$\min\{l(w), w \in W_p = \{b, t\}\} = \min\{7, 8\} = 7 = l(b)$.

Вершина b получает постоянную метку $l(b) = 7$ и окрашивается вместе с дугой (b). Текущее дерево кратчайших путей T_4 состоит из вершин s, e, a, d, b и дуг (s, e), (s, a), (a, d), (s, b) (см. рис. 6.38).

Полагаем $p = b$.

Шаг 2. b. ($p = b$). Обновление пометок, так как $t \notin T_4$.

$l(t) = \min\{l_(t), l(b) + c(b, t)\} = \min\{8, 7 + 2\} = 8$.

Шаг 3. b. ($p = b$). Превращение метки в постоянную.

$\min\{l(w), w \in W_p = \{t\}\} = 8 = l(t)$.

Вершина t получает постоянную метку $l(t) = 8$ и окрашивается вместе с дугой (d, t). Текущее дерево кратчайших путей T_5 содержит все вершины графа G и дуги (s, e), (s, a), (a, d), (s, b), (d, t) (см. рис. 6.38). Дерево T_5 является остовным деревом в G. Останов.

Кратчайший путь из s в t есть (s, a), (a, d), (d, t). Он не является единственным, такую же длину 8 имеет путь (s, e), (e, d), (d, t).

Алгоритм Форда

В алгоритме Дейкстры предполагалось, что веса (длины) всех дуг *положительны*. Возможны и неотрицательные веса $c_j \geq 0$, так как последовательное стягивание дуг с нулевыми весами приводит к взвешенному графу с положительными весами дуг.

Однако, если есть дуги с *отрицательными и положительными* весами, алгоритм Дейкстры, вообще говоря, не находит кратчайшего пути.

Еще до появления в 1959 г. алгоритма Дейкстры были предложены процедуры нахождения кратчайших путей по дугам с любыми знаками весов: 1956 г. — Форд, 1957 г. — Мур, 1958 г. — Беллман. В главном эти процедуры похожи; соответствующая вычислительная процедура называется сейчас *алгоритмом Форда построения кратчайших путей в случае произвольных весов*.

Алгоритм Форда можно изложить как *модификацию* алгоритма Дейкстры. Изменения алгоритма Дейкстры состоят в следующем:

1. На шаге 2 «Обновление пометок» пересчет величин $l(w)$ по формуле (6.25) производится *для всех вершин* — как с временными, так и с постоянными метками (неокрашенными и окрашенными). При этом длины $l(w)$ могут уменьшаться как для неокрашенных вершин, так и для окрашенных.

Понятие «постоянная метка» вершины x_p становится некорректным, и удобнее говорить, что вершина x_p *окрашена* (на шаге 3), но, возможно, временно.

2. Если для некоторой окрашенной вершины x_p после применения шага 2 происходит уменьшение «расстояния» $l(x_p)$, тогда с вершины x_p и ведущей в нее окрашенной дуги *снимается окраска*.

3. Алгоритм Форда заканчивается тогда, когда все вершины графа окрашены и еще одно выполнение шага 2 не меняет ни одно из чисел $l(x)$, $\forall x \in X$.

Условие корректности алгоритма Форда, то есть завершения, согласно замечания 3, за конечное число шагов, состоит в следующем*: граф G не должен содержать контуров отрицательной «длины»* (с отрицательной суммой весов дуг).

Вообще говоря, выявление контуров отрицательной длины является отдельной задачей. Однако, для определения корректности алгоритма Форда эту задачу не обязательно решать отдельно. Достаточно обычным образом применять алгоритм Форда, пока либо алгоритм закончится в соответствии с замечанием 3, либо число окрашиваний какой-либо одной вершины не достигнет величины n-количества всех вершин. В последнем случае граф имеет контур отрицательной длины и алгоритм Форда *не корректен.*

В принципе, алгоритм Форда нахождения оптимального пути из вершины s в вершину t можно применять к графу с *неотрицательными весами дуг*, однако в этом случае он менее эффективен по скорости в сравнении с алгоритмом Дейкстры, т.к. сложность алгоритма Форда равна $\theta(n^3)$, а алгоритма Дейкстры $-\theta(n^2)$.

Алгоритмы Флойда и Данцига

Задача поиска *кратчайших путей между всеми парами вершин графа* может быть решена путем последовательного присвоения метки «начальная вершина s» *каждой вершине графа G* и последующего применения алгоритма Дейкстры (Форда). Ясно, что сложность такого «многократного алгоритма Дейкстры» есть $\theta(n^3)$, а «многократного алгоритма Форда» — соответственно $\theta(n^4)$. Однако существуют более эффективные методы, чем многократное (именно n-кратное) повторение алгоритма Дейкстры или Форда — именно алгоритмы Флойда (1962) и Данцига (1967). Оба алгоритма применимы как в случае неотрицательных весов дуг, так и в случае весов произвольных знаков. Однако, в любом случае *необходимо, чтобы граф G не имел контуров отрицательной длины*.

Предварительно пронумеруем вершины графа G натуральными числами 1, 2, ..., n и введем обозначение $l^m_{i,j}$ для длины кратчайшего из всех таких путей от вершины i до вершины j (иначе — всех таких (i, j) — путей, которые в качестве промежуточных вершин могут содержать только первые m вершин, то есть вершины из множества $V_m = \{1, 2, ..., m\}$. При этом $l^m_{i,j} = \infty$, если не существует ни одного (i, j)-пути с промежуточными вершинами только из V_m. Для $m = 0$ считается, что $V_m = \{\varnothing\}$ — пустое множество и l^0_{ij} — длина кратчайшей (i, j)-дуги (то есть (i, j)-пути без промежуточных вершин) при $i \neq j$. Для $i = j$ полагается $l^0_{i,i} = 0$.

Алгоритм Флойда ставит своей целью последовательно для $m = 0, 1, 2, ..., n$ вычислять n^2 чисел (длин) $l^m_{i,j}$ и заканчивать вычисления получением $l^n_{i,j}(i, j = 1, 2, ..., n)$ искомых кратчайших длин путей между всеми парами (i, j) вершин в графе G. Если для фиксированного m обьединять все длины $l^m_{i,j}$ в матрицу, то задача состоит в последовательном вычислении матриц $L_0, L_1, ..., L_{m-1}, L_m, ..., L_n$.

При этом оказывается возможным вычислить матрицу L_m, используя только лишь непосредственно предшествующую матрицу L_{m-1}. Действительно, пусть известны все кратчайшие длины $l^{m-1}_{i,j}(i, j = 1, ..., n)$. Разобьем множество всех (i, j) — путей через вершины V_m на два класса: обязательно проходящие через вершину m и не проходящие через m (то есть проходящие только через вершины из V_{m-1}). Кратчайший путь во втором классе имеет длину $l^{m-1}_{j,j}$. Кратчайший путь через вершину m в первом классе состоит из двух частей: (i, j) — пути длины $l^{m-1}_{i,m}$ и (m, j) — пути длины $l^{m-1}_{m,j}$. Поэтому кратчайший (i, j) — путь через вершину m (с промежуточными вершинами только из V_m) имеет длину $l^{m-1}_{i,m} + l^{m-1}_{m,j}$. Следовательно,

$$l^m_{i,j} = \min\{l^{m-1}_{ij}, l^{m-1}_{i,m} + l^{m-1}_{m,j}\}. \qquad (6.26)$$

Существенно, что граф G не имеет контуров отрицательной длины, иначе реккурентное правило (6.26) не будет справедливым!

Теперь нам остается привести формальное описание алгоритма Флойда.

Описание алгоритма Флойда поиска кратчайших путей между всеми парами вершин

Шаг 1. Перенумеровать вершины графа G числами 1, 2, ..., n. Сформировать матрицу длин $L_0 = \{l^0_{i,j}\}$, положив число $l^0_{i,j}$ при $i \neq j$ равным минимальной длине дуги из вершины i в вершину j либо $l^0_{i,j} = \infty$, если таких дуг нет. Если $i = j$, то $l^0_{i,j} = 0$.

Шаг 2. Последовательно для $m = 1, 2, \ldots, n$, исходя из элементов $l_{i,j}^{m-1}$ матрицы L_{m-1}, вычислить элементы матрицы $L_m = \{l_{i,j}^m\}$ по рекуррентному правилу (6.26). Фиксировать «кратчайший» (i, j)-путь, имеющий длину $l_{i,j}^m$ при ее вычислении по формуле (6.26).

При $m = n$ алгоритм заканчивает работу. Числа $l_{i,j}^n$ есть длины искомых кратчайших (i, j)-путей в графе G; $i, j = 1, 2, \ldots, n$.

Замечание. При получении матрицы L_m (по L_{m-1}) значения элементов главной диагонали $l_{i,j}^m$, элементов m-той строки $l_{m,j}^m$ и m-го столбца $l_{i,m}^m$ можно не вычислять по формуле (6.26), а положить сразу

$$l_{i,i}^m = 0, \quad l_{m,j}^m = l_{m,j}^{m-1}, \quad l_{i,m}^m = l_{i,m}^{m-1}.$$

Действительно, вершина m не может выступать в качестве промежуточной для кратчайшего пути, который либо начинается в m, либо заканчивается в вершине m.

Следовательно, по формуле (6.26) необходимо вычислять лишь $(n-1)(n-2)$ элементов матрицы L_m. Тем не менее, *порядок сложности* вычисления L_m для фиксированного m равен n^2. Если учесть, что количество вычисляемых матриц L_m равно n, то получается сложность алгоритма Флойда порядка $\theta(n^3)$.

Алгоритм Данцига поиска всех кратчайших путей

Здесь используются те же понятия и обозначения, что и в алгоритме Флойда: $1, 2, \ldots, n$ — номера вершин; $V_m = \{1, 2, \ldots, m\}$ — множество первых m вершин; $l_{i,j}^m$ — длина «кратчайшего» (i, j)-пути с промежуточными вершинами из V_m. Матрица $L_0 = \{l_{i,j}^0\}(i, j = 1, 2, \ldots, n)$ «кратчайших дуг» между всеми парами вершин графа G имеет размерность $n \times n$ и формируется точно так, как в алгоритме Флойда. Однако в дальнейшем вместо последовательных $(n \times n)$-матриц L_m строятся их «главные» подматрицы D_m размерности $m \times m$:

$$D_m = \{l_{i,j}^m\}_{(i,\, j = 1,\, 2,\, \ldots,\, m)}.$$

Матрица D_1 есть число $l_{1,1}^1 = 0$. В дальнейшем все диагональные элементы также равны нулю: $l_{i,i}^m = 0$; $1 \le i \le m$; $m = 1, 2, \ldots, n$.

В матрице D_m все элементы, не входящие в последнюю строку и в последний столбец, то есть элементы $l_{i,j}^m$ при $1 \le i, j \le m-1$, вычисляются точно так, как в алгоритме Флойда по формуле (6.26). Они представляют собой длины «кратчайших» (i, j)-путей между вершинами $i, j \in V_{m-1}$ с промежуточными вершинами из множества V_m.

Для получения $l^m_{i,m}$ — длины «кратчайшего» пути из вершины $i \in V_{m-1}$ в вершину m через вершины V_m, заметим, что всякий такой путь $S^m_{i,m}$ не может содержать вершину m в качестве промежуточной и концевой одновременно, так как по условию любой контур имеет неотрицательную длину.

Пусть $p(\in V_{m-1})$ — вершина на пути $S^m_{i,m}$, которая предшествует m. Но тогда «кратчайший» путь $S^m_{i,m}$ можно разбить на два пути: $S^{m-1}_{i,p}$ и $S^m_{p,m}$, где $S^m_{p,m}$ состоит из одной дуги (p, m) и, очевидно, длина пути $S^m_{p,m}$ равна числу $l^0_{p,m}$ — элементу матрицы L_0. Ясно, путь $S^{m-1}_{i,p}$ является кратчайшим (i, p)-путем через вершины V_{m-1}, поэтому его длина равна $l^{m-1}_{i,p}$ $(i, p \in V_{m-1})$. Число $l^{m-1}_{i,p}$ является элементом предыдущей матрицы D_{m-1}. По построению

$$l^m_{i,m} = l^{m-1}_{i,p} + l^0_{p,m}. \qquad (6.27)$$

Поскольку «кратчайший» путь $S^m_{i,m}$ нам неизвестен, а известно лишь его существование, то его длину (6.27) можно найти как следующий минимум:

$$l^m_{i,m} = \min_{1 \le k \le m-1} \{l^{m-1}_{i,k} + l^0_{k,m}\}, \quad i = 1, 2, \ldots, m-1. \qquad (6.28)$$

Таким образом, элементы $l^n_{i,m}$ столбца матрицы D_m вычисляются по элементам предыдущей матрицы D_{m-1} и матрицы L_0.

Аналогично для элементов последней строки матрицы D_m, то есть для длин $l^m_{m,j}$ «кратчайших» (m, j)-путей через вершины V_m получаем формулу

$$l^m_{m,j} = \min_{1 \le k \le m-1} \{l^0_{m,k} + l^{m-1}_{k,j}\}, \quad j = 1, 2, \ldots, m-1. \qquad (6.29)$$

Остается привести формальное описание алгоритма Данцига.

Шаг 1. Перенумеровать вершины графа G числами 1, 2, ..., n. Сформировать матрицу L_0 размерности $(n \times n)$, каждый элемент которой $l^0_{i,j}$ есть длина кратчайшей дуги из вершины i в вершину j, где $i \ne j$. Если таких дуг вообще нет, то $l^0_{i,j} = \infty$ $(i \ne j)$. Если $i = j$, то $l^0_{i,i} = 0$.

Шаг 2. Последовательно с помощью рекуррентной процедуры определить последовательность квадратных матриц

$$D_1, D_2, \ldots, D_{m-1}, D_m, \ldots, D_n$$

возрастающей размерности. Размерность матрицы D_m равна $m \times m$. Элементы $l^m_{i,j} (1 \le i, j \le m)$ матрицы D_m вычисляются при $i \ne j$ через элементы матриц D_{v-1} и D_0 по формулам (6.26) для

$1 \le k \le m - 1$, по формулам (6.28) — для $j = m$, по формулам (6.29) — для $i = m$. Наконец, при $i = j$ полагается[1] $l^m_{i,\,i} = 0$.

Шаг 3. Последний шаг рекуррентной процедуры 2 дает $(n \times n)$-матрицу D_n длин кратчайших путей между всеми парами вершин графа G.

Алгоритм Данцига имеет такую же сложность, как алгоритм Флойда — $\theta(n^3)$.

Задания

1. С помощью алгоритма Дейкстры построить остовное дерево кратчайших путей из вершины x_1 в графе G на рис. 6.39. Каждое неориентированное ребро с весом c_j рассматривается как пара противоположно направленных дуг с одинаковыми весами c_j.

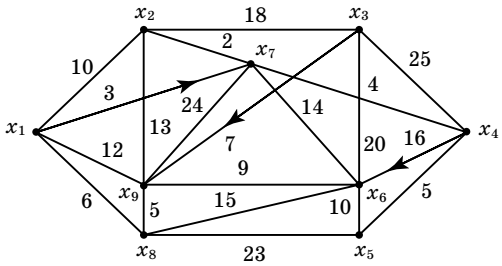


Рис. 6.39. Граф G для задания 1
(построение остовного дерева алгоритмом Дейкстры)

2. Дать полное описание алгоритма Форда по шагам, не опираясь на алгоритм Дейкстры.
3. Оценить информационный объем формирования «кратчайших» (i, j)-путей на каждой m-ой итерации алгоритма Флойда. Рассмотреть пример рис. 6.40.

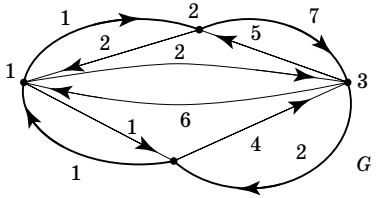


Рис. 6.40. Граф G для задания 5
(поиск кратчайших путей алгоритмом Флойда)

[1] В частности, всегда одномерная матрица D_1 есть число нуль: $D_1 = 0$.

4. Показать, что для формирования всех кратчайших (i, j)-путей в графе G достаточно знать лишь предпоследнюю вершину x_{ij} кратчайшего (i, j)-пути для всех пар вершин (i, j) на каждом шаге алгоритма Флойда.
5. Используя алгоритм Флойда, найти кратчайшие пути между всеми парами вершин графа G на рис. 6.40.
6. Найти кратчайшие пути между всеми парами вершин графа G на рис. 6.41, используя алгоритм Данцига.

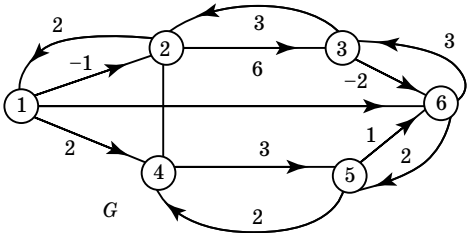


Рис. 6.41. Граф G для заданий 6 и 7

7. Сравнить работу алгоритмов Флойда и Данцига поиска кратчайших путей в графе G на рис. 6.41.
8. Влияет ли выбор нумерации вершин графа на эффективность алгоритмов Флойда и Данцига? Если влияет, то почему?
9. На нефтеперерабатывающем заводе имеется четыре емкости (A, $Б$, B, $Г$), и скорости перекачки нефти из одной емкости в другую указаны в таблице 6.4. Найти два наилучших способа перекачки нефти из емкости A в емкость $Г$.

Таблица 6.4. Скорости перекачки нефти из 4-х емкостей

Емкость	*A*	*Б*	*B*	*Г*
A	0,00	0,13	0,14	0,15
Б	0,08	0,00	0,13	0,08
B	0,17	0,12	0,00	0,18
Г	0,10	0,06	0,13	0,00

6.11. Гамильтоновы циклы и пути. Задача коммивояжера

Истоки и определения

Гамильтоновым циклом в графе $G = (X, Y)$ называется *простой цикл* $Q = (X, Y_0)$, *содержащий все вершины графа*, независимо от того, является ли G ориентированным. Требование простоты цикла является принципиальным: по гамильтоновому

циклу Q можно обойти все вершины x графа G, посещая каждую промежуточную (то есть не начальную и не конечную) вершину только один раз. Сам граф G, в котором существует гамильтонов цикл, называется ***гамильтоновым графом***. Ясно, что гамильтонов граф является связным и, более того, *двусвязным*, так как между каждой парой вершин существует не менее чем две различные простые цепи. При любом $n \geq 3$ полный простой граф K_n является гамильтоновым. Полный двудольный граф $K_{p,p}$ с равномощными долями (одноцветными множествами) также гамильтонов, см. $K_{2,2}$, $K_{3,3}$, K_5 на рис. 6.42.

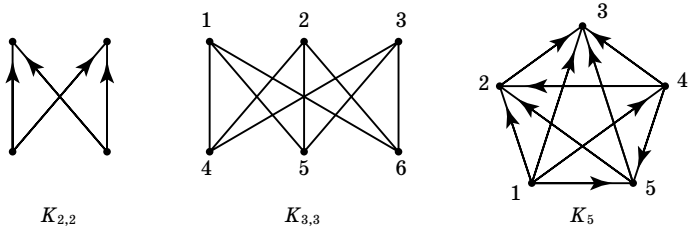


Рис. 6.42. Гамильтоновы графы $K_{2,2}$, $K_{3,3}$, K_5

Непосредственно проверяется, что графы G_1, G_2, G_3 на рис. 6.43 не содержат гамильтоновых циклов и, следовательно, не являются гамильтоновыми графами. Однако все эти графы содержат гамильтоновы цепи. ***Гамильтоновой цепью*** в графе $G = (X, Y)$ называется *простая цепь* $Z = (X, Y_z)$, *содержащая все вершины графа*. Граф, обладающий гамильтоновой цепью, называется ***трассируемым***. Гамильтоновой цепью в графе G_1 является цепь 23514, в графе G_2 — цепь 623451, в графе G_3 — цепь 643215, а также 264315. Отметим, что граф G_3 не является двусвязным, так что для трассируемости требование двусвязности графа G излишне.

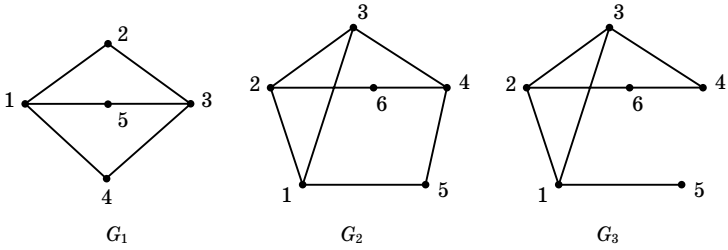


Рис. 6.43. Гамильтоновы цепи

На рис. 6.44 изображены два негамильтоновых графа $K_{2,3}$, $K_{2,4}$. Оба они двусвязны. В $K_{2,3}$ существуют гамильтоновы цепи, например, (4, 1, 5, 2, 3), (3, 1, 4, 2, 5), (4, 2, 5, 1, 3). В графе $K_{2,4}$ нет ни гамильтоновых циклов, ни гамильтоновых цепей.

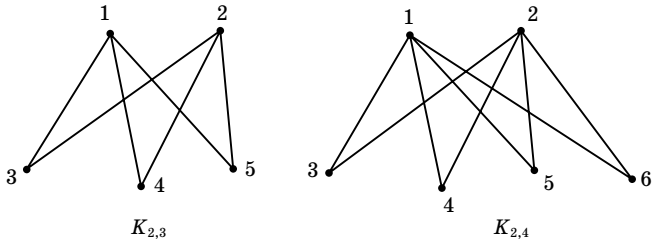


Рис. 6.44. Негамильтоновы графы

Негамильтоновы графы $G_{1,...}$ $K_{2,3}$ содержат в качестве подграфов так называемые *тэта-графы*, имеющие хотя бы две вершины степени 3, соединенные тремя простыми попарно различными цепями длины не менее двух. Например, в графе G_1 цепи (1, 2, 3), (1, 5, 3), (1, 4, 3) образуют тэта-подграф с вершинами 1, 3 степени 3. Справедлива

Теорема 6.10

Если двусвязный граф не содержит гамильтонового цикла, то он содержит тэта-подграф.

□ Пусть C — простой цикл максимальной длины в нашем графе $G = (X, Y)$. По условию цикла Q не может быть гамильтоновым, поэтому число его вершин должно быть меньше, чем $n = |X|$. Легко убедиться, что при $n = 3$ или $n = 4$ двусвязный простой граф G является гамильтоновым, поэтому $n \geq 5$. Число вершин X_Q цикла Q (и ребер) не менее 4: $|X_Q| \geq 4$. Действительно, если в двусвязном графе G есть цикл длины 3, то к нему можно добавить цепь с новой промежуточной вершиной так, чтобы возник простой цикл длины большей, чем 3 (см. рис. 6.43, G_1). Существует такое ребро $(x, v) \in Y$, что $x \in Q$, $v \in Y \backslash Q$. Обозначим через a, b вершины цикла Q, смежные с вершиной x (см. рис. 6.45). Так как цикл Q максимальный, то вершина v не смежна ни с вершиной a, ни с вершиной b: в противном случае можно построить цикл большей длины.

Удалим из графа G вершину x вместе с инцидентными ей ребрами, к множеству которых относится и ребро (x, v). В оставшемся графе для каждой вершины w на цикле Q (кроме x) существует

простая (v, w) — цепь P_w, соединяющая вершины w и v. Среди всех таких цепей $P_w (w \in C, w \neq x)$ выберем кратчайшую цепь P_c, соединяющую вершину v с некоторой вершиной $c \in Q$. Очевидно, $c \neq a$ и $c \neq b$, иначе цикл Q не был бы максимальным простым циклом в графе G. Цепь P_c не содержит вершин цикла Q, отличных от c, ибо в противном случае P_c не есть кратчайший путь из всех цепей P_w. Построим в графе G подграф $G_0 = Q \cup (x, v) \cup P_c$, объединяющий цикл Q, ребро (x, v) и цепь P_c вместе с инцидентными вершинами. Ясно, что G_0 есть тэта-граф, в котором вершины третьей степени x и c соединяются тремя различными простыми цепями. ■

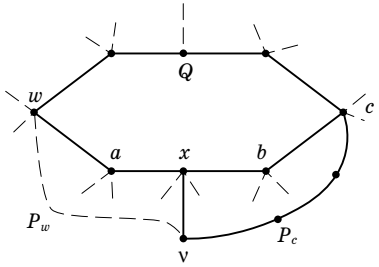


Рис. 6.45. Граф, содержащий тэта-подграф

Замечание. Существование тэта-подграфа является лишь необходимым условием для негамильтоновости двусвязного графа. В двусвязном графе $K_{3,3}$ рис. 6.42 существуют тэта-подграфы, однако $K_{3,3}$ — гамильтонов. Аналогично граф G_2 на рис. 6.43 также имеет гамильтонов цикл (6, 2, 3, 1, 5, 4, 6) и одновременно имеет тэта-подграф с тремя цепями (2, 3, 4), (2, 6, 4), (2, 1, 5, 4) между вершинами 2, 4 третьей степени. С другой стороны, требование двусвязности в теореме 6.10 существенно: негамильтоновый граф G_3 на рис. 6.43 односвязен и не содержит ни одного тэта-подграфа.

Постановка задачи о гамильтоновом цикле с однократным прохождением всех вершин графа может показаться похожей или в каком-нибудь смысле двойственной к постановке задачи об эйлеровом цикле с однократным прохождением всех ребер графа. Однако это не так. Задача о гамильтоновом цикле, к сожалению, не имеет до настоящего времени ни полного теоретического решения, ни удовлетворительного алгоритма отыскания цикла для не очень маленьких n. Как мы видели в разделе п. 6.3, и теоретическое решение, и хорошие алгоритмы имеются для эйлерова цикла. Исторически первым рассматривал задачу обхода простым циклом всех вершин графа известный ирландский математик У. Гамильтон в 1859 г.

Он построил ряд таких циклов на ребрах додекаэдра — правильного выпуклого многогранника с 20 вершинами и 12 пятиугольными гранями. Плоское изображение додекаэдра, которое можно трактовать как его проекцию на одну «растянутую» грань, изображено на рис. 6.46. Один из гамильтоновых циклов изображен на рисунке жирной ломаной. Гамильтон трактовал свою задачу в виде игры «Кругосветное путешествие», в которой предлагается выбрать маршрут посещения двадцати городов на земном шаре, двигаясь по дорогам-ребрам додекаэдра. Он даже продал свою игру торговцу игрушками за 25 гиней.

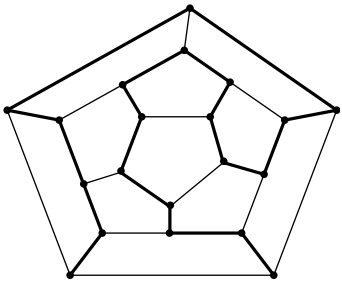


Рис. 6.46. Плоское изображение додекаэдра

В ориентированном графе наряду с гамильтоновым циклом или гамильтоновой цепью часто приходится искать *гамильтонов контур* или *путь*, обход которых осуществляется только по *направлениям дуг*. ***Гамильтонов путь*** *— это простой* (u, v)*-путь в графе* G*, содержащий все вершины графа.* Если $u = v$, то это — ***гамильтонов контур***. Всякий гамильтонов путь есть гамильтонова цепь, обратное, вообще говоря, неверно. Графы $K_{2,2}$ и K_5 на рис. 6.42 являются гамильтоновыми без учета ориентации дуг, однако они не обладают гамильтоновыми контурами. При этом граф K_5 имеет гамильтонов путь, в котором последовательность прохождения вершин есть 1, 4, 5, 2, 3. В графе $K_{2,2}$ рис. 6.42 гамильтонов путь отсутствует.

Подобная ситуация невозможна в симметрическом графе: каждой гамильтоновой цепи (циклу) неориентированного графа отвечает пара противоположно направленных гамильтоновых путей (контуров) в соответствующем ориентированном симметрическом графе.

Приложения гамильтоновых цепей и путей весьма многочисленны. В *теории расписаний* отдельные процедуры (действия) изо-

бражаются точками — вершинами x_1, x_2, ..., x_n, а из вершины x_i, ведет дуга в x_j, если после выполнения процедуры x_i допустим переход к выполнению процедуры x_j. Гамильтонов путь в таком модельном графе задает «расписание» последовательного выполнения всех «n» процедур.

В *исследовании операций* типичной является следующая ситуация. Имеется техническое устройство (станок, компьютер и т.п.) и n операций (задач), каждую из которых устройство способно осуществить (решить) *после выполнения соответствующей перенастройки (перепрограммирования)*. Точнее, при переходе от i-ой задачи к j-ой задаче перепрограммирование устройства требует затраты t_{ij} единиц времени или другого ресурса. Необходимо найти последовательность выполнения заданий, при которой время каждого перепрограммирования не превосходит заданного значения τ. Строится простой модельный граф с множеством вершин $\{1, 2, ..., n\}$, в котором дуга (i, j) присутствует тогда и только тогда, когда $t_{ij} \le \tau$. Любой гамильтонов путь в модельном графе задает искомую последовательность выполнения заданий.

Рассмотрим известную *задачу о книгах*, в которой принцип моделирования с помощью графа является весьма поучительным и может применяться для решения других задач. Для изготовления n книг печатник располагает двумя машинами: печатной и переплетной. На печатание k-ой книги расходуется a_k времени, на ее переплет расходуется b_k времени. Понятно, что в переплет k-ая книга поступает после ее печатания, так что некоторое время в начале работы переплетный станок может простаивать. Найти порядок работы с книгами, при котором время исполнения всего заказа на n книг будет минимальным. Нетрудно увидеть, что, имея оптимальный порядок работы, можно перестроить его без уменьшения полного времени выполнения заказа так, чтобы книги поступали на переплетную машину в той же последовательности, в какой они поступают на печатную машину. Следовательно, оптимальный порядок изготовления множества книг $(x_1, x_2, ..., x_n)$ определяется некоторой перестановкой $(x_{i_1}, x_{i_2}, \ldots, x_{i_n})$. Можно показать, что при оптимальном порядке i-ая книга должна быть пущена в печать раньше κ-ой книги (не обязательно подряд), если и только если $\min\{a_i, b_k\} \le \min\{a_k, b_i\}$. Теперь ясно, как построить модельный граф. На множестве вершин $\{x_1, x_2, ..., x_n\}$ проводится дуга (x_i, x_k), если выполнено указанное неравенство. Гамильтонов путь в этом графе указывает искомый оптимальный порядок изготовления книг.

Признаки существования гамильтоновых циклов, путей и контуров

Получение условий гамильтоновости графов является популярной задачей, до конца не решенной до настоящего времени. Популярность питается не только прикладной ценностью условий, но и редкой простотой и естественностью самой постановки задачи о гамильтоновом цикле, не требующей предварительных математических знаний и символьных обозначений в формулировке.

Полный граф имеет гамильтонов цикл, поэтому, выражаясь нестрого, качественно можно предположить, что чем больше ребер в графе и чем более «равномерно» они распределены, тем выше вероятность существования гамильтонова цикла. Следующие достаточные условия гамильтоновости графа подтверждают это предположение.

Все графы предполагаются *связными и простыми*.

Теорема 6.11 *(Г. Дирак, 1952 г.)*

Если число n вершин графа G не менее трех и степень $\delta_i = \deg x_i$ *любой вершины* x_i *не менее* $\frac{n}{2}$ $(\delta \geq \frac{n}{2})$, *то граф G является гамильтоновым*.

Сформулированный признак Дирака является очевидным следствием более общего признака гамильтоновости, установленного в 1960 г. Оре.

Теорема 6.12 *(О. Оре, 1960 г.)*

Если в графе G с n вершинами $(n \geq 3)$ *сумма степеней любых двух вершин u, v является не меньшей, чем n* $(\deg u + \deg v \geq n)$, *то граф G гамильтонов*.

В свою очередь, признак гамильтоновости Оре можно вывести из более «поздних» признаков Л. Поша и В. Хватала.

Теорема 6.13 *(В. Хватал, 1972 г.)*

Пусть для упорядоченной по возрастанию последовательности степеней вершин $\delta_i = \deg x_i$

$$\delta_1 \leq \delta_2 \leq \ldots \leq \delta_n \qquad (6.29)$$

графа G выполнены импликации

$$(\delta_k \leq k) \Rightarrow (\delta_{n-k} \geq n-k), \quad \forall k : 1 \leq k < \frac{n}{2},$$

тогда G — гамильтонов граф.

Существует ряд признаков гамильтоновости графов, являющихся производными от некоторых других графов, в частности, для степеней G^p и реберных графов $L(G)$. Напомним, что вершины реберного (или дуального) графа $L(G)$ находятся во взаимно однозначном соответствии с ребрами графа G, а две вершины в $L(G)$ соединены ребром, если и только если соответствующие ребра в G смежны. Степень G^p графа $G = (X, Y)$ здесь понимается как граф с тем же множеством вершин X, в котором вершины $w, v \in X$ соединены ребром (смежны), если и только если в G расстояние между w и v не больше p: $d(w, v) \le p$ в графе G. Например, если C_5 — цикл с пятью вершинами и пятью ребрами, то $(C_5)^2 = K_5$ — полный пятивершинный граф. Этот пример и само определение G^p подсказывает, что у любого связного графа G некоторая степень G^p должна быть гамильтоновым графом. Из гамильтоновости степени G^p следует гамильтоновость G^{p+1}.

Какова нижняя грань степеней p, обеспечивающих гамильтоновость G^p?

Теорема 6.14 *(Д. Караганис, 1968 г.)*

Для связного графа G с числом вершин $n \ge 3$ степень G^3 является гамильтоновым графом.

Теорема 6.15 *(Г. Флейшнер, 1971 г.)*

Если G — двусвязный граф с числом вершин $n \ge 3$, то G^2 — гамильтонов граф.

Теорема 6.16 *(Ф. Харари, С. Нэш-Вильямс, 1965 г.)*

Реберный граф $L(G)$ гамильтонов тогда и только тогда, когда в G существует цикл, содержащий хотя бы по одной вершине из каждого ребра графа G.

Следствие. *Если граф G либо эйлеров, либо гамильтонов, то реберный граф $L(G)$ гамильтонов.*

Какова разница между задачами о поиске гамильтонового цикла и поиске гамильтоновой цепи в графе? С одной стороны, гамильтонов граф всегда содержит незамкнутую гамильтонову цепь. С другой стороны, примеры графов G_1, G_3 на рис. 6.43 свидетельствуют, что гамильтоновы цепи могут существовать в негамильтоновом графе, так что класс гамильтоновых графов содержится в классе трассируемых графов, не совпадая с ним. Тем не менее, указанные две задачи имеют одинаковую трудность, а любой трассируемый граф G весьма просто вкладывается в некоторый гамильтонов

граф G_v, множеством вершин которого на одну вершину v превосходит множество вершин исходного графа G. Именно, если (a, b)-цепь Z является гамильтоновой цепью в трассируемом графе $G = (X, Y)$, то добавление одной новой вершины v и двух новых ребер (a, v), (b, v) приводит к графу $G_v = (X \cup (b, v), Y \cup v \cup (v, a))$ с гамильтоновым циклом $Z \cup (b, v) \cup (v, a)$.

Например, негамильтонов трассируемый граф G_1 на рис. 6.43 является подграфом гамильтонового графа G_v на рис. 6.47, где добавленные ребра изображены пунктиром. При этом пара добавленных ребер (b, v), (v, a) «замыкает» конкретную гамильтонову цепь Z в графе G. Чтобы не связывать вложение трассируемого графа в гамильтонов с определенной гамильтоновой цепью, часто новую вершину v соединяют со всеми вершинами исходного графа G и получают граф G_{dv} с *доминирующей вершиной* v. Тогда любая гамильтонова цепь в G дополняется до соответствующего гамильтонового цикла в G_{dv} (рис. 6.47).

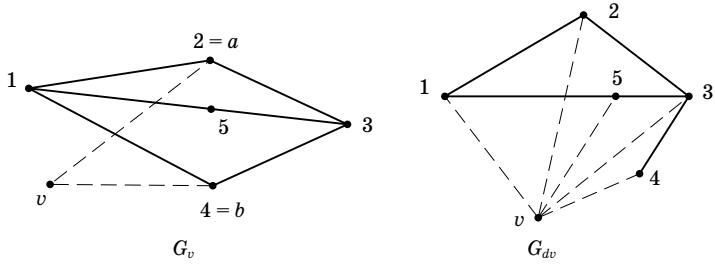


Рис. 6.47. Дополнение до гамильтонового графа в G_v и до гамильтоновых циклов в G_{dv}

Признаки существования *гамильтоновых путей* и *контуров* в ориентированном графе есть смысл рассматривать только для *несимметричных орграфов*. Одним из простейших является признак Кенига.

Теорема 6.17 *(Кенига)*

В полном орграфе G (любая пара вершин которого соединяется хотя бы в одном направлении) всегда существует гамильтонов путь[1].

□ Выделим в G простой (x_1, x_r)-путь Z длины $(r - 1)$ с последовательностью попарно различных вершин $[x_1, x_2, \ldots, x_r]$ и покажем,

[1] Но не гамильтонов контур (см. орграф K_5, рис. 6.42).

что с его помощью можно получить простой путь длины r с попарно различными вершинами. Пусть v — вершина графа $G = (X, Y)$, принадлежащая пути Z. Предположим, что для любого $k \in \{1, 2, 3, \ldots, r-1\}$ не существует пути в G длины r с последовательностью вершин $[x_1, x_2, \ldots, x_k, v, x_{k+1}, \ldots, x_r]$. Тогда при каждом k справедливы импликации

$$(x_k, v) \in Y \Rightarrow (v, x_{k+1}) \notin Y \Rightarrow (x_{k+1}, v) \in Y, \qquad (6.30)$$

причем последняя импликация вытекает из полноты графа G. Если существует дуга $(v, x_1) \in Y$, то мы сразу получаем искомый путь длины r с последовательностью вершин $[v, x_1, x_2, \ldots, x_r]$. Если дуга (v, x_1) отсутствует в G, то существует дуга (x_1, v), и в силу (6.30) последовательно получаем, что граф G содержит дуги $(x_2, v) \in Y$, $(x_3, v) \in Y$, ..., $(x_r, v) \in Y$. Но тогда существует простой путь длины r с последовательностью попарно различных вершин $[x_1, x_2, \ldots, x_r, v]$ (рис. 6.48).

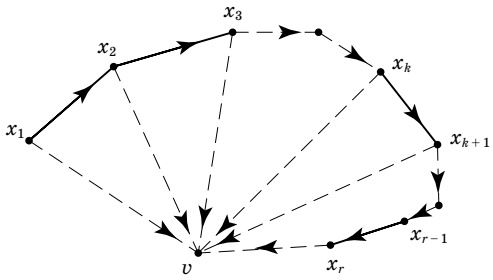


Рис. 6.48. Последовательность попарно различных вершин

Повторяя процедуру «удлинения» пути конечное число раз, получим простой путь длины $n - 1$ ($n = |X|$), содержащий все вершины из X. Это и есть искомый гамильтонов путь. ■

Следствие. *Участников турнира, в котором происходит встреча каждой пары и по правилам невозможны ничейные результаты, всегда можно после турнира упорядочить (ранжировать) так, чтобы каждый предыдущий был победителем непосредственно следующего.*

Отметим некоторые алгебраические свойства графов, обладающих гамильтоновым путем или гамильтоновым контуром. Для произвольного подмножества S вершин орграфа $G = (X, Y)$ определим множество S_+ всех тех вершин v, в которые ведут дуги из вершин множества S:

$$S_+ = \{v \in X: (x, v) \in Y, \forall x \in S\}.$$

Дефицитом простого графа $G = (X, Y)$ называется число

$$\delta_0 = (\max_{S \subset X} |S| - |S_+|). \qquad (6.31)$$

Как обычно, $|S|$ обозначает число элементов множества S.

Следующая теорема содержит алгебраические условия, выполнение которых необходимо для существования гамильтонового контура и гамильтонового пути в орграфе.

Теорема 6.18

Если в графе существует гамильтонов контур, то дефицит графа (6.31) равен нулю: $\delta_0 = 0$. *Если существует гамильтонов путь, то* $0 \le \delta_0 \le 1$.

Задача коммивояжера

Общая[1] задача коммивояжера состоит в следующем: используя заданную систему транспортных сообщений (дорог и т.п.) между пунктами (городами, фирмами и т.п.) в конкретной зоне обслуживания, посетить все пункты в такой последовательности, чтобы пройденный маршрут был кратчайшим из всех возможных.

На языке теории графов или сетей общая задача коммивояжера имеет следующую формулировку: во взвешенном связном графе G найти кратчайший маршрут, проходящий через все вершины графа. В постановке задачи возможно дополнительное требование *замкнутости* маршрута коммивояжера (возвращение коммивояжера в пункт исходного пребывания). Понятно, что в любом связном графе G общая задача коммивояжера (и замкнутая и незамкнутая) всегда имеет решение.

Существует еще одна постановка, в которой *дополнительно* к предыдущей задаче требуется, чтобы каждый пункт обслуживания коммивояжер посещал *только один раз*. Очевидно, в такой постановке задача коммивояжера не всегда имеет решение, а если имеет, то маршрут коммивояжера в графе G является кратчайшей гамильтоновой цепью или циклом. В связи со сказанным будем называть последнюю постановку ***«гамильтоновой» задачей коммивояжера*** (в отличие от «общей»). Если ищется *замкнутый* маршрут, то для разрешимости «гамильтоновой» задачи коммивояжера необходимо

[1] В отличие от «гамильтоновой» задачи коммивояжера.

и достаточно, чтобы граф G был гамильтоновым, для незамкнутого гамильтонового маршрута — граф G должен быть *трассируемым*.

В *ориентированном* взвешенном графе G встречается также задача поиска ***общего ориентированного маршрута коммивояжера*** — кратчайшего ориентированного маршрута, содержащего все вершины графа, и соответственно, ***ориентированного пути*** или ***контура коммивояжера*** (содержащего каждую вершину один раз). Замкнутый ориентированный (или неориентированный) маршрут коммивояжера при общей негамильтоновой постановке задачи не обязательно является гамильтоновым контуром (соответственно гамильтоновым циклом). Например, граф G на рис. 6.49 имеет один гамильтонов контур и несколько гамильтоновых циклов, все их длины равны 24, и каждый содержит три дуги. Однако кратчайшим замкнутым ормаршрутом коммивояжера (непростым) является замкнутый маршрут с четырьмя дугами (a, b), (b, a), (a, c), (c, a), который проходит через каждую вершину дважды и имеет длину 8.

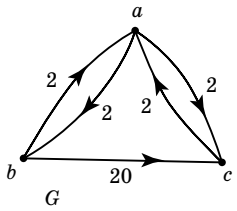


Рис. 6.49. (a, b), (b, a), (a, c), (c, a) — кратчайший замкнутый ормаршрут

В каких случаях гамильтонов контур (или цикл) является решением общей задачи коммивояжера?

Теорема 6.19

Если функция $d(x, v)$ весов (длин) ребер (или дуг) между парами вершин x, v в графе $G = (X, Y)$ с числом вершин $n = |X| \geq 3$ удовлетворяет неравенству треугольника

$$d(x, v) \leq d(x, z) + d(z, v), \quad \forall z \neq x,\ z \neq v,\ z \in X, \qquad (6.32)$$

и существует решение общей задачи коммивояжера, то существует также решение «гамильтоновой» задачи коммивояжера.

Условие треугольника означает, что в графе G длина «одношагового» пути[1] (цепи) между вершинами x и v, конечная или бесконечная, *не превышает* длины любого «двухшагового» (x, v)-пути ((x, v)-цепи) с одной промежуточной вершиной.

□ Пусть Q — оптимальный замкнутый маршрут коммивояжера, для определенности — ориентированный, и предположим,

[1] То есть пути (цепи) $l(x, v)$ без промежуточных вершин между x и v; он состоит либо из ребра (x, v), либо отсутствует.

Q не является гамильтоновым контуром в графе G. Тогда существует вершина $z \in X$, которая повторяется при обходе контура Q, по крайней мере, дважды. Предположим, при первом прохождении коммивояжер попадает в вершину z из предшествующей вершины x, а выходит из z в последующую вершину v. По условию теоремы, должен существовать «одношаговый» путь $l(x, v)$ по дуге (x, v), который не длиннее двухшагового пути (x, z, v) (рис. 6.50). Следовательно, в маршруте Q два однократных прохождения дуг (x, z), (z, v) можно заменить одним прохождением дуги (x, v).

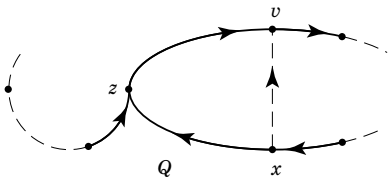


Рис. 6.50. Замена двух однократных прохождений дуг (x, z), (z, v) на одно прохождение дуги (x, v)

В результате получается замкнутый маршрут коммивояжера Q_1 длины $d(Q_1) \le d(Q)$, с числом шагов, то есть числом прохождений дуг, меньшим, чем в маршруте Q. При этом число прохождений вершины z уменьшается на единицу (точнее, на один заход и один выход), а число прохождений остальных вершин графа сохраняется. Применяя этот процесс к маршруту Q_1, мы построим замкнутый ормаршрут коммивояжера Q_2 длины $d(Q_2) \le d(Q_1)$ с числом прохождений дуг, меньшим, чем у Q_1, и т.д. После конечного числа шагов мы получим замкнутый маршрут коммивояжера Q_p, в котором каждая вершина встречается в точности один раз. Следовательно, Q_p есть гамильтонов контур в графе G. ■

Если граф G не удовлетворяет неравенству треугольника (6.32), то формально общую задачу коммивояжера необходимо решать отдельно от гамильтоновой задачи коммивояжера. Однако фактически *можно решение общей задачи сводить к решению «гамильтоновой»*. Для этого достаточно простым преобразованием перестроить граф G и получить граф G_t, удовлетворяющий неравенству (6.32) посредством замены в G длины $d(x, v)$ каждой дуги (x, v), не удовлетворяющей (6.32), на длину кратчайшего пути из x в v. Пусть Q_t — решение «гамильтоновой» задачи коммивояжера в графе G_t. Каждую дугу[1] (v, w) из гамильтонового контура Q_t,

[1] Включая отсутствовавшую дугу с формальной длиной ∞.

длина которой была ранее уменьшена при переходе $G \to G_t$, следует заменить кратчайшим (v, w)-путем из графа G. Полученный таким образом маршрут Q является оптимальным решением общей задачи коммивояжера в графе G.

Например, в графе G на рис. 6.49 дуга (b, c) не удовлетворяет условию (6.32): $20 > 2 + 2$. Поэтому следует заменить длину 20 дуги (b, c) на длину $4 = 2 + 2$ кратчайшего (b, c)-пути (b, a, c). В полученном взвешенном графе G_t (топологически он совпадает с G, отличаются лишь веса дуги (b, c)) гамильтонов контур (a, b, c, a) имеет длину $2 + 4 + 2 = 8$ и является оптимальным маршрутом коммивояжера в графе G_t. Заменяя в Q_t «укороченную» дугу (b, c) кратчайшим (b, c)-путем (b, a, c), получаем оптимальный в графе G ормаршрут коммивояжера (a, b, a, c, a) длины 8, включающий четыре дуги и не являющийся гамильтоновым.

Практические методы и алгоритмы

В самом неблагоприятном случае (полный граф и фатальное невезение) оптимальную гамильтонову цепь или цикл в графе G, начинающуюся в заданной вершине x_1, можно построить путем всевозможных перестановок на множестве остальных вершин $\{x_2, x_3, \ldots, x_n\}$. Значит, вычислительная сложность задач гамильтона и коммивояжера имеет порядок $(n - 1)!$ Вследствие такой значительной вычислительной сложности для решения задачи коммивояжера создано много алгоритмов, как автономных, так и базирующихся на других оптимизационных алгоритмах, — потоковых, строящих минимальный остов и др. При этом точные алгоритмы, гарантирующие получение маршрута коммивояжера в любом случае, являются трудоемкими и справляются с сетями не очень высокой размерности. Приближенные алгоритмы в большинстве случаев справляются с более громоздкими сетями, однако иногда приводят к неоптимальному маршруту [10, 42, 47]. Рассмотрим вкратце два алгоритма.

Алгебраический алгоритм Йоу, Даниэльсона, Дхавана

В алгоритме последовательно строятся все простые пути простого орграфа G с помощью последовательного перемножения $(n \times n)$-матриц, содержащих символы вершин $x_1, x_2, \ldots, x_n$. Пусть $A = \{a_{ij}\}_1^n$ — матрица смежности орграфа G. Через $V = \{v_{ij}\}_1^n$ обозначим так называемую *модифицированную матрицу смежности графа* G, в которой $v_{ij} = x_j$, если существует дуга из x_i в x_j, и $v_{ij} = 0$ — если

такой дуги нет. Иначе говоря, на месте единиц в матрице смежности A ставятся символы x_j вершин, в которые заходят соответствующие строкам дуги:

$$a_{ij} = 1 \Leftrightarrow v_{ij} = x_j; \qquad a_{ij} = 0 \Leftrightarrow v_{ij} = 0.$$

«*Внутренним произведением вершин*» пути $(x_1, x_2, x_3, \ldots, x_{k-1}, x_k)$ называется формальное алгебраическое выражение (слово) $x_2 \cdot x_3 \cdot \ldots \cdot x_{k-1}$, не содержащее начальной и конечной вершин пути. При $k = 2$ произведение считается равным 1.

Предлагаемый алгоритм строит последовательность матриц

$$P_1, P_2, \ldots, P_{n-1}, V \cdot P_{n-1},$$

где $P_s = \{p_s(i, j)\}_{i, j=1}^{n}$ — матрица, элемент которой $p_s(i, j)$ равен сумме внутренних произведений *всех простых путей* из x_i в x_j длины $s (1 \le s \le n-1)$, если $i \ne j$ и $p_s(i, i) = 0$. Ясно, что $P_1 = A$ (матрица смежности). Если матрица P_s уже вычислена, то для получения P_{s+1} сначала строится матрица

$$P_{s+1}^0 = V \cdot P_s = \{p_{s+1}^0(i, j)\}, \quad p_{s+1}^0(i, j) = \sum_k \mathrm{V}_{ik} \cdot p_s(k, j). \qquad (6.33)$$

Ее элемент $p_{s+1}^0(i, j)$ равен сумме внутренних произведений *всех таких цепей* (не только простых!) из вершины x_i в вершину x_j длины $s + 1$, среди которых *непростыми* являются в точности те цепи, внутренние произведения которых содержат в сомножителях $p_s(k, j)$ из (6.33) вершину x_i. Исключив из суммы в (6.33) слагаемые, содержащие x_i, получаем алгебраическое выражение $p_{s+1}(i, j)$, равное сумме внутренних произведений всех *простых* (x_i, x_j) — путей длины $s + 1$, где $i \ne j$. Полагая $p_{s+1}(i, i) = 0$, получим искомую матрицу P_{s+1} всех простых незамкнутых путей длины $s + 1$.

Наконец, при $s = n - 1$ матрица P_{n-1} дает все *гамильтоновы пути* (имеющие длину $n - 1$) в графе G между всеми парами вершин. Гамильтоновы контуры получаются добавлением дуги (если она существует), соединяющей конец гамильтонового пути с его началом. Иначе говоря, гамильтоновы контуры перечисляются членами внутренних произведений вершин, содержащихся на диагональных элементах матрицы $V \cdot P_{n-1}$.

Проиллюстрируем работу алгоритма на примере орграфа G рис. 6.51 с множеством вершин $X = \{a, b, c, d, e\}$.

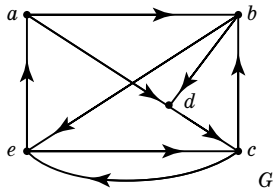


Рис. 6.51. Орграф G

Матрица смежности A и модифицированная матрица смежности V имеют вид:

$$A = \begin{array}{c} \\ a \\ b \\ c \\ d \\ e \end{array} \begin{array}{c} \begin{array}{ccccc} a & b & c & d & e \end{array} \\ \begin{bmatrix} 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 \end{bmatrix} \end{array}; \quad V = \begin{array}{c} \\ a \\ b \\ c \\ d \\ e \end{array} \begin{array}{c} \begin{array}{ccccc} a & b & c & d & e \end{array} \\ \begin{bmatrix} 0 & b & 0 & d & 0 \\ 0 & 0 & 0 & d & e \\ 0 & b & 0 & 0 & e \\ 0 & 0 & c & 0 & 0 \\ a & 0 & c & 0 & 0 \end{bmatrix} \end{array}.$$

Полагаем $P_1 \equiv A$. Вычисляя матрицу $P_2^0 = V \cdot P_1$ и заменяя в ней подчеркнутые диагональные элементы нулями, получаем P_2. Вычеркнутые с диагонали вершины e, c являются промежуточными во 2-контуре (c, e, c) и (e, c, e) соответственно.

$$P_2^0 = \begin{array}{c} \\ a \\ b \\ c \\ d \\ e \end{array} \begin{array}{c} \begin{array}{ccccc} a & b & c & d & e \end{array} \\ \begin{bmatrix} 0 & 0 & d & b & b \\ e & 0 & d+e & 0 & 0 \\ e & 0 & \underline{e} & b & b \\ 0 & c & 0 & 0 & c \\ 0 & a+c & 0 & a & \underline{c} \end{bmatrix} \end{array}; \quad P_2 = \begin{array}{c} \\ a \\ b \\ c \\ d \\ e \end{array} \begin{array}{c} \begin{array}{ccccc} a & b & c & d & e \end{array} \\ \begin{bmatrix} 0 & 0 & d & b & b \\ e & 0 & d+e & 0 & 0 \\ e & 0 & 0 & b & b \\ 0 & c & 0 & 0 & c \\ 0 & a+c & 0 & a & 0 \end{bmatrix} \end{array}.$$

Далее находим $P_3^0 = V \cdot P_2$ и после замены диагональных элементов нулями — матрицу простых 3-путей P_3

$$P_3^0 = \begin{array}{c} \\ a \\ b \\ c \\ d \\ e \end{array} \begin{array}{|c|c|c|c|c|} \hline a & b & c & d & e \\ \hline \underline{be} & dc & bd+be & 0 & dc \\ \hline 0 & \underline{dc+ea+ec} & 0 & ea & dc \\ \hline be & ea+\underline{ec} & \underline{bd+be} & ea & 0 \\ \hline ce & 0 & 0 & \underline{cb} & cb \\ \hline \underline{ce} & 0 & ad & ab+cb & \underline{ab+cb} \\ \hline \end{array};$$

$$P_3 = \begin{array}{c} \\ a \\ b \\ c \\ d \\ e \end{array} \begin{array}{|c|c|c|c|c|} \hline a & b & c & d & e \\ \hline 0 & dc & bd+be & 0 & dc \\ \hline 0 & 0 & 0 & ea & dc \\ \hline be & ea & 0 & ea & 0 \\ \hline ce & 0 & 0 & 0 & cb \\ \hline 0 & 0 & ad & ab+cb & 0 \\ \hline \end{array}.$$

Вычеркнутые в матрице P_3^0 диагональные элементы отвечают 3-контурам, которые по очевидной причине не могут быть далее

частями простых 4-путей. Вычеркнутые в P_3^0 элементы ec, ce на местах (3, 2), (5, 1) отвечают непростым путям (c, e, c, b), (e, c, e, a) соответственно, точнее, вершины c, e являются внутренними (промежуточными) в этих путях. В матрице P_3 значение $(bd + be)$ элемента (1, 3) означает, что последовательность вершин $\{bd\}$ является внутренней в простом 4-пути (a, b, d, c) и последовательность вершин $\{b, e\}$ является внутренней в простом 4-пути (a, b, e, c).

В матрице $P_4^0 = V \cdot P_3$ подчеркнутые диагональные элементы отвечают 4-контурам в графе G, а подчеркнутые недиагональные элементы — непростым 4-путям, содержащим контур длины 3 или 2 (длину контура указывает расстояние от начальной вершины пути до места ее повторения во внутренней последовательности). После замены в P_3^0 подчеркнутых элементов нулями получается матрица P_4 всех гамильтоновых путей графа G, так как здесь $4 = n - 1$. Общее количество гамильтоновых 4-путей равно 10: это число всех ненулевых слагаемых в матрице P_4. Если гамильтоновы пути (a, b, d, c, e) и (a, d, c, b, e), соответствующие элементу (1, 5) матрицы P_4, дополнить дугой (e, a), то мы получим два *гамильтоновых контура графа* G: $Q_1 = (a, b, d, c, e, a)$, $Q_2 = (a, d, c, b, e, a)$.

$P_4^0 =$

	a	b	c	d	e
a	$\underline{dce}$	0	0	$\underline{bea}$	$bdc+dcb$
b	dce	0	ead	$\underline{eab}+\underline{ecb}$	$\underline{dcb}$
c	0	0	$\underline{ead}$	$bea+\underline{ecb}+eab$	$\underline{bdc}$
d	cbe	cea	0	$\underline{cea}$	0
e	$\underline{cbe}$	$adc+\underline{cea}$	$abd+\underline{abe}$	$\underline{cea}$	$\underline{adc}$

;

$P_{n-1} = P_4 =$

	a	b	c	d	e
a	0	0	0	0	$bdc+dcb$
b	dce	0	ead	0	0
c	0	0	0	$bea+eab$	0
d	cbe	cea	0	0	0
e	0	adc	abd	0	0

.

Если любой из остальных восьми гамильтоновых 4-путей дополнить соответствующей дугой, получится один из двух контуров Q_1 или Q_2, так что других гамильтоновых контуров в графе G нет. Дополнения всех десяти незамкнутых гамильтоновых путей

до гамильтонового контура Q_1 или Q_2 перечисляются с помощью внутренних произведений вершин, стоящих на главной диагонали матрицы $V \cdot P_4 = \{\gamma_{ik}\}$. Например, элемент $\gamma_{22} = dcea + eadc$ отвечает второй вершине «b» и гамильтоновым контурам (b, d, c, e, a, b) и (b, e, a, d, c, b), из которых первый совпадает с Q_1, второй — с Q_2. Таким образом, *каждый диагональный элемент матрицы* $V \cdot P_{n-1}$ *перечисляет все гамильтоновы контуры графа* с помощью внутренних произведений, возникающих при обходе контуров с началом в той вершине, которой отвечает диагональный элемент γ_{ii}.

Если не требуется находить все незамкнутые гамильтоновы пути, а необходимо лишь перечислить *все гамильтоновы контуры графа*, то *алгебраический алгоритм* можно несколько упростить, сократив объем вычислений ориентировочно в n раз. А именно, достаточно вычислить лишь один диагональный элемент матрицы $V \cdot P_{n-1}$, например, γ_{11}, а для этого можно находить и хранить в памяти не все элементы матриц P^0_{n-1}, P_{n-1}; P^0_{n-2}, P_{n-2}; ...; P_1, *а лишь их первые столбцы.*

Замечание. После получения всех гамильтоновых незамкнутых путей или гамильтоновых контуров решение задачи коммивояжера[1] (соответственно незамкнутой или замкнутой) сводится к сравнению весов путей или контуров и нахождению минимального веса. Описанный алгебраический метод решения задач Гамильтона и коммивояжера всегда приводит к точному решению, однако является весьма трудоемким, предполагает использование вычислительных или программных средств высокого уровня, эффективно оперирующих с большими объемами символьных преобразований.

Задания

1. Подсчитать количества всех гамильтоновых циклов в графах K_4, K_5 и перечислить их.
2. При каких условиях полный двудольный граф $K_{p,q}$ является гамильтоновым? Когда $K_{p,q}$ является трассируемым?
3. Найти тэта-подграфы в негамильтоновых графах $K_{2,3}$, $K_{2,4}$ на рис. 6.44.
4. Пусть решетчатый граф G образован p горизонтальными линиями и q вертикальными. Каждая точка пересечения считается вершиной графа, а каждый отрезок между соседними точками

[1] Разумеется, в гамильтоновой постановке (см. переход к графу G_t в конце предыдущего раздела).

пересечения — ребром графа G. При каких p и q граф G является гамильтоновым?

5. Показать, что множество оптимальных решений задачи коммивояжера не зависит от того, какая из вершин выбирается в качестве начальной при обходе.

6.12. Потоки в сетях

В данном разделе будет рассматриваться взвешенный граф — граф, каждой дуге которого приписан поток некоторого вещества. Такой граф является удобной моделью при исследовании целого ряда задач в транспорте, связи и других областях, связанных с действительным или воображаемым движением товаров, информации или людей. Введем определения и обозначения, типичные для этого круга вопросов.

Сетью будем называть ориентированный связный граф без петель и параллельных ребер. (Заметим, что потоки в неориентированных графах можно представить в виде потоков в соответствующих ориентированных; потоки в несвязных графах могут изучаться покомпонентно; поток в петле не влияет на распределение потока между вершинами).

Рассмотрим сеть $G = (X, Y)$, $|X| = n$, $|Y| = m$. Пусть каждой дуге $y_j \in Y$ поставлено в соответствие неотрицательное вещественное число c_j, интерпретируемое как ***пропускная способность (емкость) дуги*** y_j. Обозначим через $x_i \to X$ множество дуг, исходящих из вершины x_i, через $X \to x_i$ — множество дуг, заходящих в вершину x_i.

Потоком в сети G ***из вершины*** x_s ***в вершину*** x_t ***величины*** v называется неотрицательная, определенная на дугах y_j, функция φ: $Y \to R_+ \cup \{0\}$, такая, что

$$\sum_{y \in x \to X} \varphi(y) - \sum_{y \in X \to x} \varphi(y) = \begin{cases} v, & x = x_s \\ 0, & x \neq x_s, x_t, \\ -v, & x = x_t \end{cases} \qquad (6.34)$$

$$\varphi(y_j) \le c_j,\ j = 1, \ldots, m. \qquad (6.35)$$

Вершина x_s называется ***источником***, вершина x_t — ***стоком***, а остальные вершины — ***промежуточными узлами***. Число $Q(x_i) = \sum_{y \in x_i \to X} \varphi(y) - \sum_{y \in X \to x_i} \varphi(y)$ называется ***чистым потоком*** из вершины x_i относительно φ. Число $\varphi(y)$ называется ***потоком по дуге*** y. Заметим, что если «реальный» поток по дуге отрицателен, то его можно сделать положительным, выбрав соответствующую ориентацию

дуги y. Систему уравнений (6.34) можно переписать в векторном виде:

$$B\Phi = l, \tag{6.36}$$

где B — матрица инциденций размерности $n \times m$, $\Phi = (\varphi(y_1) \ldots \varphi(y_m))^T$, $l = (0..0\underset{s}{\upsilon}0..0\underset{t}{-\upsilon}0..0)^T$. Так как ранг матрицы инциденций равен $n - 1$, то система уравнений (6.34) избыточна: $\sum_{i=1}^{n} Q(x_i) = 0$. Заметим также, что поток φ из x_s в x_t величины υ есть поток величины $-\upsilon$ из x_t в x_s.

Пример

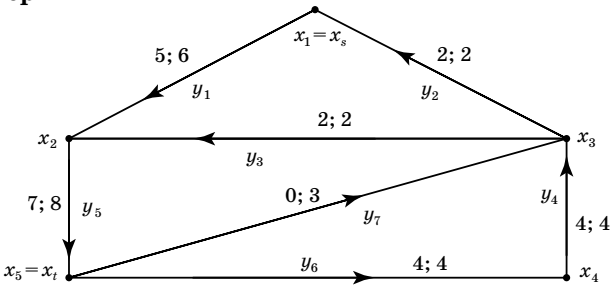


Рис. 6.52. Поток величины 3

На рис. 6.52 приведен пример сети, состоящей из пяти узлов и восьми дуг, в которой рассматривается поток из x_1 в x_5. Каждой дуге приписаны два числа: первое — величина потока по дуге, вторая — пропускная способность дуги. Величина этого потока равна 3. Действительно,

$$\begin{aligned} &Q(x_1) = 5 - 2 = 3, && Q(x_2) = 7 - (5 + 2) = 0, \\ &Q(x_3) = -4 - 0 + 2 + 2 = 0, && Q(x_4) = -4 + 4 = 0, \\ &Q(x_5) = 4 + 0 - 7 = -3. \end{aligned} \tag{6.37}$$

Систему уравнений (6.37) можно записать в векторном виде $B\Phi = l$ (6.36):

$$B = \begin{pmatrix} 1 & -1 & 0 & 0 & 0 & 0 & 0 \\ -1 & 0 & -1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & -1 & 0 & 0 & -1 \\ 0 & 0 & 0 & 1 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 & -1 & 1 & 1 \end{pmatrix}, \quad \Phi = \begin{pmatrix} 5 \\ 2 \\ 2 \\ 4 \\ 7 \\ 4 \\ 0 \end{pmatrix}, \quad l = \begin{pmatrix} 3 \\ 0 \\ 0 \\ 0 \\ -3 \end{pmatrix}.$$

Задача о максимальном потоке

Задача состоит в нахождении такого множества потоков по дугам, чтобы величина $Q(x_s)$ была максимальной. Введем необходимые определения.

Сечение S отделяет x_s от x_t, если вершины x_s, x_t принадлежат разным сторонам сечения: $x_s \in X_s$, $x_t \in X_t$, $X = X_s \cup X_t$. ***Пропускной способностью $c(S)$ сечения S*** называется сумма пропускных способностей дуг сечения, начинающихся из X_s и заканчивающихся в X_t:

$$c(S) = \sum_{y_j \in (X_s \to X_t)} c_j \ .$$

Алгоритм расстановки пометок основан на следующей теореме.

Теорема 6.20. *Теорема о максимальном потоке и минимальном разрезе*

Для любой сети максимальная величина потока из x_s в x_t равна минимальной пропускной способности сечения, отделяющего x_s от x_t.

□ Покажем сначала, что величина v любого потока φ не превосходит пропускную способность любого сечения (X_s, X_t), отделяющего x_s от x_t. Так как функция φ является потоком, то она удовлетворяет уравнениям (6.34) сохранения потока:

$$\begin{aligned}
&\sum_{y \in x_s \to X} \varphi(y) - \sum_{y \in X \to x_s} \varphi(y) = v, \\
&\sum_{y \in x \to X} \varphi(y) - \sum_{y \in X \to x} \varphi(y) = 0, \quad x \neq x_s, x_t, \\
&\sum_{y \in x_t \to X} \varphi(y) - \sum_{y \in X \to x_t} \varphi(y) = -v.
\end{aligned} \tag{6.38}$$

Сложим те уравнения из (6.38), которые соответствуют вершинам из X_s. Учитывая, что $x_s \in X_s$, $x_t \in X_t$, получаем:

$$v = \sum_{y \in X_s \to X} \varphi(y) - \sum_{y \in X \to X_s} \varphi(y).$$

Все множество вершин распадается на две стороны: $X = X_s \cup X_t$. Получаем

$$v = \sum_{y \in X_s \to X_s \cup X_t} \varphi(y) - \sum_{y \in X_s \cup X_t \to X_s} \varphi(y) =$$

$$= \sum_{y \in X_s \to X_s} \varphi(y) + \sum_{y \in X_s \to X_t} \varphi(y) - \sum_{y \in X_s \to X_s} \varphi(y) - \sum_{y \in X_t \to X_s} \varphi(y) =$$

$$= \sum_{y \in X_s \to X_t} \varphi(y) - \sum_{y \in X_t \to X_s} \varphi(y) \le \sum_{y \in X_s \to X_t} \varphi(y) \le \sum_{y \in X_s \to X_t} c(y) = c(X_s, X_t).$$

Теперь нам осталось показать, что существуют некоторые поток φ и сечение (X_s, X_t), для которых величина потока равна пропускной способности сечения. Как видно, все потоки от X_s к X_t ограничены и среди них можно выбрать максимальный поток φ. С его помощью определим сечение (X_s, X_t), для которого

$$\sum_{y \in X_s \to X_t} \varphi(y) = c(X_s, X_t), \quad \sum_{y \in X_t \to X_s} \varphi(y) = 0.$$

Определим множество X_s — одну из сторон сечения — рекуррентно:

1) $x_s \in X_s$.

2) Если $x_i \in X_s$, дуга y ведет из вершины x_i в некоторую вершину x_j и $\varphi(y) < c(y)$, то $x_j \in X_s$.

3) Если $x_i \in X_s$, дуга y ведет из некоторой вершины x_k в вершину x_i и $\varphi(y) > 0$, то $x_k \in X_s$.

Шаг 1) построения множества X_s означает, что источник x_s принадлежит построенной стороне сечения. Покажем, что сток тогда лежит по другую сторону сечения — $x_t \in X_t = X \backslash X_s$. Предположим противное: пусть $x_t \in X_s$. Тогда существует «неориентированная» цепь, ведущая от источника x_s к стоку x_t, такая, что для каждой дуги y_i цепи с направлением, совпадающим с ориентацией «от источника к стоку» $\varphi(y_i) < c$, а для каждой дуги y_j цепи с направлением, несовпадающим с ориентацией «от источника к стоку» $\varphi(y_j) > 0$. (Такая цепь называется ***аугментальной цепью потока***).

Обозначим через $l_1 = \min\{c(y_i) - \varphi(y_i)\}$, по всем дугам y_i цепи с направлением совпадающим с ориентацией «от источника к стоку», $l_2 = \min\{\varphi(y_j)\}$, по всем дугам y_j цепи с направлением, не совпадающим с ориентацией «от источника к стоку», $l = \min(l_1, l_2)$. Поток φ можно увеличить на l, увеличив на l поток на дугах цепи, ведущих «от источника к стоку» и уменьшив на l поток на дугах цепи, ведущих «от стока к источнику». Пришли к противоречию с тем, что величина потока φ является максимальной величиной допустимого потока от вершины x_s к вершине x_t. ■

В рассмотренном выше примере поток не является максимальным. Минимальную пропускную способность имеет сечение, состоящее из дуг y_1, y_2: $c(X_s, X_t) = 6$. При этом

$$X_s = \{x_s\}, \quad X_t = \{x_2, x_3, x_4, x_t\}, \quad X_s \to X_t = \{y_1\}, \quad X_t \to X_s = \{y_2\}.$$

Алгоритм расстановки пометок для задачи о максимальном потоке

Доказательство теоремы 6.20 дает алгоритм для построения минимального сечения (X_s, X_t), отделяющего x_s от x_t, и максимального потока φ от x_s к x_t. Этот алгоритм предложен Л. Фордом и Д. Фалкерсоном. Алгоритм начинает работу с известного допустимого потока φ (например, с нулевого). Затем вычисления развиваются в виде последовательности «расстановок пометок» (операция *А*), каждая из которых либо приводит к потоку с большей величиной (операция *Б*), либо же заканчивается заключением, что рассматриваемый поток максимален.

Будем предполагать, что пропускные способности c_j дуг сети целочисленные.

Операция А (расстановка пометок). Каждая вершина может находиться в одном из трех состояний: вершине приписана пометка и вершина просмотрена (т.е. она имеет пометку и все смежные с ней вершины «обработаны»), вершина помечена, но не просмотрена, вершина не помечена. Пометка вершины x_i имеет один из двух видов: $(+x_j, l)$ или $(-x_j, l)$. Часть $+x_j$ пометки первого типа означает, что поток допускает увеличение вдоль дуги (x_j, x_i) на величину l. Часть $-x_j$ пометки второго типа означает, что поток допускает уменьшение вдоль дуги (x_i, x_j) на величину l. Сначала все вершины не имеют пометок.

Шаг 1. Источнику x_s присваивается пометка $(+, \infty)$. Вершина x_s помечена, но «не просмотрена».

Шаг 2. Возьмем любую помеченную, но не просмотренную вершину. Пусть она имеет пометку $(\pm x_k, l(x_i))$. «Просмотрим» ее, то есть просмотрим все вершины, смежные с ней, и пометим те из них, которые еще непомечены.

Всем непомеченным вершинам x_j, в которые идут дуги y_r из x_i и для которых $\varphi(y_r) < c_r$, приписываем пометку $(+x_i, l(x_j))$, где

$$l(x_j) = \min(l(x_i), c_r - \varphi(y_r)).$$

Всем непомеченным вершинам x_j, из которых идут дуги y_r в x_i и для которых $\varphi(y_r) > 0$, приписываем пометку $(-x_i, l(x_j))$, где

$$l(x_j) = \min(l(x_i), \varphi(y_r)).$$

Теперь вершина x_i и помечена, и просмотрена, а вершина x_j, помечена, но не просмотрена.

Шаг 3. Повторять шаг 2 до тех пор, пока либо сток — вершина x_t — будет помечена, либо вершина x_t будет не помечена и никаких других пометок нельзя будет расставить. В первом случае переходим к операции B, а во втором случае алгоритм заканчивает работу с максимальным потоком φ. Во втором случае множество помеченных вершин и множество непомеченных вершин образуют две стороны минимального сечения (X_s, X_t).

Операция Б (увеличение потока)

Шаг 4. Положить $x = x_t$ и перейти к шагу 5.

Шаг 5. Если пометка в вершине x имеет вид $(+z, l(x))$, то изменить поток вдоль дуги (z, x) с $\varphi(z, x)$ на $\varphi(z, x) + l(x)$.

Если пометка в вершине X имеет вид $(-z, l(x))$, то изменить поток вдоль дуги (x, z) с $\varphi(x, z)$ на $\varphi(x, z) - l(x)$.

Шаг 6. Если $z = x_s$, то стереть все пометки и вернуться к шагу 1, чтобы вновь начать расставлять пометки, но используя уже улучшенный поток, найденный на шаге 5.

Если $z \neq x_s$, то взять $x = z$ и вернуться к шагу 5.

Алгоритм расстановки пометок (операция A) представляет собой поиск пути от источника x_s к стоку x_t, вдоль которого можно увеличить поток, не превышая пропускных способностей на дугах и сохраняя поток в промежуточных вершинах. Если сток оказался помеченным, то такой путь (соответственно — аугментальная цепь) найден и вдоль него легко увеличить поток. Если же операция A закончилась, а сток оказался непомеченным, то поток был максимален — его нельзя увеличить. Множество дуг, ведущих из помеченных вершин в непомеченные, образует минимальное сечение, так как помеченные вершины как раз составляют множество X_s, определенное в доказательстве теоремы 6.20.

Главная причина вычислительной эффективности алгоритма расстановки пометок состоит в том, что если какая-либо вершина помечена и просмотрена, то в дальнейшем ее можно не учитывать. Приписывание вершине x некоторой пометки соответствует выделению пути из x_s в x, который может быть началом аугментальной

цепи. Таких путей из x_s в x может быть много, но достаточно найти хотя бы один из них.

Число, на которое увеличивается поток в операции *Б*, является натуральным. Следовательно, если величина допустимого потока, с которого началась работа алгоритма, целая, то все последующие потоки будут целочисленными. Отсюда следует, что алгоритм *конечен*. Сформулируем теорему.

Теорема 6.21. *Теорема о целочисленности*

Если пропускные способности дуг c_j целые, то существует максимальный поток, имеющий также целую величину.

Реализация алгоритма Форда — Фалкерсона для сети из вышеприведенного примера

1. Возьмем поток, изображенный на рисунке 6.52 в качестве начального допустимого потока. Он имеет величину 3.

2. Присвоим источнику, вершине x_1, пометку $(+, \infty)$. Вершина x_1 помечена, но не просмотрена.

3. Просмотрим вершины, смежные с вершиной x_1. Вершине x_2 присвоим пометку $(+x_1, 1)$, а вершине x_3 — пометку $(-x_1, 2)$ ($\varphi(x_1, x_2) = 5 < 6 = c_1$, $\varphi(x_3, x_1) = 2 > 0$). Вершина x_1 помечена и просмотрена, а вершины x_2 и x_3 помечены, но не просмотрены.

4. Просмотрим вершины, смежные с вершиной x_3. Из вершин, смежных с вершиной x_3, не помечены вершины x_4 и x_5. Вершине x_4 присвоим пометку $(-x_3, 2)$, так как $\varphi(x_4, x_3) = 4 > 0$ и $\min(2, 4) = 2$. Вершину x_5 не помечаем, так как $\varphi(x_5, x_3) = 0$.

5. Просмотрим вершины, смежные с вершиной x_2. Вершине x_5 присвоим пометку $(+x_2, 1)$, так как $\varphi(x_2, x_5) = 7 < 8 = c_5$. Сток помечен. Переходим к операции *Б* — увеличению потока.

6. Сток имеет пометку $(+x_2, 1)$. Поэтому увеличиваем поток вдоль дуги (x_2, x_5) на 1.

7. Вершина x_2 имеет пометку $(+x_1, 1)$. Поэтому увеличиваем поток вдоль дуги (x_1, x_2) на 1. Мы получили новый поток величины 4 (рис. 6.53).

8. Стираем все пометки.

9. Присвоим вершине x_1 пометку $(+, \infty)$.

10. Просмотрим вершины, смежные с вершиной x_1. Вершине x_3 присвоим пометку $(-x_1, 2)$. Вершину x_2 не помечаем, так как $\varphi(x_1, x_2) = 6 = c(y_1)$.

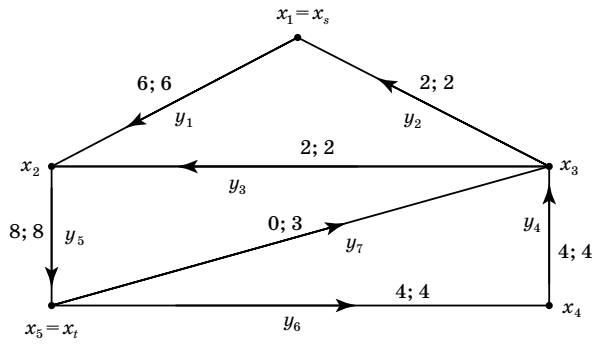


Рис. 6.53. Поток величины 4

11. Просмотрим вершины, смежные с вершиной x_3. Вершине x_4 присвоим пометку $(-x_3, 2)$, так как $\varphi(x_4, x_3) = 4 > 0$, $l(x_3) = 2$ и $\min(2, 4) = 2$.

12. Просмотрим вершины, смежные с вершиной x_4. Вершине x_5 присвоим пометку $(-x_4, 2)$, так как $\varphi(x_5, x_4) = 4 > 0$, $l(x_4) = 2$ и $\min(2, 4) = 2$. Сток помечен. Переходим к операции *Б* — увеличению потока.

13. Сток имеет пометку $(-x_4, 2)$. Поэтому уменьшаем поток вдоль дуги (x_5, x_4) на 2.

14. Вершина x_4 имеет пометку $(-x_3, 2)$. Поэтому уменьшаем поток вдоль дуги (x_4, x_3) на 2.

15. Вершина x_3 имеет пометку $(-x_1, 2)$. Поэтому уменьшаем поток вдоль дуги (x_3, x_1) на 2. Мы получили новый поток величины 6 (рис. 6.54). Из теоремы 6.20 мы знаем, что этот поток является максимальным. Проверим это.

16.Стираем все пометки.

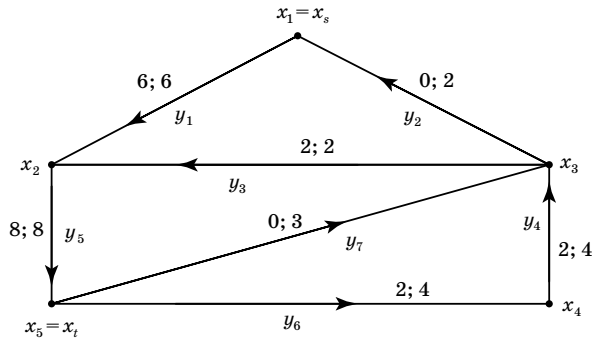


Рис. 6.54. Максимальный поток

17. Присвоим вершине x_1 пометку $(+, \infty)$.

18. Вершины, смежные вершине x_1, нельзя пометить, так как дуга (x_1, x_2) насыщена — $\varphi(x_1, x_2) = \varphi(y_1) = c(y_1) = 6$, а через дугу (x_3, x_1) поток не передается. Сток остался непомеченным. Следовательно, полученный поток является максимальным. Дуги (x_1, x_2) и (x_3, x_1) образуют минимальное сечение. Множество помеченных вершин образует ту его сторону, которая содержит источник: $X_s = \{x_1\}$. Непомеченные вершины образуют другую сторону сечения, содержащую сток: $X_t = \{x_2, x_3, x_4, x_5\}$. Построенный поток имеет вид $\varphi(y_1) = 6$, $\varphi(y_5) = 8$, $\varphi(y_6) = 2$, $\varphi(y_4) = 2$, $\varphi(y_3) = 2$, $\varphi(y_2) = \varphi(y_7) = 0$.

Графы со многими источниками и стоками

Алгоритм Форда — Фалкерсона применим и для определения величины максимального потока между множеством вершин-источников и множеством вершин-стоков. Разобьем множество вершин на множество источников $X_+ = \{x \in X: Q(x) > 0\}$, создающих поток, множество стоков $X_- = \{x \in X: Q(x) < 0\}$, потребляющих поток и множество промежуточных вершин $X_0 = \{x \in X: Q(x) = 0\}$, сохраняющих поток φ: $X = X_+ \cup X_- \cup X_0$. Преобразуем поток φ в поток, который имеет только один источник и один сток, увеличив количество вершин в сети. Для этого добавим две новые вершины — «фиктивный» источник x_s и «фиктивный» сток x_t. Соединим вершину x_s со всеми настоящими источниками. Этим дугам припишем поток, создаваемый соответствующим источником. А от каждого настоящего стока направим дугу к «фиктивному» стоку x_t. Этим дугам припишем поток, потребляемый соответствующим стоком. При этом пропускные способности добавленных дуг считаем бесконечными. В результате получаем сеть с одним источником и одним стоком. Применяя к ней алгоритм Форда — Фалкерсона, находим максимальный поток, который является максимальным и для исходной сети.

Проиллюстрируем на примере преобразование сети с несколькими источниками и несколькими стоками к сети с одним источником и одним стоком.

Пример

На рис. 6.55 изображена сеть с двумя источниками x_1 и x_3 и тремя стоками x_7, x_8, x_9:

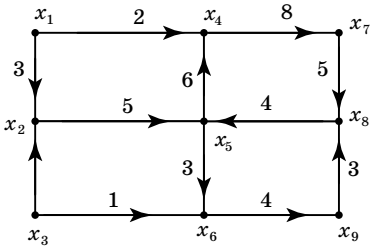


Рис. 6.55. Сеть с тремя стоками

$$Q(x_1) = 5, \quad Q(x_3) = 3, \quad Q(x_7) = 3, \quad Q(x_8) = 4, \quad Q(x_9) = 1,$$
$$Q(x_2) = Q(x_4) = Q(x_5) = Q(x_6) = 0.$$

Преобразуем эту сеть к сети с одним источником и одним стоком.

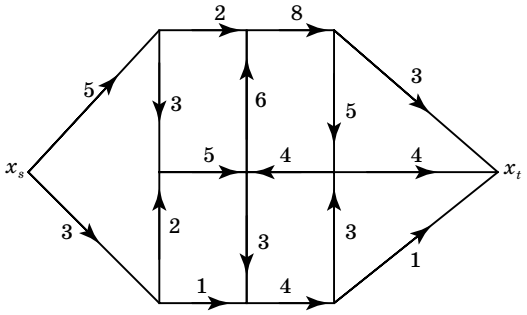


Рис. 6.56. Преобразованная сеть

На рис. 6.56 изображена сеть уже с одним источником x_s и одним стоком x_t.

Задача о многополюсном максимальном потоке

Рассмотрим задачу нахождения максимального потока для всех пар узлов в неориентированной сети. Эту задачу можно рассматривать как обобщение задачи с одним источником и одним стоком и ее можно решить, применяя алгоритм Форда — Фалкерсона для каждой пары вершин. Более эффективным является алгоритм, предложенный Р. Гомори и Т. Ху.

Алгоритм Гомори — Ху

1. Выберем некоторые две вершины графа. Обозначим одну из них через x_s, а другую через x_t.

2. Применим алгоритм Форда — Фалкерсона и найдем максимальный поток из источника x_s в сток x_t. При этом множество помеченных вершин и множество непомеченных вершин образуют две стороны минимального сечения X_s и X_t.

3. Выберем две вершины графа x_i и x_j, лежащие по одну сторону от сечения. Например, пусть $x_i, x_j \in X_s$.

4. Заменим дуги из минимального сечения (X_s, X_t) одной дугой, а вершины стороны сечения, в которой не лежат вершины x_i, x_j, (например, X_t), — одной вершиной. Пропускную способность

в так определенной дуге положим равной пропускной способности сечения (X_s, X_t).

5. Положим: $x_i = x_s$, $x_j = x_t$ и вернемся ко второму шагу.

Можно показать, что после того, как будет выбрана $n - 1$ пара вершин, мы определим все $\frac{n(n-1)}{2}$ величин максимального потока для исходной сети. Основная идея алгоритма состоит в итеративном построении максимального остовного дерева, ветви которого соответствуют сечениям, а пропускные способности ветвей равны пропускным способностям этих сечений. Если бы мы применяли алгоритм Форда — Фалкерсона к каждой паре вершин, то нам бы пришлось его применить $\frac{n(n-1)}{2}$ раз. В алгоритме же Гомори — Ху максимальный поток между парой вершин определяется с помощью алгоритма расстановки пометок только $n - 1$ раз.

Проиллюстрируем на примере алгоритм Гомори — Ху.

Пример

Рассмотрим сеть, изображенную на рис. 6.57. Числа, приписанные дугам, соответствуют их пропускным способностям. Требуется для каждой пары узлов сети определить величину максимального потока между ними. Данная задача решается за $n - 1 = 6 - 1 = 5$ итераций алгоритма Гомори — Ху. Если алгоритм Форда — Фалкерсона расстановки пометок применялся бы к каждой паре узлов, то потребовалось бы решить 15 задач о максимальном потоке.

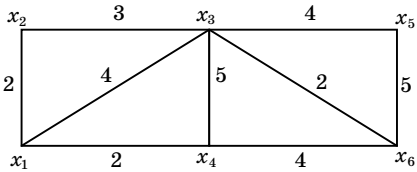


Рис. 6.57. Сеть с пропускными способностями

Реализация алгоритма Гомори — Ху для сети из примера к рис. 6.57

1. Выберем вершины $s = x_1$ и $t = x_2$. Минимальной пропускной способностью относительно источника $s = x_1$ и стока $t = x_2$ обладает сечение со сторонами $X_s = \{x_1, x_3, x_4, x_5, x_6\}$ и $X_t = \{x_2\}$. По теореме 6.20 величина максимального потока между вершинами x_1 и x_2 равна пропускной способности сечения (X_s, X_t): $v_{12} = v_{21} = c(X_s, X_t) = 2 + 3 = 5$. Объединим вершины из X_s в одну вершину и соединим ее с вершиной x_2 ветвью с пропускной способностью 5 (рис. 6.58).

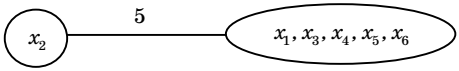


Рис. 6.58. Первый шаг алгоритма

2. Выберем вершины $s = x_1$ и $t = x_3$. Минимальной пропускной способностью относительно источника $s = x_1$ и стока $t = x_3$ обладает сечение со сторонами $X_s = \{x_1\}$ и $X_t = \{x_2, x_3, x_4, x_5, x_6\}$. По теореме 6.20 величина максимального потока между вершинами x_1 и x_3 равна пропускной способности сечения (X_s, X_t): $\upsilon_{13} = \upsilon_{31} = c(X_s, X_t) = 2 + 4 + 2 = 8$. Объединим вершины x_3, x_4, x_5, x_6 в одну вершину и соединим ее с вершиной x_1 ветвью с пропускной способностью 8 (рис. 6.59).

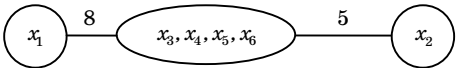


Рис. 6.59. Второй шаг

3. Выберем вершины $s = x_4$ и $t = x_3$. Минимальной пропускной способностью относительно источника $s = x_4$ и стока $t = x_3$ обладает сечение со сторонами $X_s = \{x_4\}$ и $X_t = \{x_1, x_2, x_3, x_5, x_6\}$. По теореме 6.20 величина максимального потока между вершинами x_4 и x_3 равна пропускной способности сечения (X_s, X_t): $\upsilon_{43} = \upsilon_{34} = c(X_s, X_t) = 2 + 5 + 4 = 11$. Объединим вершины x_3, x_5, x_6 в одну вершину и соединим ее с вершиной x_4 ветвью с пропускной способностью 11 (рис. 6.60).

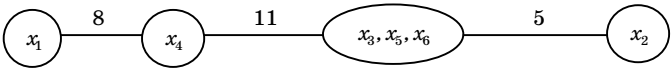


Рис. 6.60. Третий шаг

4. Выберем вершины $s = x_5$ и $t = x_3$. Минимальной пропускной способностью относительно источника $s = x_5$ и стока $t = x_3$ обладает сечение со сторонами $X_s = \{x_5\}$ и $X_t = \{x_1, x_2, x_3, x_4, x_6\}$. Величина максимального потока между вершинами x_5 и x_3 равна пропускной способности сечения (X_s, X_t): $\upsilon_{53} = \upsilon_{35} = c(X_s, X_t) = 5 + 4 = 9$. Объединим вершины x_3, x_6 в одну вершину и соединим ее с вершиной x_5 ветвью с пропускной способностью 9 (рис. 6.61).

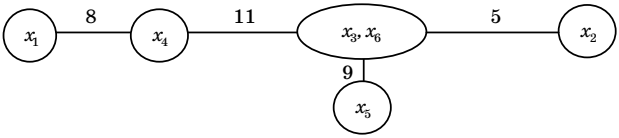


Рис. 6.61. Четвертый шаг

5. Выберем вершины $s = x_6$ и $t = x_3$. Минимальной пропускной способностью относительно источника $s = x_6$ и стока $t = x_3$ обладает сечение со сторонами $X_s = \{x_6\}$ и $X_t = \{x_1, x_2, x_3, x_4, x_5\}$. Величина максимального потока между вершинами x_6 и x_3 равна пропускной способности сечения (X_s, X_t): $v_{63} = v_{36} = c(X_s, X_t) = 5 + 6 + 4 = 15$. Соединим вершину x_3 с вершиной x_6 ветвью с пропускной способностью 15 (рис. 6.62).

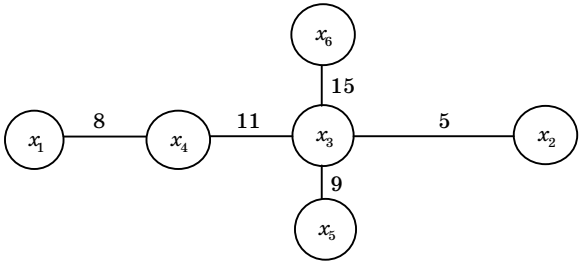


Рис. 6.62. Окончательное дерево сечений

По дереву сечений, изображенному на рис. 6.62, легко найти остальные величины максимальных потоков. Например, $v_{16} = v_{61} = 8$, так как в цепи дерева сечений, соединяющей вершины x_1 и x_6, минимальная пропускная способность ветвей равна 8. Запишем величины максимальных потоков в виде матрицы:

$$V = \begin{pmatrix} - & 5 & 8 & 8 & 8 & 8 \\ 5 & - & 5 & 5 & 5 & 5 \\ 8 & 5 & - & 11 & 9 & 15 \\ 8 & 5 & 11 & - & 9 & 11 \\ 8 & 5 & 9 & 9 & - & 9 \\ 8 & 5 & 15 & 11 & 9 & - \end{pmatrix}.$$

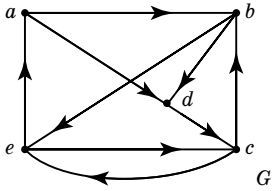
Здесь на пересечении i строки и j столбца стоит величина максимального потока между вершинами x_i и x_j.

Задания

1. Может ли в цепи быть два одинаковых ребра? Две одинаковых вершины?
2. Может ли дерево содержать цикл?
3. Любой ли цикл в ориентированном графе является контуром?

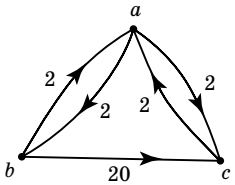
4. Могут ли отрицательные числа быть элементами матрицы смежности A некоторого графа?
5. Чему равна сумма элементов по столбцу матрицы инциденций некоторого графа?
6. Равна ли сумма по строке сумме по столбцу у матрицы смежности A неориентированного графа? Ориентированного?
7. Что означает равенство нулю матрицы смежности A в некоторой степени k?
8. Является ли связным граф, у которого число ребер $m < n - 1$, где n — число вершин?
9. Сколько ребер содержит простой полный граф с n вершинами?
10. Верно ли, что среди степеней вершин графа есть хотя бы две одинаковые?
11. Если граф связен и n — число его вершин, то какова нижняя оценка для числа его ребер?
12. Являются ли плоскими двудольные графы $K_{2,2}$ и $K_{3,3}$?
13. Построить матрицу смежности двудольных графов $K_{2,2}$ и $K_{3,3}$.
14. Какое количество компонент связности может быть у графа с n вершинами?
15. Какую геометрическую информацию несут элементы матрицы A^2, где A — матрица смежности ориентированного графа?
16. Ограничено ли множество хроматических чисел класса всех плоских графов?
17. Найти хроматические числа графов Понтрягина — Куратовского $K_{3,3}$ и K_5.
18. Любое ли натуральное число n может хроматическим числом некоторого графа?
19. Чему равно хроматическое число дерева?
20. Может ли число ребер дерева равняться числу его вершин?
21. Сколько существует различных графов с 4-мя вершинами? С 5-тью?
22. Как вычислить число линейно независимых циклов графа?
23. Построить матрицу инциденций графа, представляющего собой реберный каркас куба.
24. В каком случае матрица смежности ориентированного графа является симметрической?
25. Построить матрицы смежности и инциденции графа, полученного соединением центра правильного пятиугольника с его вершинами.
26. Построить фундаментальную матрицу циклов двудольного графа $K_{3,3}$.
27. Построить фундаментальную матрицу сечений двудольного графа $K_{3,3}$.
28. Построить фундаментальные матрицы циклов и сечений графа $K_{2,4}$.

29. Существует ли гамильтонов цикл в двудольных графах $K_{2,2}$, $K_{3,3}$, $K_{2,3}$, $K_{2,4}$?
30. Найти гамильтоновы контуры в графе.



G

31. В графе H указать кратчайший контур коммивояжера и кратчайший замкнутый маршрут коммивояжера (с возможным повторением вершин).



H

32. Каково число ребер остовного дерева связного графа G с n вершинами и m ребрами?
33. Как связаны между собой хроматическое число γ и кликовое число ρ графа G?
34. Построить примеры клик и независимых множеств графов $K_{3,3}$, $K_{2,3}$ и графа из задачи 30.
35. Какое свойство является достаточным условием того, чтобы произвольный граф содержал гамильтонов цикл?
36. Верно ли, что любой остов графа G имеет, по крайней мере, 1 общее ребро с каждым разрезом?
37. Любое ли множество $\upsilon(G)$ независимых циклов будет «фундаментальным»? ($\upsilon(G)$ — это цикломатическое число графа G)
38. Любой цикл графа G можно представить в виде суммы по модулю 2 фундаментальных циклов. Верно ли обратное утверждение, т.е. что любая сумма по модулю 2 фундаментальных циклов обязательно даст единственный цикл?
39. Однозначно ли выбирается множество фундаментальных циклов?
40. Привести примеры сильно связного, слабо связного, несвязного графов.

41. Чему равна сумма всех элементов строки матрицы смежности A? Чему равна сумма элементов в столбце?
42. Справедливо ли утверждение, что любой полный симметрический граф содержит гамильтонов цикл?
43. Показать, что ранг матрицы инциденций B связного графа с n вершинами равен $n - 1$.
44. Чему равно число помеченных графов с n вершинами?
45. Как связаны число помеченных четных графов с n вершинами и число всех помеченных графов с числом вершин $n - 1$?
46. Чему равно число связных помеченных графов с n вершинами?
47. Всякий ли граф является эйлеровым?

раздел 7

Языки и грамматики

7.1. Задача формализации языков и перевода

Необходимость формального задания языков и решения задачи перевода в программировании

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. Одним из основных источников задач является проблема задания языка и оптимального перевода с одного языка на другой. Задача перевода 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 котором решаются данные задачи. Таким образом, речь идет о переводе с языка программирования на машинный язык с одновременной оптимизацией выходной программы. Эта задача решается при проектировании трансляторов и компиляторов. В то же время исходным может быть алгоритм работы некоторого

устройства ЭВМ, записанный на предназначенном для этой цели алгоритмическом языке, а язык, на который переводится 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 производится оптимизация и присутствует свой язык. Формальное задание языка необходимо как для самого процесса перевода, так и для определения класса допустимых записей на этом языке, т.е. для определения корректности или допустимости переводимой на другой язык записи. Языки состоят из строк символов и, следовательно, весь вычислительный процесс можно рассматривать как преобразование одного множества строк в другое, то есть как математический объект.

Вопросы

1. В каких случаях для работы ЭВМ необходимо решение задачи перевода с одного языка на другой?
2. Для чего решается задача формального задания языков?
3. Чем отличается процесс перевода в литературе от переводов, выполняемых при работе ЭВМ?

7.2. Преобразование строк символов

Алфавит, строка, символ, подалфавит, расширение алфавита, пустая строка, длина строки, конкатенация строк, подстрока

Формальный алфавит представляет собой множество неделимых символов. Конечные последовательности букв алфавита называются строками, выражениями в этом алфавите.

Алфавиты мы будем обозначать большими буквами латинского алфавита, например, A, B, C.

Символы мы будем обозначать маленькими буквами латинского алфавита с индексами или без индексов, например, a_1, a_2, a_n.

Поскольку алфавиты являются множествами, то к ним применимы теоретико-множественные операции: $\cap$, $\cup$, $\subset$, $\subseteq$, $\in$, $\notin$ и т.д.

Определение

Если A и B — алфавиты и $A \subseteq B$, то A называется ***подалфавитом*** B и B — ***расширением*** A.

Строки являются упорядоченными совокупностями символов алфавита и, следовательно, являются элементами декартова произ-

ведения $A^n = \underbrace{A \times A \times \ldots \times A}_{n}$. Здесь A — алфавит, n — длина строк. Строки мы будем записывать в виде последовательности символов, например, $a_1 a_2 \ldots a_n$. Символы также являются строками для случая $n = 1$. В тексте значение строки приводится в кавычках, например, строки «*abcd*», «1 + 2 =», «00001110101». Строки обычно обозначаются маленькими буквами греческого алфавита, например, α, β, γ.

> **Определение**
> ***Пустая строка*** — это строка, которая не имеет символов. Обозначим ее ε. Пустая строка не является символом и не принадлежит никакому алфавиту. Это можно записать так: $\forall A\ \varepsilon \notin A$, где A — алфавит.

Через A^* обозначается множество всех строк алфавита A, включая пустую строку. Ясно, что $A^* = A^0 \cup A^1 \cup A^2 \cup \ldots$, где $A^0 = \varepsilon$ — пустая строка, $A^1 = A$ — всевозможные строки алфавита A длины 1, $A^2 = A \times A$ — всевозможные строки алфавита A длины 2, ... Множество A^* бесконечно.

Через A^+ обозначается множество всех строк алфавита A, не включая пустую строку: $A^+ = A^1 \cup A^2 \cup A^3 \ldots$; $A^* = A^0 \cup A^+$.

Длина строки α равна количеству символов в строке и обозначается $|\alpha|$. Если $\alpha = a_1 a_2 \ldots a_n$, то $|\alpha| = n$. Длина пустой строки равна нулю: $|\varepsilon| = 0$

> **Определение**
> ***Конкатенацией*** или слиянием строк α и β называется строка αβ. Например, конкатенацией строк «*sdf*», «*htuwq*» является строка «*sdfhtuwq*». Считается, что $\alpha\varepsilon = \varepsilon\alpha = \alpha$.

Запись α^n означает $\underbrace{\alpha\alpha\ldots\alpha}_{n\text{ раз}}$, α^1 означает α, α^0 означает ε. Здесь в качестве α может быть как строка символов, так и символ.

Например, $0^3 1^2 = 00011$. Здесь $a^n b^m c$ означает строку, состоящую из n символов a, за которыми следуют m символов b, за которыми следует один символ c.

> **Определение**
> Строка β является ***подстрокой*** строки вида αβγ, причем α и γ могут быть любыми, в том числе и пустыми строками.

Пустая строка ε является подстрокой любой строки.

Например, подстроками строки «$x_1 = y_1 + z_1$» являются строки «x_1», «x_1 =», «$1 = y_1 + z$», «+», пустая строка ε и другие строки. Число подстрок у конечной строки — конечно.

Вопросы

1. Назовите два различных подхода к изучению явлений с информационной точки зрения.
2. Является ли алфавитный способ достаточным для представления любой дискретной информации?
3. Приведите примеры алфавитного способа представления в повседневной жизни:
   а) непрерывной информации;
   б) дискретной информации.
4. Дайте определение алфавитного способа задания информации, алфавита, строки.
5. Даны два алфавита A^1 и A^2, причем $A^1 \subseteq A^2$. Как называется A^1 по отношению к A^2? Как называется A^2 по отношению к A^1?
6. Что такое пустая строка?
7. Дайте определение операции конкатенации строк.

Задания

1. Определите алфавиты для записи:
   а) арифметических выражений;
   б) азбуки Морзе;
   в) последовательности ходов при игре в шахматы.
   Приведите примеры строк в этих алфавитах.
2. Для алфавитов $A = \{a, b, c\}$, $B = \{0, 1\}$ составьте:
   а) A^*, B^*;
   б) A^+, B^+;
3. Пусть задана строка α = «последовательность». Найдите длину строки $|\alpha|$.
4. Постройте строки $0^5 1^2$, $a^3 b^2 c$, $0^n A 1^n$, для $n = 3, 5, 0$.
5. α = «alpha» β = «bet» Что является конкатенацией строк $\alpha\beta$?
6. Какие из перечисленных строк являются подстроками строки α = «индустриализация»:
   β_1 = «индус», β_2 = «индустрия», β_3 = «три», β_4 = «акция»?

7.3. Задание языков с помощью грамматик

Язык, распознающая и порождающая грамматики, терминальные и нетерминальные символы, продукция, начальный символ, вывод строк

Определение

Язык L — это множество строк конечной длины в заданном конечном алфавите.

Первый важный вопрос: как описать язык L в том случае, когда он бесконечен. Если длины всех строк ограничены, то L состоит из

конечного числа строк и можно составить список всех строк из L. Однако, для многих языков нельзя или нежелательно установить верхнюю границу допустимой длины строки языка. Такие языки нельзя определить перечислением всех строк. Требуемое описание языка должно быть конечным, в то время, как язык может быть бесконечным. Известно несколько таких способов описания языков. Один из них состоит в использовании системы, называемой ***грамматикой***. В зависимости от способа определения языка грамматики делят на распознающие и порождающие.

> **Определение**
> ***Распознающая грамматика*** — это грамматика, которая для любой строки может *определить*, является эта строка правильной или нет. ***Порождающая грамматика*** — это грамматика, которая может *построить* любую правильную строку и не строит неправильных строк.

Мы будем рассматривать только порождающие грамматики, называя их в дальнейшем просто грамматиками. Для каждой грамматики определяются два конечных непересекающихся множества: множество T терминальных символов и множество N нетерминальных символов.

Из ***терминальных*** символов образуются строки определяемого языка, это означает, что множество терминальных символов составляет алфавит этого языка. Например, если речь идет о грамматике, порождающей арифметические выражения, в этом случае терминальными символами являются цифры, скобки и знаки операций. ***Нетерминальные*** символы используются как вспомогательные или промежуточные в процедурах построения строк языка, но они не входят в окончательные («правильные») выражения языка. Для обозначения нетерминальных символов мы будем использовать большие буквы латинского алфавита.

Сердцевину грамматики составляет конечное множество продукций.

> **Определение**
> ***Продукция*** — это упорядоченная пара строк, первым элементом которой является любая строка, составленная из символов алфавита $N \cup T$, причем эта строка обязательно содержит хотя бы один нетерминальный символ. Вторым элементом пары является любая строка, составленная из символов алфавита $N \cup T$. Множество продукций P описывает процесс порождения строк языка.

Множество первых элементов продукций мы можем записать как $(N \cup T)^* N\ (N \cup T)^*$. Здесь $(N \cup T)^*$ — это множество строк, составленных из символов алфавита $N \cup T$, N — множество нетерминальных символов, из которых берется один обязательный для первой строки. Множество вторых элементов продукций мы можем записать как $(N \cup T)^*$. При записи первый и второй элемент продукции будем разделять символом «→». Таким образом, множество всех возможных продукций можно записать как $(N \cup T)^* N(N \cup T)^* \to (N \cup T)^*$.

Порождение строк в грамматиках начинается с особой строки, состоящей из одного выделенного символа, который называется ***начальным символом***. Он обозначается S.

Дадим теперь формальное определение грамматики.

Определение

Грамматикой называется структура $G = (N, T, P, S)$, где

N — конечное множество нетерминальных символов;

T — непересекающееся с N конечное множество терминальных символов;

P — множество продукций — конечное подмножество множества $(N \cup T)^* N(N \cup T)^* \times (N \cup T)^*$;

S — начальный символ из N.

Применение продукций грамматики приводит к порождению (выводу) строк в грамматике.

Приведем пример грамматики.

Пример. Рассмотрим грамматику $G1 = (\{A, X, S\}, \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, (,)\}, P, S)$, где P состоит из набора следующих продукций:

1. $S \to A$.	9. $X \to 3$.
2. $A \to A * A$.	10. $X \to 4$.
3. $A \to A + A$.	11. $X \to 5$.
4. $A \to A - A$.	12. $X \to 6$.
5. $A \to (A)$.	13. $X \to 7$.
6. $A \to X$.	14. $X \to 8$.
7. $X \to 1$.	15. $X \to 9$.
8. $X \to 2$.	16. $X \to 0$.

В этой грамматике множество нетерминальных символов — $N = \{A, X, S\}$, Множество терминальных символов — $T = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9,(,)\}$, грамматика содержит 16 продукций,

составляющих множество P, начальный символ грамматики S принадлежит N.

Грамматика определяет язык рекурсивно, с помощью выводимых строк.

Определение

Выводимые строки (строки, которые можно вывести) в грамматике определяются следующим образом:

Пусть $G = (N, T, P, S)$ — некоторая грамматика, тогда

1) S — выводимая строка;

2) Если $\alpha\beta\gamma$ — выводимая строка и продукция $\beta \to \delta$ содержится в P, то $\alpha\delta\gamma$ — тоже выводимая строка.

Язык L(G), определяемый грамматикой G, — это множество строк, которые состоят только из терминалов и выводятся в грамматике G. Грамматики ***эквивалентны***, если порождаемые ими языки равны. Равенство языков определяется по правилу равенства множеств.

Например, продукцией в некоторой грамматике может быть пара $AB \to CDE$. Применяется эта продукции таким образом:

1. Пусть AB является подстрокой строки, выводимой в грамматике G.
2. Заменим в этой строке AB на CDE — получим новую строку.
3. Эта новая строка тоже считается выводимой в грамматике G.

В частности, если строка $FGABH$ выводима, то $FGCDEH$ тоже выводима.

Возможность вывода строк в грамматиках обозначается так:

$\varphi \Rightarrow \psi$ (читается «ψ непосредственно выводима из φ») означает, что результатом применения одной продукции из P к строке φ является строка ψ.

$\varphi \Rightarrow^* \psi$ (читается «ψ выводима из φ») означает, что существует последовательность $\varphi \Rightarrow \alpha_1 \Rightarrow ... \Rightarrow \alpha_k \Rightarrow \psi$.

Пример. Вернемся к грамматике $G1 = (\{A, X, S\}, \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, (,)\}, P, S)$ из предыдущего примера.

Эта грамматика порождает выражения из цифр, содержащие знаки суммы и произведения.

Выведем строку $(1 + 5) \cdot 7$.

Номер продукции, использованной на каждом шаге, указан возле знака «$\Rightarrow$».

$$S \Rightarrow {}_1A \Rightarrow {}_2A \cdot A \Rightarrow {}_5(A) \cdot A \Rightarrow {}_3(A + A) \cdot A \Rightarrow {}_6(X + A) \cdot A \Rightarrow$$
$$\Rightarrow {}_6(X + X) \cdot A \Rightarrow {}_6(X + X) \cdot X \Rightarrow {}_7(1 + X) \cdot X \Rightarrow {}_{11}(1 + 5) \cdot X \Rightarrow$$
$$\Rightarrow {}_{13}(1 + 5) \cdot 7$$

Вопросы

1 Дайте определение языка.
2 Почему для того, чтобы описать язык, необходимо применять специальные формальные системы (например, грамматики)?
3 Какие грамматики являются распознающими и какие порождающими?
4 Дайте определение грамматики и ее составляющих.
5 Каким образом порождающие грамматики определяют язык? Какие строки называются выводимыми в грамматике?

Задания

1 Для грамматики $G1$, рассмотренной в примере, сделайте вывод строки $(4 - 5) + 2 \cdot 3$.
2. Измените запись предыдущего вывода, используя символ $\Rightarrow^*$.
3. Измените список продукций грамматики $G1$ так, чтобы в допускаемых ею выражениях могли присутствовать:
   а) однозначные и двузначные числа;
   б) числа с любым количеством разрядов;
   в) отрицательные числа.
4. Перепишите список продукций грамматики $G1$, не изменяя порождаемый ею язык,
   а) уменьшив число используемых нетерминальных символов;
   б) увеличив число используемых нетерминальных символов.

7.4. Форма Бекуса — Наура

Форма Бекуса — Наура записи продукций грамматики

Для удобства записи продукций грамматик используется нотация, введенная Бекусом — ***форма Бекуса — Наура*** (***форма Бекуса, БНФ***). Она особенно полезна, когда мы используем элементы из N, которые можно спутать с элементами из T. Эта нотация состоит в следующем:

1. Символ «$\rightarrow$» заменяется на «::=».

2. В угловые скобки «< >» заключаются нетерминальные символы.

3. Знак « | » используется для сокращенной записи продукций, обозначает «или» и объединяет группу продукций с одинаковой левой частью, но различными правыми частями.

Пример. Рассмотрим грамматику $G_{число}$:

$G_{\text{число}}$ = ({ Число, Знак, Целое Число, Дробная Часть, Цифра, S}, {+, –, 0, ..., 9,,}, P, S),

где P состоит из набора продукций:

1. Число → Знак ЦелоеЧисло ДробнаяЧасть.
2. Знак → +.
3. Знак → –.
4. Знак → ε.
5. ЦелоеЧисло → Цифра.
6. ЦелоеЧисло → Цифра ЦелоеЧисло.
7. ДробнаяЧасть →, ЦелоеЧисло.
8. ДробнаяЧасть → ε.
9. Цифра → 0.
10. Цифра → 1.
11. Цифра → 2.
12. Цифра → 3.
13. Цифра → 4.
14. Цифра → 5.
15. Цифра → 6.
16. Цифра → 7.
17. Цифра → 8.
18. Цифра → 9.

Запишем продукции этой грамматики соответственно БНФ:

<Число> ::= <Знак> <Целое Число> <Дробная Часть>

<Знак> ::= + | – | ε

<Целое Число> ::= <Цифра> | <Цифра> <Целое Число>

<ДробнаяЧасть> ::=, <Целое Число> | ε

<Цифра>:: = 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9

На практике при записи грамматик могут использоваться не все символы БНФ, а только « | » или «<>», « | ».

Вопросы

1. Для чего применяется форма Бекуса — Наура?
2. В чем состоит форма Бекуса — Наура?
3. Назовите преимущества записи продукций грамматик с помощью БНФ.

Задания

1. Запишите продукции грамматики $G1$ (п. 7.3) в БНФ.
2. Каким продукциям грамматики $G = (\{A, B, S\},\{a, b\},P, S)$ соответствует запись $S \to aA \mid bA \mid ABba$?
3. Запишите в БНФ продукции грамматики $G = (\{A, B, S\},\{0, 1\},P, S)$:
   1) $S \to AB$. 5) $A \to 010$. 9) $S \to BA$.
   2) $AB \to 000$. 6) $B \to 101$. 10) $BA \to 111$.
   3) $A \to 001$. 7) $A \to 100$. 11) $S \to \varepsilon$.
   4) $B \to 110$. 8) $B \to 011$.
4. Какой язык определяется грамматикой $G_{число}$? Произведите вывод строк «–12,178», «100», «0,5», «0» в грамматике $G_{число}$.

7.5. Типы грамматик

Иерархия Хомского: грамматики общего вида, контекстно-зависимые, контекстно-свободные, регулярные. Проблемы принадлежности, пустоты, эквивалентности для языков

Грамматики можно классифицировать по виду их продукций. Такая классификация называется ***иерархией Хомского***.

Пусть $G = (N, T, P, S)$ — грамматика.

Тип 0. Любая грамматика определенного ранее вида называется грамматикой ***общего вида*** (с фразовой структурой, без ограничений).

Тип 1. Грамматика называется ***контекстно-зависимой*** (***КЗ***) или неукорачивающей, если каждая продукция из P имеет вид $\alpha \to \beta$, где $|\alpha| \le |\beta|$ или $\alpha \to \varepsilon$, где ε — пустая строка (ε-продукции).

Тип 2. Грамматика называется ***контекстно-свободной*** (***КС***), если каждая продукция из P имеет вид $A \to \alpha$, где $A \in N$, $\alpha \in (N \cup T)^*$.

Тип 3. Грамматика называется ***праволинейной*** (выровненной вправо), если каждая продукция из P имеет вид $A \to xB$ или $A \to x$, где $A, B \in N$, $x \in T^*$. Грамматика называется ***леволинейной*** (выровненной влево), если каждая продукция из P имеет вид $A \to Bx$ или $A \to x$, где $A, B \in N$, $x \in T^*$. Леволинейные и праволинейные грамматики называются ***регулярными***.

Эта иерархия является монотонно включающей, т.е. все грамматики типа 3 являются грамматиками типа 2, все грамматики типа 2 — грамматиками типа 1 и т.д., как показано на рис. 7.1. Эта иерархия включения справедлива и для порождаемых языков. При этом языку присваивается тип и качественная характеристика порождающей грамматики — общего вида, КЗ, КС, регулярный язык соответственно.

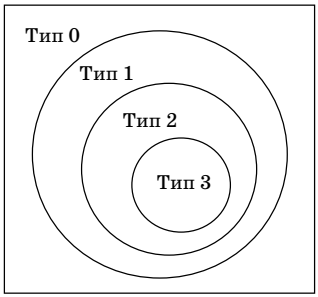


Рис. 7.1. Типы грамматик

Например, поскольку каждая праволинейная грамматика является контекстно-свободной, то каждый праволинейный язык является КС-языком, и существуют КС-языки, не являющиеся регулярными.

Включения типов являются строгими в том смысле, что существуют языки, которые относятся к типу k и не относятся к типу $k + 1$ ($0 \le k \le 2$).

Но если язык можно генерировать грамматикой типа k, то возможно, что его можно генерировать грамматикой типа $k+1$. Например, язык, генерируемый посредством нерегулярной грамматики может оказаться регулярным.

Уже первое ограничение, накладываемое на продукции грамматик типа 1, определяет важное свойство распознаваемости языка. Для неукорачивающих грамматик без ε-продукций доказано, что порождаемые ими языки легко распознаваемы, т.е. за конечное число шагов для любой строки можно определить, принадлежит данная строка языку или нет.

Второе ограничение на продукции в грамматике типа 2 освобождает от влияния контекста. Например, продукция вида $\alpha_1\beta\alpha_2 \to \alpha_1\gamma\alpha_2$ означает разрешение заменять β на γ только в контексте α_1, α_2. Такие продукции называются контекстно-связанными. В противоположность этому продукции, не использующие контекста (правила КС-грамматик), называются ***контекстно-свободными***. Грамматики типа 1 называются контекстно-зависимыми, поскольку могут содержать контекстно-связанные продукции.

Сравнивая продукции грамматики G_1 из примера п. 7.3 и ограничения в иерархии Хомского, приходим к выводу, что эта грамматика является контекстно-свободной. Приведем примеры других грамматик и определим, к какому типу относится каждая из них.

Пример 1. Эта грамматика порождает последовательность (набор) из цифр 0, 1, 2, где каждая последующая цифра равна сумме двух предыдущих по модулю 3.

$G_{сумма} = (\{A, C, S\}, \{0, 1, 2\}, P, S)$,
где P состоит из набора продукций:

1. $S \to AAC$.
2. $A \to 0$.
3. $A \to 1$.
4. $A \to 2$.
5. $C \to \varepsilon$.
6. $00C \to 000C$.
7. $01C \to 011C$.
8. $02C \to 022C$.
9. $10C \to 101C$.
10. $11C \to 112C$.
11. $12C \to 120C$.
12. $20C \to 202C$.
13. $21C \to 210C$.
14. $22C \to 221C$.

Пример вывода: $S \Rightarrow_1 AAC \Rightarrow_2 0AC \Rightarrow_4 02C \Rightarrow_8 022C \Rightarrow_{14} 0221C \Rightarrow$
$\Rightarrow_{13} 02210C \Rightarrow_9 022101C \Rightarrow_7 0221011C \Rightarrow_{10} 02210112C \Rightarrow_5 02210112$.

По виду продукций определяем, что эта грамматика является контекстно-зависимой.

Пример 2. Построим грамматику G_{nn}, порождающую строки вида 0^n1^n, $n \in N$ (запись a^n означает n символов «a», расположенных подряд).

$G_{nn} = (\{S\}, \{0, 1\}, P, S)$,

где P состоит из двух продукций:

1. $S \to 0S1$.
2. $S \to \varepsilon$.

По виду продукций легко определить, что эта грамматика относится к типу контекстно-свободных. Однако эта грамматика не является регулярной, поскольку она не является левосторонней или правосторонней.

Пример 3. Построим грамматику, порождающую строки вида $0^n1^n2^n$, $n \in N$.

$G_{nnn} = (\{A, B, S\}, \{0, 1, 2\}, P, S)$,

где P состоит из семи продукций:

1. $S \to 0SAB$.
2. $S \to \varepsilon$.
3. $BA \to AB$.
4. $0A \to 01$.
5. $1A \to 11$.
6. $1B \to 12$.
7. $2B \to 22$.

По виду продукций определяем, что это контекстно-зависимая грамматика. Действительно, в левой части продукций 4–7. расположено по два символа, первый из которых терминальный, а для контекстно-свободных грамматик в левой части продукции должен находиться только один символ, который является нетерминальным.

Из четырех типов грамматик иерархии Хомского класс контекстно-свободных грамматик наиболее важен с точки зрения приложений к языкам программирования и компиляции. С помощью КС-грамматики можно определить большую часть синтаксической структуры языка программирования. Кроме того, она служит основой различных схем задания переводов.

Существует ли в некоторой фиксированной КС-грамматике вывод входной строки и если да, то определить вывод этой строки — один из основных вопросов, которые решает компилятор и другие программы идентификации и перевода. Для упрощения этой задачи применяются деревья вывода и эквивалентные преобразования грамматик. Для грамматик как средств описания языка требуется решать следующие проблемы.

Проблема принадлежности: «Даны описание языка определенного типа и строка ω, принадлежит ли ω этому языку?»

Проблема пустоты: «Дано описание языка определенного типа; пуст ли этот язык?» Пустота языка означает, что не существует строк, принадлежащих этому языку.

Проблема эквивалентности: «Даны два описания языков одинакового типа; совпадают ли эти языки?»

Доказано, что все эти проблемы разрешимы для класса КС-грамматик.

Очевидно, что грамматики в основном содержат рекурсии, т.е. из строки, содержащей некоторый нетерминальный символ, выводится за один или несколько шагов строка, содержащая тот же самый нетерминал и поэтому не являющаяся «финальной» строкой вывода, то есть собственно строкой языка. Напомним, что финалом вывода должна быть строка языка, состоящая только из терминальных символов. Грамматики, не содержащие рекурсий, порождают ***конечные языки*** и большого интереса не представляют. Грамматики можно классифицировать по виду содержащихся в них рекурсий, как это сделано при определении типа 3 грамматик. Праволинейные грамматики называют также ***праворекурсивными***, а леволинейные — ***леворекурсивными***. Грамматики, содержащие продукции вида $X \Rightarrow^* \alpha X \beta$, где $X \in N$, α, $\beta \in (N \cup T)^+$, называют ***самовключающими***.

От того, какого вида рекурсии содержит грамматика, зависит организация вывода для эффективного порождения необходимых строк языка.

Вопросы

1. Определите иерархию Хомского для грамматик.
2. Может ли:
   а) контекстно-свободная грамматика быть регулярной;
   б) регулярная грамматика не являться контекстно-свободной?
3. Всегда ли:
   а) контекстно-зависимая грамматика является регулярной;
   б) контекстно-свободная грамматика является неукорачивающей?
4. Какой тип языков порождает каждый из типов грамматик?
5. Приведите пример влияния контекста на определение значения слов «ключ», «лук» в предложениях русского языка. Что называют контекстом?
6. Приведите пример контекста в продукциях грамматик.
7. Приведите примеры продукций, содержащих левую рекурсию, правую рекурсию, самовключение.

Задания

1. Для грамматики $G_{сумма}$ проведите вывод другой возможной строки.
2. Докажите, что грамматика $G_{пп}$ действительно выводит все цепочки вида 0^n1^n. Приведите пример вывода.
3. Докажите, что грамматика $G_{ппп}$ действительно выводит все цепочки вида $0^n1^n2^n$. Приведите пример вывода.
4. Для грамматики $G = (\{A, B, S\}, \{a, b\}, P, S)$ определите тип в иерархии Хомского. Список продукций P следующий:
   а) $S \to aAB$, $A \to Bb$, $B \to \varepsilon$;
   б) $S \to aB$, $AB \to a$;
   в) $S \to ABA$, $A \to aB$, $B \to ab$;
   г) $S \to bA$, $A \to aB$, $B \to abA$.
5. Существуют ли регулярные конечные языки?

7.6. Регулярные выражения и языки

Языки, определяемые граматиками типа 3 иерархии Хомского, называются ***регулярными***. Для задания регулярных языков (множеств строк) используются также специальные выражения, называемые регулярными выражениями.

Определение

Регулярное выражение в некотором алфавите A определяется рекурсивно:

$\varnothing$ — регулярное выражение (пустое множество строк);

ε — регулярное выражение (пустая строка);

x — регулярное выражение, если $x \in A$ (x — символ алфавита A);

XY, $X \cup Y$, X^* — регулярные выражения, если X, Y — регулярные выражения, где

XY — конкатенация строк, определяемых выражениями X и Y;

$X \cup Y$ — объединение множеств строк, определяемых X и Y;

X^* — конкатенация любого числа строк, определяемых выражением X или пустая строка.

В регулярных выражениях для определения области действия операций используются скобки. Если скобки отсутствуют, то приоритет операций следующий:

1. X^*.
2. XY.
3. $X \cup Y$.

Пример 1. Выражение 10^* понимается как $1(0^*)$ и означает строки, первым символом которых является «1», а затем следует любое число нулей или ни одного нуля. Например, «100000000», «10», «1» и др.

Пример 2. Выражение $(10)^*$ означает любое количество «10», идущих подряд, или пустая строка. Например, «101010101010», «10», ε и др.

Пример 3. Выражение $aaa \cup bbb$ означает две строки — строку «*aaa*» или строку «*bbb*».

Пример 4. Выражение $a(a \cup b \cup c)^*$ означает любую строку, состоящую из символов «*a*», «*b*», «*c*», которая начинается с символа «*a*», например, «*abccbbaaabbcc*», «*aaaaaaaa*», «*accc*»,«*a*» и др.

Понятно, что регулярное выражение является строкой некоторого языка. Доказано, что язык можно задать с помощью регулярной грамматики тогда и только тогда, когда его можно задать с помощью регулярного выражения. Поэтому регулярный язык иногда называют регулярным множеством, имея в виду множество его строк.

Пример 5. Расмотрим грамматику $G_{число}$ из п. 7.4.

Эта грамматика является регулярной, поскольку каждая продукция из P имеет вид $A \to xB$ или $A \to x$, где A, $B \in N$, $x \in T^*$. Построим регулярное выражение, определяющее тот же язык, что и грамматика. Число, строящееся с помощью грамматики $G_{число}$, начинается знаком, но знак может и отсутствовать. Указание знака или его отсутствие реализуют продукции 2, 3, 4. Этому соответствует регулярное выражение $(+ \cup - \cup \varepsilon)$. Далее следует целая часть числа (продукции 1, 5, 6), которая определяется как целое число — одна или более цифр, причем нули в начале числа допускаются. Регулярное выражение для целой части следующее: $(0 \cup 1 \cup 2 \cup 3 \cup 4 \cup 5 \cup 6 \cup 7 \cup 8 \cup 9)\ (0 \cup 1 \cup 2 \cup \cup 3 \cup 4 \cup 5 \cup 6 \cup 7 \cup 8 \cup 9)^*$. Таким же образом определяется и регулярное выражение для дробной части, с тем отличием, что перед дробной частью находится запятая (продукция 7), и дробная часть может отсутствовать (продукция 8), поэтому мы добавим знак $\cup \varepsilon$.

Регулярное выражение для дробной части имеет вид:

$$(,(0 \cup 1 \cup 2 \cup 3 \cup 4 \cup 5 \cup 6 \cup 7 \cup 8 \cup 9)$$
$$(0 \cup 1 \cup 2 \cup 3 \cup 4 \cup 5 \cup 6 \cup 7 \cup 8 \cup 9)^*) \cup \varepsilon.$$

Объединив полученные выражения с помощью конкатенации, получим искомое регулярное выражение:

$$(+ \cup - \cup \varepsilon)\ (0 \cup 1 \cup ... \cup 9)(0 \cup 1 \cup ... \cup 9)^*$$

$$((,(0 \cup 1 \cup ... \cup 9)\ (0 \cup 1 \cup ... \cup 9)^*) \cup \varepsilon).$$

Это выражение определяет тот же язык, что и грамматика $G_{число}$.

Вопросы

1. Как связаны регулярные грамматики, регулярные выражения и регулярные языки?
2. Дайте определение регулярного выражения.
3. Как определяется приоритет операций в регулярных выражениях?
4. Существует ли язык, заданный с помощью регулярной грамматики, который невозможно задать с помощью регулярного выражения? Существует ли язык, заданный с помощью регулярного выражения, который невозможно задать с помощью регулярной грамматики?

Задания

1. Для следующих регулярных выражений дайте словесное описание множества строк и определите, принадлежит ли строка «1011» этому множеству:
   а) 10^*1^*;
   б) $1(01)^*1^*$;
   в) $(10)^*(11)^*$;
   г) $(10)^*1011$;
   д) $0^*(10 \cup 11)^*$;
   е) $1^*01(0 \cup 1)$;
   ж) $1(00)^*(11)^*$;
   з) $(1 \cup 00)(01 \cup 0)1^*$.
2. Опишите следующие языки с помощью регулярных выражений и грамматик:
   а) строки, начинающиеся с одного или более нулей, за которыми следует одна или более единиц;
   б) строки из букв русского языка, начинающиеся с гласной буквы;
   в) строки из букв русского языка, содержащие в качестве подстроки слово «компьютер»;
   г) правильные предложения русского языка, которые можно составить из набора слов: {Поздравляю, , ,Вы, очень, хорошо, плохо, отлично, не, решили, выполнили, задание, задачу, пример}.
3. Составьте регулярные выражения, определяющие следующие множества строк:
   а) {ε, 0};
   б) {0, 11};
   в) {0, 11, 000};
   г) {00010, 01010, 00011, 01011}.
4. Составьте регулярные выражения для грамматик $G = (\{A, B, S\}, \{0, 1\}, P, S)$ со следующими наборами продукций P:

а) $S \to 0A$, $S \to 1B$, $A \to 0$, $B \to 0$;
б) $S \to 1A$, $S \to 0$, $S \to \varepsilon$, $A \to 0B$, $B \to 1B$, $B \to 1$;
в) $S \to 1B$, $S \to 0$, $A \to 1A$, $A \to 0B$, $A \to 1$, $A \to 0$, $B \to 1$.

7.7. Деревья выводов. Стратегии выводов

Дерево вывода. Стратегии вывода: сверху вниз, слева направо, снизу вверх

В грамматике может быть несколько выводов, эквивалентных в том смысле, что в каждом выводе применяются одни и те же продукции в одних и тех же местах, но в различном порядке. Определить понятие эквивалентности двух выводов для грамматик произвольного вида сложно, но в случае КС-грамматик можно ввести удобное графическое представление класса эквивалентных выводов, называемое деревом вывода.

Определение

Помеченное упорядоченное дерево D называется ***деревом вывода*** (или ***деревом разбора***) в КС-грамматике $G = (N, T, P, S)$, если выполнены следующие условия:

1. Корень дерева D помечен S, а каждая вершина дерева D — символом из $N \cup T \cup \{\varepsilon\}$.
2. Для любой внутренней вершины X_0 дерева и упорядоченного списка ее сыновей $X_1, X_2, \ldots, X_n$ множество продукций грамматики P включает такую продукцию $A \to a_1, a_2, \ldots, a_n$, что $A, a_1, a_2, \ldots, a_n$ являются пометками вершин $X_0, X_1, X_2, \ldots, X_n$ соответственно.
3. Если корень дерева имеет единственного потомка, помеченного ε, то этот потомок доминирует над деревом, состоящим из единственной вершины ε, и $S \to \varepsilon$ — продукция из множества P.

На рис. 7.2 изображены деревья выводов в грамматике G с продукциями $S \to aSbS \mid bSaS \mid \varepsilon$.

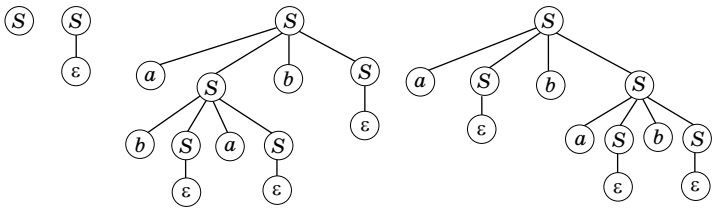


Рис. 7.2. Деревья выводов в грамматике с продукциями $S \to aSbS \mid bSaS \mid \varepsilon$

Поскольку любое дерево вывода является упорядоченным, для его висячих вершин существует единственное линейное упорядочение. Выписывая пометки висячих вершин в этой упорядоченности, можно получить строку, выводы которой представлены этим деревом.

Вывод в грамматиках схематически представляется в виде треугольника, изображенного на рис. 7.3, и заполняется в соответствии с набором продукций грамматики.

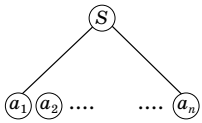
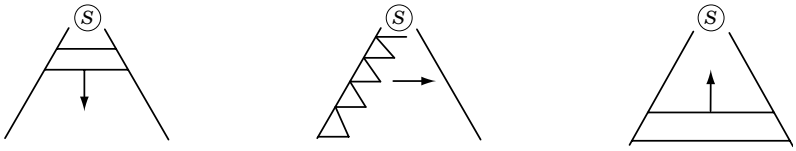


Рис. 7.3. Схема вывода в грамматиках

В большинстве случаев заполнение треугольника нельзя разумно выполнить за один шаг, и обычно оно получается при помощи последовательности поддеревьев. Эти последовательности поддеревьев могут быть получены несколькими стратегиями, которые изображены на рис. 7.4.



а) Сверху вниз

б) Слева направо

в) Снизу вверх

Рис. 7.4. Стратегии вывода (разбора) в грамматиках

При грамматическом разборе порядок заполнения дерева сводится к одной из двух стратегий: ***сверху вниз*** (рис. 7.4,*а*) и ***снизу вверх*** (рис. 7.4,*в*). При заполнении сверху вниз — ***стратегия нисходящего анализа*** — все предшественники каждой заполняемой в дереве внутренней вершины к данному моменту уже заполнены. Заполнение снизу вверх — ***стратегия восходящего анализа*** — характеризуется тем, что все преемники каждой заполняемой в дереве вершины уже принадлежат построенной части дерева. Таким образом, при заполнении дерева сверху вниз движение осуществляется от его корня к висячим вершинам, а при заполнении снизу вверх — наоборот. При заполнении слева направо — ***стратегия горизонтального анализа*** (рис. 7.4,*б*) — на каждом шаге происходит замена самого левого нетерминального символа терминальным.

Вопросы

1. Что такое вывод в грамматике?
2. Какие выводы эквивалентны?

3. Что такое дерево вывода?
4. Какой символ является вершиной дерева вывода?
5. Символы какого алфавита являются листьями дерева вывода?
6. Какие стратегии вывода в грамматиках вы знаете?

Задания

1. Примените различные стратегии вывода к грамматикам, приведенным в качестве примеров в пп. 7.3–7.5.
2. Постройте деревья для выводов, полученных при решении задания 1.
3. G — грамматика, $N = \{S\}$, $T = \{a, b, c\}$, начальный символ — S и продукции: $S \to bcS$, $S \to bbS$, $S \to a$, $S \to cb$. Постройте деревья выводов для:
   а) *bcbba*;
   б) *bbbcbba*;
   в) *bcabbbbbcb*.
4. G — грамматика, $N = \{A, B, C, S\}$, $T = \{a, b, c\}$, начальный символ — S и продукции: $S \to AB$, $A \to Ca$, $B \to Ba \mid Cb \mid b$, $C \to cb \mid b$. Используя различные стратегии вывода, определите принадлежат ли данные строки языку, порождаемому грамматикой G:
   а) *baba*;
   б) *abab*;
   в) *cbaba*;
   г) *bbbcba*.

7.8. Построение грамматики языка программирования

Алфавит языка, служебные слова, идентификаторы, операторы

Рассмотрим некоторые аспекты формального представления языка программирования на примере языка Pascal.

Текст программы должен содержаться в дисковом файле и представлять собой последовательность строк, состоящих из символов, образующих алфавит языка. Строки программы завершаются специальными управляющими символами, не входящими в алфавит.

Алфавит языка Pascal состоит из следующих символов:

- Заглавные и строчные латинские буквы и символ '_':

  $A, B, C..., X, Y, Z, a, b, c, ..., x, y, z, _$.

Буквы используются для формирования идентификаторов и служебных слов.

- Десять арабских цифр от 0 до 9:

  0, 1, 2, 3, 4, 5, 6, 7, 8, 9.

Цифры используются для записи чисел и идентификаторов.

- Двадцать два специальных символа:

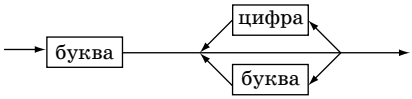
  + − * / = > < . , ; : @ ‘ ( ) [ ] { } # $ ^.

Специальные символы используются для конструирования знаков операций, выражений, комментариев, а также как синтаксические разделители.

Символы из алфавита языка используются для построения базовых элементов программ — лексем. Лексема — минимальная единица языка, имеющая самостоятельный смысл.

В числе лексем языка Pascal находятся служебные (зарезервированные) слова. Это ограниченная группа слов, построенных из букв. Каждое служебное слово представляет собой неделимое образование, смысл которого фиксирован в языке. Например: begin, do, for, or, string, virtual.

Идентификаторы (имена) — еще один вид лексем, используемых в языке Pascal. Они обозначают имена переменных, констант, типов, меток, процедур и функций и формируются из букв и цифр согласно следующей диаграмме:



Построим грамматику G_{id}, которая порождает идентификаторы:

G_{id} = ({БЦ, Буква, Цифра, Идент.},
{A,..,Z, a,.., z,_,0,1,..,9}, P, Идент.),

где P состоит из набора продукций:

1. <Идент.>:: = <Буква> <БЦ>.
2. <БЦ>:: = <Буква> <БЦ> | <Цифра> <БЦ> | ε.
3. <Буква>:: = A | B | C | ... | X | Y | Z | a | b | c | ... | x | y | z | _.
4. <Цифра>:: = 0 | 1 | 2 | ... | 9.

Центральным элементом языка Pascal является оператор. Набор операторов языка Pascal составляет минимальное множество конструкций, необходимых для наглядного и однозначного представления алгоритмов.

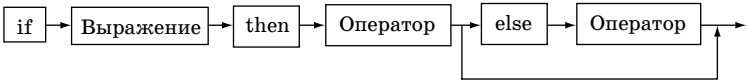
Язык содержит следующие операторы:

– оператор присваивания;

– оператор процедуры;

– оператор перехода;
– составной оператор;
– условный оператор;
– оператор варианта;
– оператор цикла с предусловием;
– оператор цикла с постусловием;
– оператор цикла с параметром;
– оператор над записями;
– пустой оператор.

Рассмотрим грамматику G_{if}, порождающую условные операторы, которые схематично можно представить так:



G_{if} = ({Оператор, Выраж., Иначе, Условный Оператор}, A, P, Условный Оператор),

где A — алфавит языка Pascal.

P состоит из набора продукций:

1. <Условный Оператор>:: = if <Выраж.>
   then <Оператор> <Иначе> ;
2. <Иначе>→ else <Оператор> |ε
3. <Оператор>::=begin<Оператор>;<Оператор>end |<Оператор>
   ….

Очевидно, что в наборе продукций не хватает продукций порождения операторов, выражений и др., которые не приводятся здесь из-за их громоздкости. Только разновидностей операторов языка Pascal, как было перечислено ранее, существует около 10.

Таким образом, для создания грамматики языка программирования необходимо задать такие продукции для порождения всех его единиц, как оператор, идентификатор, выражение, описание, определение, число и др. Первой продукцией в грамматике языка программирования является продукция, определяющая конструкцию программы.

Вопросы

1. Что составляет алфавит языка программирования?
2. Что такое идентификатор, оператор, условный оператор, оператор цикла?

3. Каков принцип построения грамматики языка программирования?
4. В каких случаях строка, построенная с помощью грамматики G_{id}, не будет являться допустимым идентификатором в программе?

Задания

1. Запишите в форме Бэкуса — Наура грамматику, которая которая будет порождать оператор цикла из знакомого вам языка программирования.
2. Найдите записанную в форме Бэкуса — Наура грамматику известного вам языка программирования. Попытайтесь построить неизвестные вам синтаксические конструкции языка, пользуясь записанной в такой форме грамматикой.
3. Постройте грамматику, выводящую идентификаторы используемого вами языка программирования. Запишите, если возможно, соответствующее регулярное выражение.
4. Постройте контекстно-свободную грамматику для любого известного вам оператора цикла. Запишите, если возможно, соответствующее регулярное выражение.

раздел 8

Алгоритмы

8.1. Понятие алгоритма

Интуитивное понятие алгоритма, алфавитные операторы, детерминированные и недетерминированные алгоритмы

Как и понятие множества, в самом общем случае понятие алгоритм не имеет строгого математического (и, если угодно, логического) определения, и приходится довольствоваться интуитивными представлениями, основанными на опыте. Поскольку опыт в разных областях свой, то и слово «алгоритм» употребляется в разных смыслах: алгоритм поведения, алгоритм построения, алгоритм решения проблемы (задачи) и т.д. Иначе говоря, под алгоритмом в широком смысле понимают последовательность определенных действий или операций, более или менее элементарных.

Здесь мы употребляем термин «алгоритм» исключительно в смысле ***процедуры вычисления***, допускающей программирование и компьютерную реализацию. При этом вычисление рассматривается как в числовом, так и в алгебраическом (символьном) алфавитах. Такое понимание алгоритма также не формализовано, однако позволяет высказать ряд требований, которым должна удовлетворять совокупность стандартных преобразований, образующих «вычислительный» алгоритм. Прежде всего, ***распознаваемость*** и ***однозначная определенность*** (***детерминированность***) для любого представителя из класса пользователей, которым адресуется алгоритм. Далее — ***универсальность*** (***массовость***) алгоритма, то есть способность «перерабатывать» любые исходные данные из допустимого множества. Наконец, ***дискретность*** и ***конечность*** (результативность) ***алгоритма*** — получение окончательного

(«выходного») результата за конечное число шагов или получение информации о том, что результат существует.

В качестве примеров наметим принципиальный план построения алгоритмов решения двух известных задач; полное описание этих алгоритмов требует значительно больших усилий.

Пример. Рассмотрим алгоритм сложения целых чисел в десятичной системе счисления. Этот алгоритм перерабатывает строки вида «$a + b$» в алфавите, состоящем из 10 цифр и знака сложения, в строки в том же алфавите, из которого исключен знак суммы. Конечная система правил, определяющих алгоритм, состоит из правила поразрядного сложения, правила нахождения соответствующих друг другу разрядов слагаемых (в том числе младших разрядов), правил сложения цифр (таблица сложения) и правила переноса в старший разряд в случае «переполнения» младшего.

Пример. Рассмотрим игру в шахматы, трактуя ее как преобразование строк в конечном алфавите. В качестве алфавита здесь естественно использовать стандартные в шахматной теории символы. Строками должны быть конечные последовательности символов алфавита. Некоторые из таких последовательностей представляют собой шахматные позиции, другие — просто бессмысленные наборы символов (например, *aee*11). Алгоритм задается лишь для строк, представляющих осмысленные позиции, и заключается в преобразовании любой заданной позиции в позицию, возникающую из нее после выполнения очередного хода. Как известно, правила движения шахматных фигур еще не определяют какой-нибудь разумный выбор хода в той или иной конкретной позиции и, следовательно, не позволяют построить осмысленный алгоритм игры. Однако, как показывает опыт, можно составить конечную систему стратегических правил, позволяющих в каждой конкретной позиции выбирать единственный наилучший (в некотором смысле) ход. Присоединяя эту систему правил к правилам движения фигур, мы получаем алгоритм, который естественно назвать алгоритмом шахматной игры. Следует сразу оговориться, что в зависимости от того, каким образом определяется качество позиции, меняется и смысл понятия «наилучшего» хода. Это обстоятельство определяет существование не одного единственного алгоритма шахматной игры, а множества таких алгоритмов.

Мы оставляем в стороне вопрос о вероятностных алгоритмах, где выбор образа для той или иной строки определяется не однозначно, а случайно, в соответствии с какими-либо вероятностными

критериями. Ясно, что применительно к шахматным алгоритмам рассмотрение случайных переходов является не уместным. Таким образом, традиционно будем называть алгоритмом лишь строго детерминированную систему правил.

> **Определение**
> Алгоритм, при котором в каждом шаге можно однозначно определить следующий шаг, называется ***детерминированным***.

Если в системе правил присутствует недетерминированность, то речь идет об исчислении.

В соответствии с принятой нами дискретной точкой зрения преобразование информации можно представить как отображение множества строк в некотором конечном алфавите во множество строк в том же самом или любом другом конечном алфавите. Назовем такие отображения ***алфавитными операторам***.

Алгоритмы можно трактовать как алфавитные операторы, задаваемые с помощью конечных систем правил. В таком определении также нет математической точности. Не преодолев этого недостатка, нельзя сделать понятие алгоритма предметом математической теории. Были предприняты многочисленные попытки уточнения понятия алгоритма, которые привели к установлению способов точного задания алгоритмов различных классов.

Вопросы

1. Приведите примеры алгоритмов, используемых в повседневной жизни.
2. Дайте определение алфавитного оператора.
3. Какой алгоритм называется детерминированным? Приведите пример.

8.2. Нормальные алгоритмы Маркова

Нормальные алгоритмы Маркова, принцип нормализации, алгоритмическая разрешимость

Алгоритмы А. А. Маркова, называемые также ***нормальными алгоритмами***, преобразуют строки, заданные в любом конечном алфавите A, в строки в том же самом алфавите A. Напомним, что строка в алфавите A — это упорядоченная конечная совокупность символов этого алфавита. Нормальный алгоритм задается конечной последовательностью подстановок. ***Подстановка*** — это упорядоченная пара строк, необязательно одинаковой длины,

разделенная символом «→». Например, подстановкой является $ab \to cd$. Подстановка применима к исходной строке, если эта строка содержит в качестве подстроки первую компоненту подстановки. В результирующей строке имеем вместо первой компоненты подстановки вторую. Например, применив подстановку $ab \to cd$ к строке «*sabfg*», получим строку «*scdfg*». К строке «*sbaff*» эта подстановка не применима, поскольку подстроки «*ab*» в строке «*sbaff*» нет. Подстановки также могут быть помечены как заключительные. Применяют нормальный алгоритм к исходной строке следующим образом:

1) Из списка подстановок выбираем первую подстановку, которая применима к исходной строке.

2) Если ни одна подстановка из списка подстановок не применима к исходной строке, то алгоритм прекращает работу.

3) Применяем подстановку из п. 1 к исходной строке, получаем результирующую строку.

4) Если выполненная подстановка является заключительной, то алгоритм прекращает работу. Иначе переходим к п. 5.

5) Результирующую строку считаем исходной и переходим к п. 1.

Примечание: алгоритм может не содержать заключительных подстановок.

Порядок применения подстановок, указанный в п. 1, имеет большое значение для правильной работы алгоритма. Список подстановок просматривается, начиная с подстановки № 1. Если к исходной строке применить подстановку № 1 нельзя, то рассматривается подстановка № 2, и т.д. Если и последняя подстановка не применима к исходной строке, то алгоритм прекращает работу, как указано в п. 2.

Если нормальный алгоритм, примененный к некоторой исходной строке, не прекращает работу, а работает бесконечно, то считается, что он не применим к этой исходной строке, поскольку не выдает результата.

Пример. Рассмотрим пример нормального алгоритма, который располагает буквы в алфавитном порядке. Алфавит — $A = \{a, b, c\}$. Набор подстановок:

1. $ba \to ab$.
2. $ca \to ac$.
3. $cb \to bc$.

Применим этот алгоритм к строке «*cbabc*». В процессе работы алгоритма последовательно получаем строки:

«*cbabc*» исходная;
«*cabbc*» подстановка 1;
«*acbbc*» подстановка 2;
«*abcbc*» подстановка 3;
«*abbcc*» подстановка 3.

Результирующая строка — «*abbcc*».

Пример. В алфавите $A = \{a, b, c, d, e, ..., y\}$ к строке «*abcd*» список подстановок

1. $a \to c$;
2. $b \to d$;
3. $cdc \to ab$

не применим, так как возникновение цикла

$$(abcd) \xrightarrow{1} (cbcd) \xrightarrow{2} (cdcd) \xrightarrow{3} (abcd)$$

приводит к бесконечной работе алгоритма Маркова, если не указано, какая из трех подстановок в списке является заключительной.

Пример. Алфавит $A = \{1, 0, \cap, \cup, (,)\}$. Набор подстановок:

1. $1 \cap 0 \to 0$.
2. $1 \cap 1 \to 1$.
3. $0 \cap 1 \to 0$.
4. $0 \cap 0 \to 0$.
5. $1 \cup 0 \to 1$.
6. $1 \cup 1 \to 1$.
7. $0 \cup 1 \to 1$.
8. $0 \cup 0 \to 0$.
9. $(1) \to 1$.
10. $(0) \to 0$.

Применим нормальный алгоритм к строке «$(0 \cup 1) \cap 1 \cup 0$». В процессе работы алгоритма последовательно получаем строки:

«$(0 \cup 1) \cap 1 \cup 0$» исходная;
«$(0 \cup 1) \cap 1$» подстановка 5;
«$(1) \cap 1$» подстановка 7;
«$1 \cap 1$» подстановка 9;
«1» подстановка 2.

Результирующая строка — «1».

Можно показать, что любой алгоритм эквивалентен некоторому нормальному алгоритму. Иначе говоря, для любого алгоритма над произвольным конечным алфавитом A можно построить нормальный алгоритм, перерабатывающий строки точно так же, как и исходный алгоритм.

Таким образом, можно сформулировать ***принцип нормализации алгоритма***, который звучит так: всякий алгоритм в алфавите A

эквивалентен некоторому нормальному алгоритму в алфавите A. Коротко формулируют так: ***всякий алгоритм в A нормализуем***.

Принцип нормализации, с одной стороны, утверждает существование нормального алгоритма для решения каждой задачи, для которой существует алгоритм в A. С другой стороны, он позволяет утверждать, что если для решения какой-либо задачи нормальный алгоритм невозможен (его не существует), то не имеет смысла искать какой-либо другой алгоритм, т.е. принцип нормализации является средством исследования вопроса ***алгоритмической разрешимости*** или ***неразрешимости задачи***.

Вопросы

1. Что представляют собой алгоритмы Маркова?
2. В каком порядке подстановки применяются к исходной строке?
3. В каком случае считается, что подстановка применима к исходной строке?
4. В каком случае алгоритм Маркова прекращает работу?
5. В каком случае считается, что алгоритм Маркова не применим к исходной строке?
6. Сформулируйте принцип нормализации.

Задания

1. Примените алгоритм из примера 1 к строкам «*cabcc*», «*cccba*».
2. Для алфавита $A = (0, 1, 2, 3)$ постройте нормальный алгоритм, располагающий цифры исходной строки в порядке убывания.
3. Примените алгоритм из примера 3 к строкам «$1 \cup (0 \cap 1) \cup 1$», «$(0 \cap 1 \cup 0) \cup 0 \cap 1$».
4. В алфавите $A = \{0, 1, \neg, \cap, \cup, (,)\}$ постройте нормальный алгоритм, который вычисляет значение выражения аналогично примеру 3 (в выражении может присутствовать знак «$\neg$» — «не»).

8.3. Алгоритмы и рекурсивные функции

Вычислимая функция, тезис Черча, базовые рекурсивные функции, операторы построения рекурсивных функций

Изучая понятие алгоритма, естественно обратиться к уже изученному понятию функции. Считается, что такие объекты, как рекурсивные функции существуют или не существуют в точности тогда же, когда и алгоритмы.

Теоретико-множественное определение функции звучит так: функцией $y = f(x)$ называется отображение множества X во

множество Y, которое каждому элементу $x \in X$ ставит в соответствие в точности один элемент $y \in Y$. Алгоритм можно считать конструктивным правилом, ставящим в соответствие каждому допустимому значению независимого переменного значение функции, т.е. правилом, индуцирующим функцию. Не для всякой функции существует алгоритм, вычисляющий ее значение.

> **Определение**
> Функция, для которой существует какой-либо конструктивный способ получения ее значения по значениям аргумента *за конечное число шагов*, называется ***вычислимой***.

Очевидно, что функции, индуцируемые алгоритмами, вычислимы. В некоторых вопросах вместо рассмотрения алгоритмов можно обойтись изучением вычислимых функций, например, если будет доказано, что для решения некоторой проблемы не существует вычислимой функции, то тем самым будет доказано и несуществование алгоритма. Изучая различные вычислимые функции, А. Черч в 1936 г. высказал утверждение, известное теперь как тезис Черча.

Тезис Черча: Всякая вычислимая функция от n натуральных аргументов, принимающая натуральные значения, тождественно равна некоторой рекурсивной функции с тем же числом независимых переменных.

Тезис Черча сопоставляет интуитивное понятие вычислимой функции с точным математическим понятием рекурсивной функции. Утверждается, что каков бы ни был алгоритм, оперирующий целыми неотрицательными числами, существует алгоритм, сопутствующий рекурсивной функции, который ему эквивалентен. Алгоритмом, сопутствующим рекурсивной функции, называется алгоритм вычисления ее значения (т.е. являющийся законом задания рекурсивной функции). Рассмотрим формальное определение класса рекурсивных функций.

Рекурсивной является функция, использующая саму себя в своем определении. Рекурсии мы встречаем как в математике, так и в повседневной жизни. Например, рекурсивное определение строки можно сформулировать так: «Строкой называется либо пустая строка, либо строка, к которой приписана буква».

Второй пример рекурсии — метод математической индукции для доказательств. Этот метод состоит в следующем:

1) Проверяем истинность утверждения для x_0.
2) Предполагаем, что оно истинно для x_i.

3) Если, используя 2), удается доказать, что утверждение истинно для x_{i+1}, то можем сделать вывод об истинности этого утверждения для x_i при любом $i \in N$.

Класс рекурсивных функций строится путем указания некоторого числа элементарных (базовых) функций и трех операций над функциями, применение которых и порождает весь класс.

Определение

Следующие функции объявляются элементарными или ***базовыми функциями***, и их область определения, и область значений задаются из расширенного натурального ряда $N_0 = N \cup \{0\}$:

1) Функции любого числа переменных $O_n(x_1, ..., x_n)$, тождественно равные нулю, где n — число переменных:

$$O_n(x_1, ..., x_n) = 0, \forall x_i.$$

2) Функции выбора $V_{n,i}(x_1, ..., x_n)$ или селекторные функции, где n — число переменных, i — номер переменного в списке переменных функции, значение которого принимает селекторная функция: $V_{n,i}(x_1, ..., x_n) = x_i$.

3) Функция следования, обозначающаяся через $\lambda(x) = x + 1$. Значением функции является число, непосредственно следующее за x.

Нижний индекс около знака функции обозначает ***ранг функции*** — количество переменных этой функции. Например, функция F_n зависит от n переменных. Нижний индекс около обозначения переменной означает порядковый номер этой переменной в списке переменных функции.

Для каждой базовой функции существует простой алгоритм, вычисляющий ее значение. Он называется ***сопутствующим алгоритмом*** этой ***функции***.

Перечислим ***операторы***, с помощью которых из рекурсивных функций можно получать новые рекурсивные функции.

Оператор подстановки (суперпозиции) S порождает функцию Φ_m от m аргументов $(x_1, ..., x_m)$ по функциям F_n от n аргументов $f_1, ..., f_n$. Функция Φ_m определяется условием $\Phi_m = F_n(f_1, ..., f_n)$. Оператор подстановки вычисляет значения функций $f_1, ..., f_n$, зависящих от m аргументов, и принимает их за значения соответствующих этим функциям аргументов функции F_n, после чего вычисляет значение функции F_n. Результат считается значением функции Φ_m и обозначается

$$\Phi_m = S(F_n, f_1, f_2, ..., f_n).$$

Оператор примитивной рекурсии R по двум функциям φ_{n-1} и Ψ_{n+1} строит функцию n независимых переменных $\Phi_n = R(\varphi_{n-1}, \Psi_{n+1})$:

$$\Phi_n(x_1, ..., x_{n-1}, 0) = \varphi_{n-1}(x_1, ..., x_{n-1});$$

$$\Phi_n(x_1, ..., x_{n-1}, i+1) = \Psi_{n+1}(x_1, ..., x_{n-1}, i, \Phi_n(x_1, ..., x_{n-1}, i)),$$

где $i = 0, 1, ..., x_{n-1}$.

Оператор минимизации μ: $\varphi_{n+1} \to \Phi_n$ по заданной функции $\varphi_{n+1}(x_1, ..., x_n, x_{n+1})$ строит «минимизированную» функцию Φ_n от меньшего числа аргументов по следующему правилу. Аргументу x_{n+1} функции $\varphi_{n+1}(x_1, ..., x_n, x_{n+1})$ придают последовательно значения 0, 1, 2 и т.д. до тех пор, пока функция φ_{n+1} не станет первый раз равной нулю. Полученное значение x_{n+1} принимают за значение результирующей функции:

$$\Phi_n(x_1, ..., x_n) = x_{n+1}, \text{ если } \varphi_{n+1}(x_1, ..., x_n, x_{n+1}) = 0.$$

Если до окончания процесса на каком-нибудь шаге φ_{n+1} станет неопределенной, то процесс безрезультатно обрывается.

Рекурсивными называются базовые функции и функции, получаемые путем применения к базовым функциям конечного числа операторов подстановки, рекурсии и минимизации. Как уже упоминалось, каков бы ни был алгоритм, оперирующий целыми неотрицательными числами, существует эквивалентный ему алгоритм, сопутствующий некоторой рекурсивной функции. Это означает, что выполнение алгоритма в определенном смысле эквивалентно вычислению значения рекурсивной функции.

Рассмотрим простейшие рекурсивные функции.

Функция $F(x) = x + 1$ совпадает с базовой функцией $\lambda(x)$, а значит является рекурсивной.

Функция $F(x, y) = x + y$ также является рекурсивной. Докажем это.

1) Построим рекурсивную функцию $f(x, y, z) = z + 1$, применив оператор подстановки к базовым функциям $V_{3,3}(x, y, z) = z$ и $\lambda(z) = z + 1$. Подставим $\lambda(z)$ вместо z. Полученная функция $V_{3,3}(x, y, \lambda(z)) = z + 1$ эквивалентна функции $f(x, y, z) = z + 1$.

2) Применим оператор рекурсии к функциям $V_{1,1}(x) = x$ и $f(x, y, z) = z + 1$.

Положив в определении оператора рекурсии $\Phi_2(x_1, x_2) = F(x_1, x_2)$, $x_1 = x$, $x_2 = y$, $x_3 = z$, $\Psi_{2+1}(x, y, z) = f(x, y, z)$, $\Phi_2(x, 0) = F(x, 0) = V_{1,1}(x) = x$, получаем последовательно:

$$F(x, 0) = V_{1,1}(x) = x$$
$$F(x, 0) = f(x, 0, F(x, 0)) = f(x, 0, x) = x + 1$$
$$F(x, 2) = f(x, 1, F(x, 1)) = f(x, 1, x + 1) = x + 2$$
...
$$F(x, y) = f(x, y - 1, F(x, y - 1)) = F(x, y - 1) + 1 =$$
$$= x + (y - 1) + 1 = x + y.$$

Искомая функция $F(x, y) = x + y$ представлена с помощью оператора рекурсии R через базовые функции: $F(x, y) = R(V_{1,1}(x), f(x, y, z))$. Аналогично доказывается, что $F(x, y) = x \cdot y$ также представима в виде рекурсивной функции.

Заметим, что алгоритм, сопутствующий рекурсивной функции, не является единственным. Так, в примере 1 на первом шаге вычисления функции $F(x, y) = x + y$ можно построить функцию f как $f(x, y, z) = \lambda(V_{3,3}(x, y, z)) = z + 1$.

Согласно тезису Черча, для любой вычислимой функции можно построить эквивалентную ей рекурсивную функцию. Если для решения какой-либо задачи можно построить рекурсивную функцию, то существует и алгоритм, решающий эту задачу. В противном случае не существует и алгоритма, решающего эту задачу.

Вопросы

1. Дайте определение вычислимой функции.
2. Приведите примеры рекурсии.
3. Любая ли вычислимая функция может быть представлена в рекурсивном виде?
4. Для чего в теории алгоритмов применяются рекурсивные функции?
5. Построить алгоритм вычисления функции $F(x, y) = x \cdot y$ через базовые функции и операторы.

8.5. Примеры построения алгоритмов

Правильный алгоритм, алгоритм сортировки вставками, алгоритм сортировки слиянием, принцип «разделяй и властвуй»

В программировании алгоритм — это формально описанная вычислительная процедура, использующая исходные данные, называемые также ***входом алгоритма*** или его ***аргументом***,

и выдающая результат вычислений на выход. Формулировка задачи содержит требования, которым должно удовлетворять решение задачи, а алгоритм, решающий эту задачу, находит объект, удовлетворяющий этим требованиям.

Построение алгоритмов рассмотрим на примере задачи сортировки:

Вход: Последовательность n чисел $(a_1, a_2, ..., a_n)$.

Выход: Перестановка $a'_1, a'_2, ..., a'_n$ исходной последовательности, для которой $a'_1 \le a'_2 \le ... \le a'_n$.

Например, получив на вход (31, 41, 59, 26, 41, 58), алгоритм сортировки должен выдать на выход (26, 31, 41, 41, 58, 59).

Многие алгоритмы используют сортировку в качестве промежуточного шага. Имеется много разных алгоритмов сортировки; выбор одного из них в конкретной ситуации зависит от длины сортируемой последовательности, от того, в какой степени она уже отсортирована, а также от типа имеющейся памяти.

> **Определение**
> Алгоритм считают ***правильным***, если на любом допустимом (для данной задачи) входе он заканчивает работу и выдает результат, удовлетворяющий требованиям задачи. В этом случае говорят, что алгоритм решает данную вычислительную задачу.

Неправильный алгоритм может (для некоторого входа) вовсе не остановиться или дать неправильный результат. Впрочем, неправильный алгоритм может быть полезен, если ошибки достаточно редки. Но это все же, скорее, исключение, чем правило.

Алгоритм может быть записан на русском, английском или любом естественном языке в виде компьютерной программы или даже в машинных кодах. Важно только, чтобы процедура вычислений имела точное описание, не допускающее двусмысленных толкований на каждом шаге.

В наших примерах будем записывать алгоритмы с помощью псевдокода, напоминающего языки программирования. Здесь иногда позволительно описывать действия алгоритма «своими словами», если так получается понятнее. Кроме того, опускаются технологические подробности (обработка ошибок и др.), которые необходимы в реальной программе.

Перечислим основные соглашения, которые используются в псевдокоде:

1. Знак «←» означает присваивание значения.

2. Циклы *while*, *for*, *repeat* и условные конструкции *if*, *then*, *else* имеют тот же смысл, что и в Pascal.

3. Индекс массива пишется в квадратных скобках; $A[i]$ есть i-й элемент массива A. Знак «..» выделяет часть массива. Символ $A[1..j]$ обозначает участок массива A, включающий $A[1]$, $A[1]$, ..., $A[j]$.

4. Отступ от левого поля указывает на уровень вложенности. Приведем пример записи процедуры:

```
PROCEDURE 1 (X, n, s, k)
1. s ← 0
2. k ← 0
3. for i ← 1 to n do
4. s ← s + X[i]
5. if s = 0 then
6. k ← i
7. x ← k + 1.
```

Тело цикла for (строка 3) состоит из строк 4–6, а к условию if (строка 5) относится только строка 6, но не 7. Это делает излишним специальные команды типа begin и end для начала и конца блока. (В реальных языках программирования такое соглашение применяется редко, поскольку затрудняет чтение программ, переходящих со страницы на страницу.)

5. Переменные (в нашем примере — i) локальны внутри процедуры.

6. Символ ◊ начинает комментарий, идущий до конца строки.

Сортировка вставками

Сортировка вставками удобна для сортировки коротких последовательностей. Именно таким способом обычно сортируют карты: держа в левой руке уже упорядоченные карты и взяв правой рукой очередную карту, мы вставляем ее в нужное место, сравнивая с имеющимися и идя справа налево.

Запишем этот алгоритм в виде процедуры INSERTION-SORT, параметром которой является массив $A[1..n]$ (последовательность длины n, подлежащая сортировке). Обозначим число элементов в массиве A через length[A]. Последовательность сортируется «на месте» без дополнительной памяти: помимо массива мы используем лишь фиксированное число ячеек памяти. После выполнения процедуры INSERTION-SORT массив A упорядочен по возрастанию.

```
INSERTION-SORT (A)
1   for j ← 2 to length [A] do
2       key ← A[j]
3       ◊ добавить A[j] к отсортированной части A[1...j − 1]
4       i ← j − 1
5       while i > 0 and A[i] > key do
6       A[i + 1] ← A[i]
7       i ← i − 1
8       A[i + 1] ← key.
```

На рис. 8.1 показана работа алгоритма при $A = \{5, 2, 4, 6, 1, 3,\}$, $n = 6$, индекс j указывает очередную карту (только что взятую со стола). Участок $A[1...j-1]$ составляют уже отсортированные карты (левая рука), $A[j+1...n]$ — еще не просмотренные. В цикле for индекс j пробегает массив слева направо. Мы берем элемент $A[j]$ (строка 2 алгоритма) и сдвигаем идущие перед ним и больше его по величине элементы (начиная с $(j-1)$-го) вправо, освобождая место для взятого элемента (строки 4–7). В строке 8 элемент $A[j]$ помещается на освобожденное место.

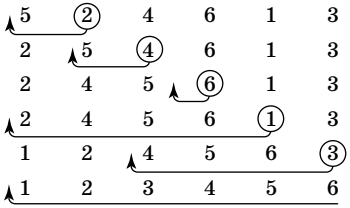


Рис. 8.1. Работа процедуры INSERTION-SORT
Исходная строка — $A = \{5, 2, 4, 6, 1, 3, \}$. Позиция j показана кружком

Сортировка слиянием. Принцип «разделяй и властвуй»

Есть много стандартных приемов, используемых при построении алгоритмов. Сортировка вставками является примером алгоритма, действующего по шагам: к отсортированной части массива добавляются элементы один за другим. Покажем в действии другой подход, который называют «разделяй и властвуй», и построим с его помощью значительно более быстрый алгоритм сортировки.

Многие алгоритмы по природе своей рекурсивны: решая некоторую задачу, они вызывают самих себя для решения ее подзадач. Идея метода «разделяй и властвуй» состоит как раз в этом. Сначала

задача разбивается на несколько подзадач меньшего размера. Затем эти задачи решаются (с помощью рекурсивного вызова — или непосредственно, если размер достаточно мал). Наконец, их решения комбинируются, и получается решение исходной задачи.

Для задачи сортировки эти три этапа выглядят так. Сначала массив разбивается на две половины меньшего размера. Затем каждая из половин сортируется отдельно. После этого остается соединить два упорядоченных массива половинного размера в один. Рекурсивное разбиение задачи на меньшие происходит до тех пор, пока размер массива не дойдет до единицы (любой массив длины 1 можно считать упорядоченным).

Соединение двух упорядоченных массивов в один выполняется с помощью вспомогательной процедуры MERGE (A, p, q, r). Параметрами этой процедуры являются массив A и числа p, q, r, указывающие границы сливаемых участков. Процедура предполагает, что $p \le q \le r$ и что участки $A[p..q]$ и $A[q+1..r]$ уже отсортированы, и сливает их в один участок $A[p..r]$.

Теперь составим процедуру сортировки слиянием MERGE-SORT (A, p, r), которая сортирует участок $A[p..r]$ массива A, не меняя остальную часть массива. При $p \ge r$ участок содержит максимум один элемент и тем самым уже отсортирован. В противном случае мы отыскиваем число q_1, которое делит участок $[p-r]$ с числом элементов $n = r - p + 1$ на две примерно равные части $A[p..q]$ (содержит $[n/2]$ элементов) и $A[q+1..r]$ (содержит $[n/2]$ элементов). Здесь через $\lfloor x \rfloor$ мы обозначаем целую часть x (наибольшее целое число, меньшее или равное x), а через $\lceil x \rceil$ — наименьшее целое число, большее или равное x.

MERGE-SORT (A, p, r)
1 if $p < r$ then
2 $q \leftarrow \lfloor (p+r)/2 \rfloor$
3 MERGE-SORT (A, p, q)
4 MERGE-SORT (A, $q+1$, r)
5 MERGE (A, p, q, r).

Весь массив теперь можно отсортировать с помощью вызова MERGE-SORT (A, 1, *lenght*[A]). Если длина массива n = *lenght*[A] есть степень двойки, то в процессе сортировки произойдет слияние пар элементов в отсортированные участки длины 2, затем слияние пар таких участков в отсортированные участки длины 4 и так далее до n (на последнем шаге соединяются два отсортированных участка длины $n/2$). Этот процесс показан на рис. 8.2.

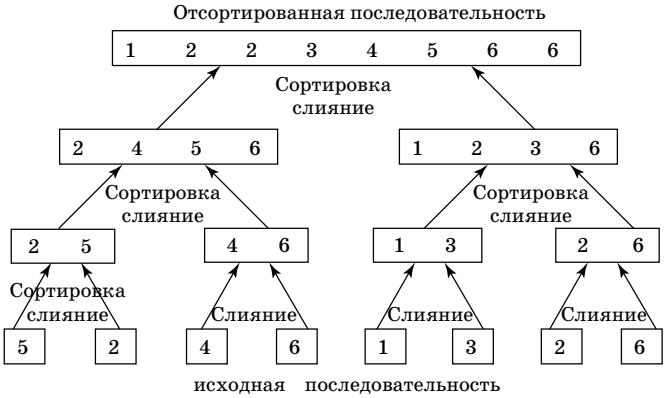


Рис. 8.2. Сортировка слиянием для массива $A = \{5, 2, 4, 6, 1, 3, 2,6\}$

Вопросы

1. Что называется алгоритмом в программировании?
2. Какой алгоритм можно считать правильным?
3. Запишите алгоритмы сортировки вставками и слиянием, объясните принципы их работы.

Задания

1. Следуя образцу (рис. 8.1), покажите, как работает INSERTION-SORT на входе $A = (31, 41, 59, 26, 58)$.
2. Измените процедуру INSERTION-SORT так, чтобы она сортировала числа в невозрастающем порядке (вместо неубывающего).
3. Рассмотрим следующую задачу поиска:
   Вход: Последовательность n чисел $A = \{a_1, a_2, ..., a_n\}$ и число ν.
   Выход: Индекс i, для которого $\nu = A[i]$, или специальное значение NIL, если ν не встречается в A. Напишите алгоритм линейного поиска, который последовательно просматривает A в поисках ν.
4. Следуя образцу (рис. 8.2), покажите работу сортировки слиянием для массива $A = (3, 41, 52, 26, 38, 57, 9, 49)$.
5. Напишите текст процедуры MERGE (A, p, q, r).

8.6. Сложность алгоритмов

Сложность описания и вычисления алгоритма, размер задачи, временная сложность, емкостная сложность, порядок сложности, анализ сложности алгоритмов сортировки

Рассматривая различные алгоритмы решения одной и той же задачи, следует проанализировать, сколько вычислительных ресурсов они требуют, и выбрать наиболее эффективный алгоритм.

Конечно, эффективность зависит от выбора вычислительной модели. В наших примерах рассматривается однопроцессорная машина с произвольным доступом (random-access machine, RAM), не предусматривающая параллельного выполнения операций.

Различают сложность самого алгоритма и сложность его описания (реализации).

> **Определение**
> ***Сложность описания алгоритма*** — это величина, характеризующая длину описания алгоритма. ***Сложность алгоритма*** (сложность вычисления алгоритма) — это функция, дающая числовую оценку длительности работы алгоритма с момента поступления на его вход начальных данных до момента получения алгоритмом выходных данных.

Для оценки сложности алгоритмов существует много критериев. Чаще всего рассматривается оценка времени полного решения задачи и рост емкости памяти при увеличении объема входных данных.

> **Определение**
> ***Размером*** задачи называется число, которое выражает меру количества входных данных. Например, размером задачи умножения матриц может быть наибольший размер матриц-сомножителей. Размером задачи на графе может быть число ребер данного графа.

Время, затрачиваемое алгоритмом, как функция размера задачи, называется ***временной сложностью*** этого алгоритма.

Объем памяти, используемый алгоритмом, как функция размера задачи, называется ***емкостной сложностью*** этого алгоритма.

При заданных ограничениях памяти и скорости компьютера сложность алгоритма определяет в итоге размер задач, которые можно решить этим алгоритмом. Рассмотрим временную сложность на примере алгоритмов сортировки.

Анализ сортировки вставками

Время сортировки вставками зависит от размера сортируемого массива: чем больше массив, тем больше требуется времени. Обычно изучается функциональная оценка роста времени работы алгоритма от роста размера входа. Впрочем, для алгоритма сортировки вставками важен не только размер массива, но и порядок его элементов: если массив почти упорядочен, то времени требуется меньше.

Как измерить размер входа? Это зависит от конкретной задачи. В одних случаях размером разумно считать число элементов на входе (как в задаче сортировки). В других — более естественно считать размером общее число битов, необходимое для представления всех входных данных. Иногда размер входа измеряется не одним числом, а несколькими (например, число вершин и число ребер графа).

Время работы алгоритма выразим через число элементарных шагов, которые выполняет этот алгоритм в ходе работы. Вопрос только в том, что считать элементарным шагом. Будем предполагать, что одна строка псевдокода требует вполне определенного фиксированного числа операций (если только это не словесное описание каких-то сложных действий). Будем различать также вызов процедуры (на который уходит фиксированное число операций) и ее исполнение, которое может быть долгим.

Вернемся к процедуре алгоритма INSERTION-SORT и отметим около каждой строки ее стоимость (число операций) и число раз, которое эта строка исполняется. Для каждого j от 2 до n (здесь $n = lenght[A]$ — размер массива) подсчитаем, сколько раз будет исполнена строка 5, и обозначим это число через t_i. Заметим, что строки внутри цикла выполняются на один раз меньше, чем проверка, поскольку последняя проверка выходит из цикла.

	Insertion-SORT (A)	стоимость	число раз
1.	for $j \leftarrow 2$ to $lenght[A]$ do	c_1	n
2.	key $\leftarrow A[j]$	c_2	$n-1$
3.	◊ добавить $A[j]$ к отсортированной части $A[1..j-1]$.	0	$n-1$
4.	$i \leftarrow j-1$	c_4	$n-1$
5.	while $i > 0$ and $A[i] >$ key do	c_5	$\sum_{j=2}^{n} t_j$
6.	$A[i+1] \leftarrow A[i]$	c_6	$\sum_{j=2}^{n} (t_j - 1)$
7.	$i \leftarrow i-1$	c_7	$\sum_{j=2}^{n} (t_j - 1)$
8.	$A[i+1] \leftarrow$ key	c_8	$n-1$.

Строка стоимостью c, повторенная m раз, дает вклад cm в общее число операций (для количества использованной памяти этого

сказать, вообще говоря, нельзя). Сложив вклады всех строк, получим выражение, определяющее временную сложность процедуры INSERTION-SORT — $T(n)$:

$$T(n) = c_1 n + c_2(n-1) + c_4(n-1) + \\ + c_5\sum_{j=2}^{n} t_i + c_6\sum_{j=2}^{n}(t_i - 1) + c_7\sum_{j=2}^{n}(t_j - 1) + c_8(n-1).$$

Время работы алгоритма зависит не только от n, но и от того, какой именно массив размера n подан на вход. Наиболее благоприятен случай, когда массив уже отсортирован. Тогда цикл в строке 5 завершается после первой же проверки (поскольку $A[i] \le \text{key}$ при $i = j - 1$), так что все t_i равны 1, и общее время есть

$$T(n) = c_1 n + c_2(n-1) + c_4(n-1) + c_5(n-1) + c_8(n-1) = \\ = (c_1 + c_2 + c_4 + c_5 + c_8)n - (c_2 + c_4 + c_5 + c_8).$$

Таким образом, в наиболее благоприятном случае время $T(n)$, необходимое для сортировки массива размера n, является линейной функцией от n, т.е. имеет вид $T(n) = an + b$ для некоторых констант a и b (эти константы определяются выбранными значениями c_1, ..., c_8).

Если же массив расположен в обратном (убывающем) порядке, то время работы процедуры будет максимальным, каждый элемент $A[j]$ придется сравнить со всеми элементами $A[1]$, ..., $A[j-1]$. При этом $t_j = j$. Вычислим суммы

$$\sum_{j=2}^{n} j = \frac{2+n}{2}(n-1) = \frac{n(n+1)}{2} - 1;$$

$$\sum_{j=2}^{n}(j-1) = \sum_{j=2}^{n-1} j = \frac{1+n-1}{2}\cdot(n-1) = \frac{n(n-1)}{2}.$$

Получаем, что в худшем случае время работы алгоритма равно

$$T(n) = c_1 n + c_2(n-1) + c_4(n-1) + c_5\left(\frac{n\cdot(n+1)}{2} - 1\right) + \\ + c_6\frac{n(n-1)}{2} + c_7\frac{n(n-1)}{2} + c_8(n-1) = \\ = \left(\frac{c_5}{2} + \frac{c_6}{2} + \frac{c_7}{2}\right)n^2 + \left(c_1 + c_2 + c_4 + \frac{c_5}{2} + \frac{c_6}{2} + \frac{c_7}{2} + c_8\right)n - \\ - (c_2 + c_4 + c_5 + c_8).$$

Теперь функция $T(n)$ — квадратичная, т.е. имеет вид $T(n) = an^2 + bn + c$, где константы a, b и c определяются значениями c_1, ..., c_8.

Этот пример показывает, что времена работы алгоритма в худшем и лучшем случаях могут сильно различаться. Для получения гарантированных рамок оценивается максимальное время работы для входов заданного размера.

В некоторых случаях нас будет интересовать также среднее время работы алгоритма на входах данной длины. Конечно, эта величина зависит от выбранного распределения вероятностей, и на практике реальное распределение входов может отличаться от предполагаемого, которое обычно считают равномерным. Иногда равномерное распределение моделируют, используя датчик случайных чисел.

Наш анализ времени работы процедуры INSERTION-SORT был основан на нескольких упрощающих предположениях. Сначала мы предположили, что время выполнения i-й строки постоянно и равно c_1. Затем мы огрубили оценку до $an^2 + bn + c$. Далее будем учитывать только порядок величин. Время работы в худшем случае имеет порядок роста n^2 (мы отбросили члены меньших порядков и коэффициент при n^2). Это записывают так: $T(n) = \Theta(n^2)$. В этом случае говорят, что временная сложность алгоритма — порядка n^2.

Алгоритм с меньшим порядком временной сложности предпочтительнее: если, скажем, один алгоритм имеет время работы $\Theta(n^2)$, а другой — $\Theta(n^3)$, то первый более эффективен, по крайней мере, для больших n.

Анализ сортировки слиянием

Время работы процедуры слияния MERGE (A, p, q, r) есть $\Theta(n)$, где n — общая длина сливаемых участков ($n = r - p + 1$). Это легко объяснить на картах. Пусть мы имеем две стопки карт (рубашкой вниз), и в каждой стопке карты идут сверху вниз в возрастающем порядке. Как сделать из них одну стопку? На каждом шаге мы берем меньшую из двух верхних карт и кладем ее (рубашкой вверх) в результирующую стопку. Когда одна из исходных стопок становится пустой, мы добавляем все оставшиеся карты второй стопки к результирующей стопке. Ясно, что каждый шаг требует ограниченного числа действий, и общее число действий есть $\Theta(n)$.

Для простоты будем предполагать, что размер n входного массива есть степень двойки (это ограничение не очень существенное). Тогда на каждом шаге сортируемый участок делится на две равные половины (см. рис. 8.2). Разбиение на части (вычисление границы) требует времени $\Theta(1)$, а слияние — времени $\Theta(n)$. Получаем соотношение

$$T(n) = \begin{cases} \Theta(1), & \text{если } n = 1, \\ 2T(n/2) + \Theta(n), & \text{если } n > 1. \end{cases}$$

Это — рекуррентное соотношение. Можно доказать, что оно влечет оценку

$$T(n) = \Theta(n \log n),$$

где через log мы обозначаем двоичный логарифм (основание логарифмов не играет роли, так как приводит лишь к изменению константы). Поэтому для больших n сортировка слиянием эффективнее сортировки вставками с временной сложностью $\Theta(n^2)$.

Вопросы

1. Чем отличается сложность алгоритма от сложности описания алгоритма?
2. Что называется размером задачи?
3. Что называется временной и емкостной сложностью алгоритма?
4. От чего зависит время работы алгоритма сортировки вставками? В каком случае оно будет максимальным, а в каком — минимальным?
5. Какой порядок сложности процедуры сортировки вставками? Проведите вычисления.
6. От чего зависит время работы алгоритма сортировки слиянием?
7. Какой порядок сложности процедуры сортировки вставками? Проведите вычисления.

Задания

1. Будем сортировать массив из n элементов так: просмотрим его и найдем минимальный элемент, который скопируем в первую ячейку другого массива. Затем просмотрим массив снова и найдем следующий элемент и т.д. Такой способ сортировки можно назвать сортировкой выбором. Запишите этот алгоритм на псевдокоде. Укажите время его работы в лучшем и худшем случаях, используя Θ-обозначения.

2*. Дана последовательность чисел $x_1, x_2, \ldots, x_n$. Покажите, что за время $\Theta(n \log n)$ можно определить, есть ли в этой последовательности два одинаковых числа.

3. Как записать выражение $n^3/1000 - 100n^2 - 100n + 3$ с помощью Θ-обозначений?

4. Почти любой алгоритм можно немного изменить, радикально уменьшив время его работы в лучшем случае. Как?

5*. Докажите по индукции, что если

$$T(n) = \begin{cases} \Theta(1), & \text{при } n = 1, \\ 2T(n/2) + \Theta(n), & \text{при } n = 2^k \text{ и } k > 1, \end{cases}$$

то $T(n) = \Theta(n \log n)$.

6. Сортировку вставками можно оформить как рекурсивную процедуру: желая отсортировать $A[1..n]$, мы (рекурсивно) сортируем $A[1..n-1]$, а затем ставим $A[n]$ на правильное место в отсортированном массиве $A[1..n-1]$. Напишите рекуррентное соотношение для времени работы такой процедуры.

7. При поиске в отсортированном массиве мы можем сначала сравнить искомый элемент со средним элементом массива и узнать, в какой половине его следует искать, а затем применить ту же идею рекурсивно. Такой способ называется двоичным поиском. Напишите соответствующую программу, используя цикл или рекурсию. Объясните, почему время ее работы есть $\Theta(\log n)$?

8. Заметим, что цикл while в строках 5–7 процедуры Insertion-Sort просматривает элементы отсортированного участка $A[1..j-1]$ подряд. Вместо этого можно было бы использовать двоичный поиск (задание 7), чтобы найти место вставки за время $\Theta(\log n)$. Удастся ли таким образом сделать общее время работы равным $\Theta(n \log n)$?

9. Сортировка вставками для коротких кусков.

   Сортировка слиянием быстрее сортировки вставками, но для малых n соотношение обратное. Поэтому имеет смысл достаточно короткие куски не разбивать дальше, а применять к ним сортировку вставками. Вопрос в том, где следует провести границу.

   9.1. Пусть массив размера n разбит на k частей размера n/k. Покажите, что можно отсортировать все части по отдельности с помощью сортировки вставками за время $\Theta(nk)$.

   9.2. Покажите, что после этого можно слить все части в один упорядоченный массив за время $\Theta(n \log(n/k))$.

   9.3. Тем самым общее время работы такого смешанного алгоритма есть $\Theta(nk + n \log(n/k))$. Какова максимальная скорость роста k как функции от n, при которой это время по-прежнему есть $\Theta(n \log n)$?

   9.4. Как бы вы стали выбирать оптимальное значение k на практике?

10. Число инверсий.

Пусть $A[1..n]$ — массив из n различных чисел. Нас будет интересовать число инверсий в этом массиве, т.е. число пар $i < j$, для которых $A[i] > A[j]$.

10.1. Укажите пять инверсий в массиве (2, 3, 8, 6, 1).

10.2. Каково максимально возможное число инверсий в массиве длины n?

10.3. Как связано время работы алгоритма сортировки вставками и число инверсий? Объясните свой ответ.

10.4. Постройте алгоритм, который считает число инверсий в массиве длины n за время $\Theta(n \log n)$. Указание: модифицируйте алгоритм сортировки слиянием.

8.7. Использование быстрых алгоритмов

Исследование зависимости возможного размера задачи от сложности алгоритма и скорости работы компьютера

Часто разница между плохим и хорошим алгоритмом более существенна, чем между быстрым и медленным компьютером. Пусть мы хотим отсортировать массив из миллиона чисел. Что быстрее — сортировать его вставками на быстром компьютере, который выполняет 100 миллионов операций в секунду, или слиянием на медленном компьютере, который выполняет 1 миллион операций в секунду? Пусть к тому же сортировка вставками написана чрезвычайно экономно и для сортировки n чисел нужно лишь $2n^2$ операций. В то же время алгоритм слиянием написан без особой заботы об эффективности и требует $50n \cdot \log n$ операций. Для сортировки вставками миллиона чисел получаем

$$\frac{2(10^6)^2 \text{ операций}}{10^8 \text{ операций в секунду}} = 20000 \text{ с} \approx 5{,}56 \text{ ч}$$

для суперкомпьютера и всего

$$\frac{50(10^6)\log(10^6) \text{ операций}}{10^6 \text{ операций в секунду}} \approx 1000 \text{ с} \approx 17 \text{ мин}$$

для второго (медленного) компьютера. Преимущество более эффективного алгоритма очевидно.

Рассмотрим еще один пример. Допустим, у нас есть пять алгоритмов A_1, ..., A_5 со следующими временными сложностями:

Алгоритм	Временная сложность — $T(n)$
A_1	n
A_2	$n \log n$
A_2	n^2
A_4	n^3
A_5	2^n

В таблице 8.1 приведены размеры задач, которые можно решить за одну секунду, одну минуту, один час каждым из этих пяти алгоритмов.

Таблица 8.1. Зависимость возможного размера задачи от сложности алгоритма

Алгоритм	Временная сложность	Максимальный размер задачи (колич. условных единиц информации), которую можно решить за		
		1 с	1 мин	1 ч
A_1	n	1000	6×10^4	$3{,}6 \times 10^6$
A_2	$n \log_2 n$	140	4893	2×10^5
A_3	n^2	31	244	1897
A_4	n^3	10	39	153
A_5	2^n	9	15	21

Пусть единица информации перерабатывается за одну миллисекунду. Тогда алгоритм A_1 может обработать за одну секунду вход размера 1000 единиц информации, в то время, как A_5 — вход размера не более 9 единиц информации.

Рассмотрим, как повлияет увеличение скорости компьютера в 10 раз на возможности использования алгоритмов A_1 ... A_5 для решения задач с временной сложностью порядка n^2. Пусть s_1 — размер некоторой задачи 1. Пусть время решения ее на первом компьютере — T_1. На втором компьютере эта задача решается в 10 раз быстрее, то есть для второго компьютера верно $T_1/10 = \Theta(s_1^2)$. Определим, какого размера задачу может решить второй компьютер за время T_1. Это означает, что необходимо найти s_2 такое, что верно $T_1 = \Theta(s_1^2)$. Исходя из равенства $T_1/10 = \Theta(s_1^2)$, выполним преобразования:

$$T_1 = 10\ \Theta(s_1^2) = (3{,}16)^2\ \Theta(s_1^2) = \Theta((3{,}16 s_1)^2).$$

Получаем, что $s_2 = 3{,}16s_1$. Это значит, что для алгоритмов с временной сложностью порядка n^2 десятикратное увеличение скорости вычислений увеличивает размер задачи, которую можно решить, только в 3 раза.

В таблице 8.2 указано увеличение максимального размера задачи, которую возможно решить при увеличении скорости машины в 10 раз.

Таблица 8.2. Зависимость возможного размера задачи от сложности алгоритма

Алгоритм	Временная сложность	Максимальный размер задачи	
		до ускорения в 10 раз	после ускорения в 10 раз
A_1	n	s_1	$10s_1$
A_2	$n \log_2 n$	s_2	примерно $10s_2$
A_3	n^2	s_3	$3{,}16s_3$
A_4	n^3	s_4	$2{,}15s_4$
A_5	2^n	s_5	$s_5 + 3{,}3$

Сравнивая приведенные таблицы, приходим к выводу, что применение более действенного алгоритма для решения задачи во многих случаях дает больший выигрыш во времени, чем применение более быстродействующей машины. Так как вычислительные машины работают все быстрее и мы можем решать все большие задачи, именно сложность алгоритма определяет то увеличение размера задачи, которое можно достичь с увеличением скорости машины. Мы видим, что разработка эффективных алгоритмов не менее важна, чем разработка быстрой электроники.

Вопросы

1. Что является более важным: эффективность алгоритма или скорость работы компьютера? Обоснуйте ответ.
2. Постройте таблицу зависимости максимально допустимого размера задачи для решения за 1 с и 1 мин для алгоритмов с временными сложностями порядка n, n^2, 2^n.
3. Приведите расчет, как влияет увеличение скорости работы компьютера в 10 раз на допустимый размер задачи для алгоритмов с временными сложностями порядка n, n^2, 2^n.

Задания

1. Пусть сортировки вставками и слиянием исполняются на одной и той же машине и требуют $8n^2$ и $64n \log n$ операций

соответственно. Для каких значений n сортировка вставками является более эффективной? Как можно улучшить алгоритм сортировки слиянием?

2. При каком наименьшем значении n алгоритм, требующий $100n^2$ операций, эффективнее алгоритма, требующего 2^n операций?
3. Сравнение времени работы. Пусть имеется алгоритм, решающий задачу размера n за $f(n)$ микросекунд. Каков максимальный размер задачи, которую он сможет решить за время t? Найти его для функций и времен, перечисленных в таблице.

$f(n)$	1 с	1 мин	1 ч	1 день	1 год	1 век
$\log n$						
$n \log n$						
n						
n^2						
n^3						
2^n						
$n!$						

раздел 9

Автоматы

9.1. Общая характеристика автоматов

Автомат, схема автомата, односторонний автомат, только читающий автомат, читающий и пишущий автомат, память автомата, такт работы автомата, управляющее устройство, начальная и заключительная конфигурация, детерминированный и недетерминированный автоматы

Вычисление по алгоритму можно рассматривать как некоторый процесс, который описывается своим множеством состояний, начальным состоянием и правилами перехода из состояния в состояние. Эти переходы могут выполняться в зависимости от внешних воздействий (входных данных), под влиянием некоторого случайного механизма или как-то иначе, например, по выбору некоторого «одушевленного существа». Процесс, в котором переходы выполняются под влиянием внешних воздействий, моделируется с помощью автомата. В сущности, автомат — это схематизированный алгоритм. Используя автоматы, можно моделировать многие машины, включая компоненты компьютера.

Когда определена задача, которую необходимо решить, возникают три вопроса. Первый — возможно ли решение этой задачи. Второй — каким образом возможно решить эту задачу. Третий — какова сложность решения этой задачи. Используя такие вычислительные модели, как автоматы, можно исследовать вопросы разрешимости и сложности различных задач. Как и грамматики, автоматы также используются при исследовании грамматической структуры языков.

Автомат можно представить в виде схемы (рис. 9.1).

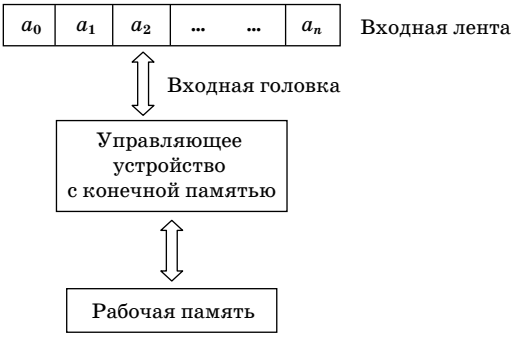


Рис. 9.1. Схема автомата

ОСНОВНЫЕ ПОНЯТИЯ. Автомат состоит из трех частей — ***входной ленты***, ***управляющего устройства*** с конечной памятью и ***вспомогательной,*** или ***рабочей, памяти.***

Входную ленту можно рассматривать как линейную последовательность клеток, или ячеек, причем каждая ячейка содержит точно один входной символ из некоторого конечного ***входного алфавита.*** Самую левую и самую правую ячейки могут занимать особые ***концевые маркеры***, обозначим их λ. Пустые ячейки обозначим ε. С помощью входной ленты представляется информация, поступающая на вход автомата.

Входная головка в каждый момент читает одну входную ячейку. За один шаг работы автомата входная головка может сдвинуться на одну ячейку влево, остаться неподвижной или сдвинуться на одну ячейку вправо. Автомат, который никогда не передвигает свою входную головку влево, называется ***односторонним.*** Схема входной ленты и входной головки представлена на рис. 9.2.

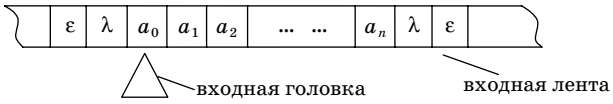


Рис. 9.2. Входная лента и входная головка

Входная головка может быть ***только читающей***, если в ходе работы автомат не создает строку — результат. Существуют такие

автоматы, входная головка которых ***читающая*** и ***пишущая***. Такие автоматы могут записывать результирующую строку или изменять символы на входной ленте.

Память автомата — это структура, в которой записываются, хранятся и считываются дополнительные данные, используемые автоматом при работе. Для каждого вида автоматов строго определен тип памяти, функции доступа к памяти и преобразования памяти. Автомат может не иметь памяти. Тип памяти часто определяет название автомата.

Работа автомата состоит из последовательности ***тактов***. Каждый такт состоит из следующих действий:

1. Читается текущий входной символ.
2. С помощью функции доступа исследуется память, и в нее помещается некоторая информация.
3. Изменяется состояние управляющего устройства.
4. Записывается выходная информация в ячейки входной ленты.
5. Входная головка сдвигается на одну ячейку влево, вправо или сохраняется в исходном положении.

Текущий входной символ и информация, извлеченная из памяти, вместе с текущим состоянием управляющего устройства определяют, каким должен быть этот такт.

Таким образом, на каждом такте определяется текущая ***конфигурация*** автомата, в которую входят:

- текущее состояние управляющего устройства;
- текущее содержимое входной ленты;
- текущее содержимое рабочей памяти.

Конфигурация автомата изменяется на каждом такте под действием управляющего устройства

Управляющее устройство состоит из конечного множества состояний $S = \{s_0 \ldots s_n\}$ и отображений, которые в зависимости от предыдущей конфигурации позволяют определить новую конфигурацию автомата, направление сдвига входной головки (если она не односторонняя), информацию на печати (если в автомате предусмотрена функция печати), информацию, помещаемую в память и извлекаемую из памяти (если автомат имеет рабочую память).

Автомат начинает работу с ***начальной*** конфигурации. При начальной конфигурации управляющее устройство находится в заданном начальном состоянии, входная головка обозревает

самый левый символ на входной ленте, и память имеет заранее установленное начальное содержимое. Конфигурация, при которой автомат прекращает работу, называется ***заключительной***. Автомат прекращает работу, перейдя в одно из состояний заранее выделенного множества заключительных состояний, или тогда, когда просмотр входных данных закончен. Условия прекращения работы четко определяются для каждого конкретного автомата.

Управляющее устройство является ***недетерминированным***, если для каждой конфигурации существует конечное множество возможных следующих конфигураций (более одной), в любую из которых автомат может перейти на следующем шаге. Управляющее устройство называется ***детерминированным***, если для каждой конфигурации существует не более одной возможной следующей конфигурации. Соответственно автомат называется детерминированным и недетерминированным в зависимости от вида управляющего устройства.

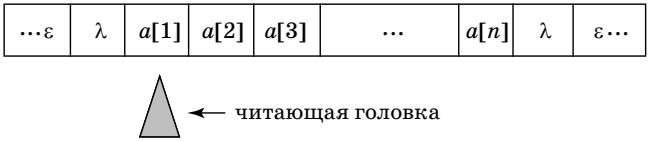
Недетерминированный автомат рассматривается как множество параллельно работающих детерминированных его экземпляров. Например, если для данной конфигурации существует три возможных следующих конфигурации, то рассматривается три параллельно работающих детерминированных автомата. Недетерминизм конечного автомата не следует смешивать со «случайностью», при которой автомат может случайно выбрать одно из следующих состояний с фиксированными вероятностями, но сам автомат всегда имеется только в одном экземпляре.

Вопросы

1. Каким образом связаны понятия атомата и алгоритма?
2. Для исследования каких вопросов испрользуются автоматы?
3. Изобразите схему автомата и назовите составные части автомата.
4. Объясните принцип действия и построение каждой части автомата.
5. Что включает в себя такт работы автомата?
6. Определите понятие конфигурации автомата. Что такое начальная и конечная конфигурации автомата?
7. Чем отличаются детерминированный и недетерминированный конечные автоматы?

Задания

1. Определите такты работы автомата, решающего задачу подсчета суммы элементов массива, схема автомата представлена на рисунке (предполагаем, что сложение является элементарной операцией для этого автомата).



Управляющее устройство			
Состояние		Входная информация (I)	
		Число	λ
▷	s_0	$S = 0, H, s_1$	$S = 0, H, s_3$
	s_1	$S = S + I, R, s_1$	H, s_2
	s_2	stop	

Рабочая память
S

В таблице управляющего устройства указаны действия, производимые автоматом, и состояние, в которое автомат переходит в зависимости от текущего состояния и значения входной информации.

Обозначения:

▷ — указатель текущего состояния;

H — читающая головка остается на месте;

R — читающая головка сдвигается вправо;

I — информация, прочитанная на данном шаге.

2. Составьте схемы автоматов по примеру схемы в задании 1, выполняющего действия:

а) определение длины входной строки;

б) подсчет количества нулей во входной строке.

9.2. Распознаватели

Распознаватель, допускаемая строка, определяемый распознавателем язык

Рассматривая грамматики, мы определили два способа задания языков: распознаванием строк и порождением строк. Для грамматик в основном применяется второй способ. Автоматы, используемые для определения языка, чаще являются распознающими, их называют ***распознавателями***. С помощью распознавателей, также как и с помощью грамматик, можно однозначно определить язык.

Обычно предполагается, что входная головка распознавателя только читает.

Рассмотрим некоторую строку ω. Распознаватель ***допускает*** входную строку ω, если, начиная с начальной конфигурации, в которой строка ω записана на входной ленте, распознаватель может проделать последовательность шагов, заканчивающуюся заключительной конфигурацией. Следует указать, что, начиная с данной начальной конфигурации, недетерминированный распознаватель может проделать много различных последовательностей шагов. Если хотя бы одна из этих последовательностей заканчивается заключительной конфигурацией, то начальная входная цепочка будет допущена.

Язык, определяемый распознавателем, представляет собой множество входных цепочек, которые он допускает.

Для каждого класса грамматик из иерархии Хомского существует класс распознавателей, определяющий тот же класс языков. Этими распознавателями являются конечные автоматы, автоматы с магазинной памятью, линейно ограниченные автоматы и машины Тьюринга. Точнее языки из иерархии Хомского можно охарактеризовать так:

1. Язык регулярный (праволинейный или леволинейный) тогда и только тогда, когда он определяется конечным автоматом.
2. Язык контекстно-свободный тогда и только тогда, когда он определяется автоматом с магазинной памятью.
3. Язык контекстно-зависимый тогда и только тогда, когда он определяется линейно ограниченным автоматом.
4. Язык определяется грамматикой общего вида тогда и только тогда, когда он определяется машиной Тьюринга.

Все указанные четыре типа автоматов описываются в следующих разделах.

Вопросы

1. Какую задачу решают автоматы, называемые распознавателями?
2. Каким образом определяется, что автомат допускает входную строку?
3. Что такое язык, определяемый распознавателем?
4. Как связаны различные виды распознавателей и грамматик?

9.3. Конечные автоматы

Конечный автомат, автомат с выходом и без выхода, задание автомата таблицей и диаграммой состояний, автоматы Мили и Мура, соответствие конечного распознавателя регулярной грамматике и регулярному выражению

Конечный автомат состоит только из входной ленты и управляющего устройства с конечной памятью (рис. 9.3). Число его состояний конечно, что и определяет его название. Управляющее устройство может быть детерминированным или недетерминированным, но входная головка — односторонняя — сдвигающаяся вправо.

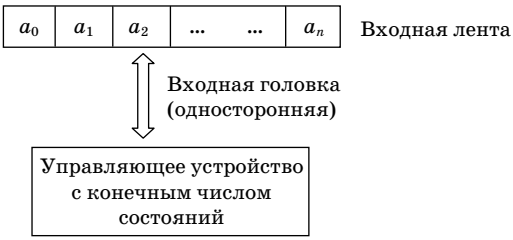


Рис. 9.3. Схема конечного автомата

Мы определим конечный автомат, задав конечное множество его состояний, допустимые входные символы, допустимые выходные символы, начальное состояние и множество заключительных состояний. Задаются также функция переходов состояний, которая по данному текущему состоянию и текущему входному символу указывает все возможные следующие состояния, и функция печати.

Определение

Конечный автомат — это структура вида

$$M = (S, I, O, f, g, s_0, F),$$

где S — конечное множество ***состояний***;

I — конечное множество допустимых ***входных символов***;

O — конечное множество допустимых ***выходных символов***;

f — отображение множества $S \times I$ в множество S, определяющее выбор следующего состояния (функцию f называют ***функцией переходов***);

g — отображение множества $S \times I$ в множество O, определяющее выходной символ (функцию g называют ***функцией выходов***);

$s_0 \in S$ — ***начальное состояние*** управляющего устройства;

$F \subseteq S$ — множество ***заключительных состояний***.

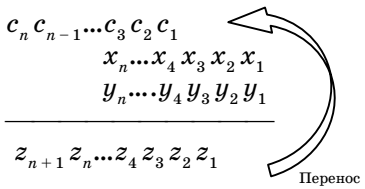
Автомат может не иметь функции g и множества O, если не предусмотрена выдача выходной информации. В этом случае считается, что автомат ***без выхода***. Если автомат содержит функцию g, тогда считается, что он ***с выходом***. Множество F может быть не определено.

Конечный автомат можно задать с помощью ***таблицы***. В этой таблице для каждой комбинации состояния автомата и входного символа указывается следующее состояние и выходной символ. Кроме того, конечный автомат можно задать с помощью ориентированного графа, называемого ***диаграммой состояний***. В этом графе вершины представляют состояния автомата, дуги — переходы из одного состояния в другое. Каждая дуга помечена входным и выходным символом, соответствующим этому переходу.

Пример. Построить конечный автомат, осуществляющий сложение двух натуральных чисел, используя их бинарное представление.

Поясним процедуру поразрядного бинарного сложения. Пусть имеются два числа, записанные с помощью n-разрядного двоичного кода: $x_n \ldots x_2 x_1$ и $y_n \ldots y_2 y_1$. Сначала складываются x_1 и y_1, в результате чего получается результат z_1 и перенос c_1. После этого производится сложение x_2, y_2 и c_1, в результате получается z_2 и перенос c_2. Эта процедура продолжается до шага n, на котором

при сложении x_n, y_n и c_{n-1} получается z_n и c_n. Последняя сумма z_{n+1} равна переносу c_n.



Конечный автомат, выполняющий это сложение, можно построить, используя только два состояния.

Рассматриваемый автомат $M = (S, I, O, f, g, s_0)$ имеет множество состояний $S = \{s_0, s_1\}$, множество входных символов $I = \{00, 01, 10, 11\}$, множество выходных символов $O = \{0, 1\}$, начальное состояние — $s_0 \in S$. Будем считать, что автомат прекращает работу, когда закончилось поступление входных данных, это означает, что сложение завершено.

Входными данными на одном такте являются два бита — соответствующие биты слагаемых, поэтому возможные варианты входной информации для одного такта — 00, 01, 10, 11. Переходы построены в соответствии с получаемой суммой, которая представлена выходным битом, и переносом, который представлен состоянием (s_0 — перенос равен 0, s_1 — перенос равен 1).

Отображения f: $S \times I \to S$ (функция переходов) и g: $S \times I \to O$ (функция выходов) представлены в таблице 9.1. В таблице показано, что если складываемые биты — 00 и текущее состояние s_0 (переноса нет), то автомат остается в состоянии s_0 (переноса нет) и выдает результат 0. Если складываемые биты — 00 и текущее состояние s_1 (перенос — 1), то автомат остается в состоянии s_0 (переноса нет) и выдает результат 1. Если, например, складываемые биты — 10 и текущее состояние s_1 (перенос — 1), то автомат остается в состоянии s_1 (перенос — 1) и выдает результат 0.

Таблица 9.1. Функции переходов f и выходов g конечного автомата для сложения

Состояния S	f					g			
	I:	00	01	10	11	00	01	10	11
s_0		s_0	s_0	s_0	s_1	0	1	1	0
s_1		s_0	s_1	s_1	s_1	1	0	0	1

Диаграмма состояний, соответствующая этому автомату представлена на рис. 9.4.

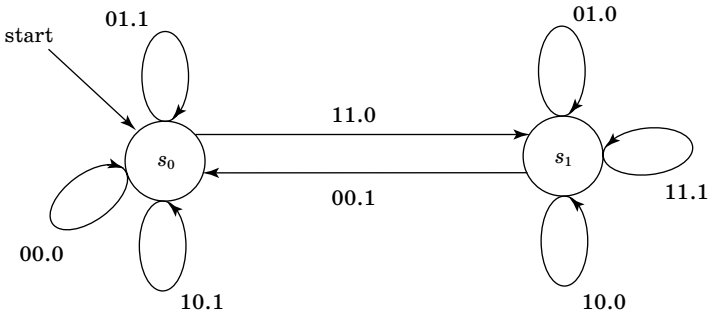


Рис. 9.4. Конечный автомат для сложения

Около каждой стрелки перехода указана входная информация и выходной сигнал. Например, запись 01.1 у петли в состоянии s_0 означает, что если на вход автомата с состоянием s_0 поступает пара битов 01, то на выход необходимо выдать 1 и в качестве следующего состояния сохранить s_0. Рассмотрим работу автомата. Если, например, текущее состояние — s_1 и на вход поступает 01, следующее состояние будет s_1, а выход — 0. Происходит сложение $0 + 1 + 1 = (10)_2$. Поэтому выходной сигнал — 0, а перенос 1 определяет следующее состояние s_1.

В рассмотренном нами автомате выходная информация соответствует переходам, однако существуют автоматы, в которых выходная информация зависит только от состояния, в которое переходит автомат. В зависимости от этого различают два вида конечных автоматов.

> **Определение**
> Конечный автомат, в котором выходная информация поставлена в соответствие переходам, называется ***атоматом Мили***. Конечный автомат, в котором выходная информация поставлена в соответствие состояниям, называется ***атоматом Мура***.

Определение означает, что в автоматах Мили функция выходов — это зависимость выходной информации от текущего состояния и текущей входной информации, а в автоматах Мура функция

выходов — это зависимость выходной информации только от текущего состояния.

Конечный автомат для сложения из нашего примера является автоматом Мили. Расмотрим пример автомата Мура.

Пример. Построим автомат Мура, который на выходе печатает «1», если количество единиц, прочитанных автоматом во входной строке, делится на 3, и печатает «0» в противном случае (автомат также печатает 0, если не прочитано ни одной единицы в начальном состоянии s_0). Входной алфавит автомата $I = \{0, 1\}$. Назовем этот автомат A_3.

Определим состояния автомата: s_0 — начальное состояние, s_1 — прочитано число единиц, дающее при делении на 3 в остатке 1, s_2 — прочитано число единиц, дающее при делении на 3 в остатке 2, s_3 — прочитано число единиц, кратное 3. Очевидно, что выходную информацию в таком автомате можно поставить в соответствие состояниям. Построим таблицу переходов и граф автомата (таблица 9.2, рис. 9.5).

Таблица 9.2. Функции переходов f и выходов g автомата Мура A_3

S \ I	f		g
	0	1	
s_0	s_0	s_1	0
s_1	s_1	s_2	0
s_2	s_2	s_3	0
s_3	s_3	s_1	1

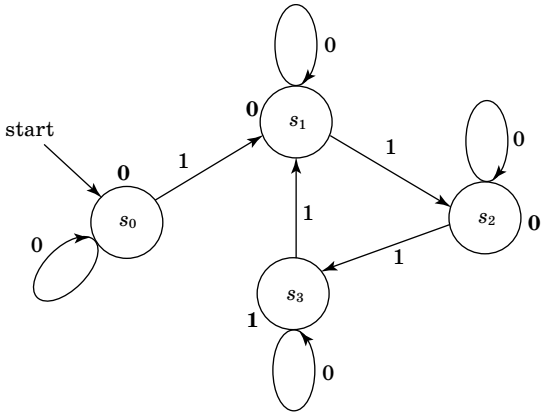


Рис. 9.5. Автомат Мура A_3

Допустим, на вход автомата поступила строка «0001010100». Рассмотрим, какую выходную информацию выдаст автомат. Построим соответствующую таблицу (таблица 9.3).

Таблица 9.3. Обработка входной информации «0001010100» автоматом A_3

Прочитанная строка	f	g
0	s_0	0
00	s_0	0
000	s_0	0
0001	s_1	0
00010	s_1	0
000101	s_2	0
0001010	s_2	0
00010101	s_3	1
000101010	s_3	1
0001010100	s_3	1

После прочтения строки «0001010100» автомат печатает «1», значит количество единиц, прочтенных автоматом во входной строке, делится на 3.

С помощью конечных автоматов можно распознавать строки языков. Автомат A_3, построенный в предыдущем примере, можно рассматривать как распознаватель языка, состоящего из строк бит, содержащих число единиц, кратное 3. Назовем этот язык L_3. Языку L_3 принадлежат, например, строки «111», «01000110001010100», «0101010101010101010» и др. Для распознавания языка обычно не требуется наличие у автомата выходных сигналов, но определяется множество конечных состояний автомата. Если прочтение входной строки автомат завершил, находясь в конечном состоянии, то строка считается распознанной. В нашем примере множество конечных состояний состоит из единственного состояния s_3.

Утверждение. Язык является регулярным тогда и только тогда, когда он распознается с помощью конечного автомата.

Поскольку существует конечный автомат, распознающий язык L_3, то следовательно, L_3 — регулярный язык, и его можно задать с помощью регулярного выражения и регулярной грамматики. Составим соответствующее регулярное выражение. В состоянии s_0

автомат находится, пропуская любое количество нулей, в начале входной строки, это соответствует регулярному выражению 0^*. Состояние s_1 допускает одну единицу (при переходе из s_0 в s_1), затем может следовать любое количество нулей, автомат остается в состоянии s_1. Таким образом, состояние s_1 соответствует присоединению выражения 10^*. Аналогично переходу от s_1 к состоянию s_2, от s_2 — к s_3 отвечает двукратное присоединение строки вида 10^*. Состояние s_3 является конечным. Составим выражение с помощью операции конкатенации: $0^*10^*10^*10^*$. Поскольку из состояния s_3 автомат может опять перейти в состояние s_1, прочтя единицу, и затем также последовательно пройти следующие состояния сколь угодно раз от s_1 до s_3, то в виде регулярного выражения это выглядит так: $(10^*10^*10^*)^*$. Таким образом, регулярное выражениye для языка L_3 имеет вид:

$$L_3 : 0^*10^*10^*10^*(10^*10^*10^*)^*.$$

Можно построить грамматику для языка L_3. Это есть грамматика

$$G_3 = (\{A, B, C, S\}, \{0, 1\}, P, S),$$

где P состоит из набора продукций:

1. $S \to AB$.
2. $A \to C1C1C1C$.
3. $B \to 1C1C1CB \mid \varepsilon$.
4. $C \to 0C \mid \varepsilon$.

Действительно, продукция 4 соответствует выражению 0^*, тогда продукция 3 соответствует выражению $(10^*10^*10^*)^*$, а продукция 2 — выражению $0^*10^*10^*10^*$.

Вопросы

1. Изобразите схему конечного автомата и опишите его устройство.
2. Дайте определение конечного автомата, его составляющих.
3. Опишите два способа задания конечного автомата.
4. Чем отличаются конечные автоматы с выходом и без выхода.
5. Дайте определение автоматов Мили и Мура.
6. Каким образом функционируют конечные распознаватели?
7. Какой вид языков определяется конечными автоматами? С помощью каких формальных систем, кроме конечных автоматов, можно задать этот вид языков?

Задания

1. Постройте диаграммы для конечных автоматов, представленных следующими таблицами:

а)

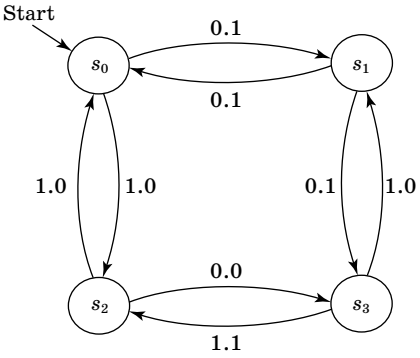
S	f	g
	0 1	0 1
s_0	s_1 s_0	0 1
s_1	s_0 s_2	0 1
s_2	s_1 s_1	0 0

б)

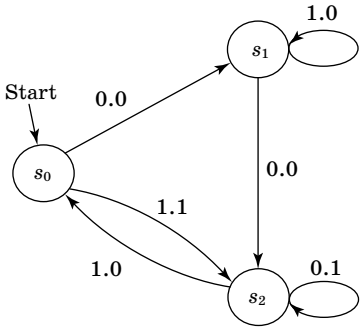
S	f	g
	0 1	0 1
s_0	s_1 s_0	0 0
s_1	s_0 s_2	1 1
s_2	s_1 s_1	0 1
s_3	s_1 s_2	1 0

2. Постройте таблицы состояний для автоматов, заданных диаграммами:

а)



б)



3. Постройте конечный автомат, который изменяет каждый четный бит в исходной строке, начиная со второго, а нечетные биты, начиная с первого, оставляет неизменными.
4. Постройте конечный автомат, моделирующий работу замка для сейфа, содержащего числа от 1 до 40, открывающегося комбинацией: поворот вправо на 10, влево на 8, вправо на 37. Входные данные состоят из пар: направление поворота (L, R), число

от 1 до 40. Если считать этот автомат распознавателем, то какой язык он определяет?

5. Постройте конечный автомат, который определяет, является ли последними символами входных данных слово «автомат». Входные данные — строка из букв русского алфавита. Постройте соответствующее регулярное выражение.
6. Исследуйте, каким образом можно осуществить построение конечного автомата, соответствующего заданному регулярному выражению, и наоборот.

9.4. Автоматы с магазинной памятью

Автомат с магазинной памятью, распознавание строк автоматом, определяемый автоматом язык

Автомат с магазинной памятью (МП-автомат) — это конечный автомат, имеющий устройство памяти типа стэк. В стэке элементы информации помещаются в самую верхнюю ячейку памяти, сдвигая тем самым все ее содержимое на одну ячейку вниз, и извлекаются только из самой верхней ячейки памяти, сдвигая тем самым все ее содержимое на одну ячейку вверх. При этом в каждый момент времени доступен только самый верхний элемент памяти. Теоретически, объем памяти МП-автомата не ограничен. Схема МП-автомата представлена на рисунке 9.6.

Рис. 9.6. Автомат с магазинной памятью

Определение

Автомат с магазинной памятью — это структура

$$M = (S, I, O, Z, f, g, s_0, z_0, F),$$

где S — конечное множество ***состояний***;

I — конечное множество допустимых ***входных символов***;

O — конечное множество допустимых ***выходных символов***;

Z — конечное множество допустимых ***символов памяти***;

f — ***функция переходов*** — отображение множества $S \times I \times Z$ в множество $S \times Z$, определяющее выбор следующего состояния и помещаемый в память символ;

g — ***функция выходов*** — отображение множества $S \times I \times Z$ в множество O, определяющее выходной символ;

$s_0 \in S$ — ***начальное состояние*** управляющего устройства;

$z_0 \in Z$ — ***начальный символ*** — символ, находящийся в памяти (стэк) в начальный момент;

$F \subseteq S$ — множество ***заключительных состояний***.

Пример. Работу автомата с магазинной памятью рассмотрим на примере МП-распознавателя, допускающего множество $\{0^n1^n \mid n = 0, 1, 2, ...\}$. Запись 0^n1^n означает множество строк, состоящих из равного количества нулей и единиц, например «000111», «01», «0000011111». В п. 7.5 этот язык был обозначен через L_{nn}. Он является контекстно свободным, но не является регулярным.

Рассматриваемый автомат $M = (S, I, Z, f, s_0, z_0, F)$ имеет множество состояний $S = \{s_0, s_1, s_2\}$, множество входных символов $I = \{0, 1\}$, множество символов памяти $Z = \{0, 1\}$, начальное состояние s_0, заключительное состояние s_0, изначально в память записан символ $x = z_0$. Будем считать, что автомат допускает входную строку $a_1, a_2, ..., a_n \in I^*$, если после прочтения этой строки (то есть последовательного обозрения головкой всех символов a_i строки), автомат переходит в заключительное состояние.

Функция переходов f задается следующими значениями:

$$f(s_0, 0, x) = \{(s_1, 0x)\};$$
$$f(s_1, 0, 0) = \{(s_1, 00)\};$$
$$f(s_1, 1, 0) = \{(s_2, \varepsilon)\};$$

$$f(s_2, 1, 0) = \{(s_2, \varepsilon)\};$$
$$f(s_2, \varepsilon, x) = \{(s_0, \varepsilon)\}.$$

Строка ε — пустая строка. Переход $f(s_0, 0, x) = \{(s_1, 0x)\}$ означает, что если автомат находится в состоянии s_0, получает на вход 0 и извлекает из памяти текущий символ x, то он должен перейти в состояние s_1, вернуть в память символ x и поместить в стэк символ 0. Переход $f(s_2, 1, 0) = \{(s_2, \varepsilon)\}$ означает, что если автомат находится в состоянии s_2, получает на вход символ 1 и извлекает из памяти текущий символ 0, то он должен остаться в состоянии s_2 и в память ничего не добавлять. Таким образом, получается, что на этом такте из стэка памяти «вытолкнут» и потерян символ 0. Конечная конфигурация автомата — (s_0, ε). При работе автомат считывает в память первую пловину цепочки, состоящую из нулей, а затем удаляет из памяти по одному нулю на каждую единицу, поступающую на вход. Кроме того, переходы состояний гарантируют, что все нули предшествуют единицам. Например, для входной цепочки 0011 автомат проделает такую последовательность тактов:

$$(s_0, 0011, x) \to (s_1, 011, 0x) \to (s_1, 11, 00x) \to$$
$$\to (s_2, 1, 0x) \to (s_2, \varepsilon, x) \to (s_0, \varepsilon).$$

Символ «→« означает переход от такта к такту. Для каждого такта записана конфигурация, которая включает текущее состояние, оставшуюся непрочитанной часть входной цепочки, содержимое памяти. Например, для конфигурации $(s_1, 011, 0x)$ текущее состояние — s_1, непрочитанная часть входной цепочки — 011, текущий входной символ (расположен слева) — 0. Содержимое памяти — $0x$, извлекаемый символ (расположен слева) — 0.

С помощью автоматов с магазинной памятью можно распознавать строки языка двумя способами. Первый — строка считается распознанной, если после ее прочтения автомат переходит в заключительное состояние. Второй — если после прочтения строки стэк памяти оказывается пустым. В рассмотренном нами примере требовалось выполнение обоих условий.

Утверждение. Язык является контекстно-свободным тогда и только тогда, когда он распознается автоматом с магазинной памятью.

Вопросы

1. Изобразите схему автомата с магазинной памятью. Объясните устройство блока памяти МП-автомата.

2. Дайте определение автомата с магазинной памятью.
3. Какими способами можно распознавать строки с помощью МП-автомата?
4. Какой вид языков можно распознавать с помощью МП-автомата?

Задания

1. Убедитесь, что для языка L_{nn}: 0^n1^n, $n \geq 0$ невозможно построить конечный распознаватель. Постройте конечный распознаватель для этого языка при ограничении $0 \leq n \leq 10$.
2. Произведите распознавание строк «000011» «111000» с помощью МП-распознавателя для языка L_{nn} (этот распознаватель рассматривался в качестве примера в данной главе). Убедитесь, что эти строки не могут быть распознаны.
3. Постройте автомат с магазинной памятью, который распознает язык, состоящий из строк $\omega\omega^n$, где ω определяется регулярным выражением $(a \cup b)(a \cup b)^*$.
4. Определите, какой язык задают грамматики. Постройте автомат с магазинной памятью, который распознает этот язык.

   а)

   $G = (\{A, S\}, \{a, b, d\}, P, S)$,

   P:

   1. $S \to Ad$
   2. $A \to aAbb \mid abb$.

   б)

   $G = (\{A, S\}, \{a, b, d\}, P, S)$,

   P:

   1. $S \to SS \mid A$
   2. $A \to 0A1 \mid 01$.
5. Напишите алгоритм, моделирующий работу МП-автомата.

9.5. Машина Тьюринга. Линейно-ограниченные автоматы

Машина Тьюринга, тезис Черча — Тьюринга, распознавание строк, распознаваемые языки, линейно-ограниченные автоматы

Машина Тьюринга является наиболее мощным автоматом из всех рассмотренных нами. Названа она в честь английского математика Алана Тьюринга. Машина Тьюринга содержит все элементы конечного автомата, кроме того, ее входная лента

бесконечна в обоих направлениях, считывающая головка является также и печатающей, она может передвигаться вдоль ленты в обоих направлениях.

Утверждение. С помощью машины Тьюринга можно моделировать любой вычислительный процесс, который можно назвать алгоритмом. Это утверждение известно как ***тезис Черча — Тьюринга***. С помощью машины Тьюринга можно распознавать любые языки, определяемые грамматиками общего вида.

Схема машины Тьюринга представлена на рисунке 9.7.

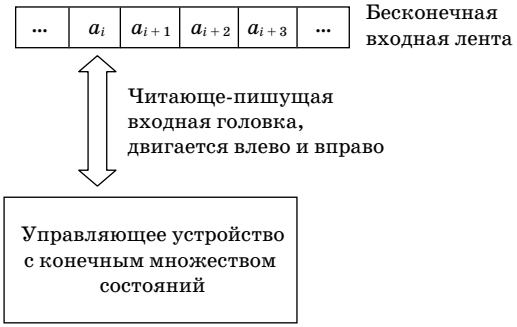


Рис. 9.7. Схема машины Тьюринга

Определение

Машина Тьюринга — это структура

$$T = (S, I, f, s_0),$$

где S — конечное множество состояний;

I — конечное множество допустимых символов ленты, включает пустой символ λ;

f — отображение множества $S \times I$ в множество $S \times I \times \{L, R\}$, определяющее для комбинации состояния и входного символа выбор следующего состояния, печатаемый символ, направление перемещения головки — L (влево), R (вправо).

Наконец, $s_0 \in S$ — начальное состояние управляющего устройства.

Поскольку лента бесконечна, то информация, которая должна быть прочитана, занимает лишь часть ленты. Пустые ячейки занимает символ λ. В начальном состоянии читающе-пишущая головка

устанавливается на самую левую из непустых ячеек, и управляющее устройство находится в состоянии s_0. Машина работает, совершая действия, определяемые функцией f. Машина останавливается, если для текущей комбинации состояния и входного символа функция f не определена.

Может быть задано множество конечных состояний. Конечные состояния определяются, в основном, когда машина моделирует распознавание строк.

Пример. Построим машину Тьюринга, распознающую строки вида 0^n1^n, $n \geq 1$. Множество этих строк образует контекстно-свободный язык L_{nn}.

Положим $T = (S, I, f, s_0)$,

где

$S = \{s_0, ..., s_6\}$,

$I = \{0, 1, M\}$,

s_0 — начальное состояние;

s_6 — конечное состояние.

Функция переходов f представлена в таблице 9.4. В каждой ячейке содержится номер следующего состояния, печатаемый символ и направление перемещения головки (L — влево, R — вправо).

Таблица 9.4. Функция переходов f

	Входная информация			
	0	1	M	λ
s_0	$s_1\ M\ R$	-	-	-
s_1	$s_1\ 0\ R$	$s_1\ 1\ R$	$s_2\ M\ L$	$s_2\ \lambda\ L$
s_2	-	$s_3\ M\ L$	-	-
s_3	$s_4\ 0\ L$	$s_3\ 1\ L$	$s_5\ M\ R$	-
s_4	$s_4\ 0\ L$	-	$s_0\ M\ R$	-
s_5	-	-	$s_6\ M\ R$	-
s_6	-	-	-	-

Входная строка считается распознанной, если после ее прочтения машина переходит в конечное состояние s_6. При распознавании используется вспомогательный символ M, который записывается вместо уже прочитанных символов строки.

Рассмотрим последовательность шагов данной машины Тьюринга при распознавании строки «000111» = $0^3 1^3$.

На первом шаге машина T считывает первый ноль, заменяет его на M — «M00111», переходит в состояние s_1, передвигается

вправо, текущим становится второй 0. На втором шаге машина считывает текущий ноль, не изменяет его, остается в состоянии s_1, передвигается вправо, текущим становится третий 0, и т.д. Представим эти действия схематично следующим образом:

$$(s_0) \ldots \lambda\underline{0}00111\lambda \ldots \to (s_1) \ldots \lambda M\underline{0}0111\lambda \ldots \to$$
$$\to (s_1) \ldots \lambda M0\underline{0}111\lambda \ldots \to (s_1) \ldots \lambda M00\underline{1}11\lambda \ldots \to$$
$$\to (s_1) \ldots \lambda M001\underline{1}1\lambda \ldots \to (s_1) \ldots \lambda M0011\underline{1}\lambda \ldots \to$$
$$\to (s_1) \ldots \lambda M00111\underline{\lambda} \ldots \to (s_1) \ldots \lambda M00111\underline{\lambda} \ldots \to$$
$$\to (s_2) \ldots \lambda M0011\underline{1}\lambda \ldots \to (s_3) \ldots \lambda M001\underline{1}M\lambda \ldots \to$$
$$\to (s_3) \ldots \lambda M00\underline{1}1M\lambda \ldots \to (s_3) \ldots \lambda M0\underline{0}11M\lambda \ldots \to$$
$$\to (s_4) \ldots \lambda M\underline{0}011M\lambda \ldots \to (s_4) \ldots \lambda \underline{M}0011M\lambda \ldots \to$$
$$\to (s_0) \ldots \lambda M\underline{0}011M\lambda \ldots \to (s_1) \ldots \lambda MM\underline{0}11M\lambda \ldots \to$$
$$\to (s_1) \ldots \lambda MM0\underline{1}1M\lambda \ldots \to (s_1) \ldots \lambda MM01\underline{1}M\lambda \ldots \to$$
$$\to (s_1) \ldots \lambda MM011\underline{M}\lambda \ldots \to (s_2) \ldots \lambda MM01\underline{1}M\lambda \ldots \to$$
$$\to (s_3) \ldots \lambda MM0\underline{1}MM\lambda \ldots \to (s_3) \ldots \lambda MM\underline{0}1MM\lambda \ldots \to$$
$$\to (s_4) \ldots \lambda M\underline{M}01MM\lambda \ldots \to (s_0) \ldots \lambda MM\underline{0}1MM\lambda \ldots \to$$
$$\to (s_1) \ldots \lambda MMM\underline{1}MM\lambda \ldots \to (s_1) \ldots \lambda MMM1\underline{M}M\lambda \ldots \to$$
$$\to (s_2) \ldots \lambda MMM\underline{1}MM\lambda \ldots \to (s_3) \ldots \lambda MM\underline{M}MMM\lambda \ldots \to$$
$$\to (s_5) \ldots \lambda MMM\underline{M}MM\lambda \ldots \to (s_6) \ldots \lambda MMMM\underline{M}M\lambda \ldots \to$$
$$\to \text{останов (переход не определен).}$$

Поскольку машина закончила работу в состоянии s_6, то это означает, что строка $0^3 1^3$ распознана. Итак, контекстно-свободный язык L_{nn} может быть задан с помощью автомата с магазинной памятью. Рассмотрим язык $0^n 1^n 2^n$ (назовем его L_{nnn}). Язык L_{nnn} является контекстно-зависимым и требует более сложный автомат для распознавания. Построим машину Тьюринга, распознающую этот язык.

Пример. Машина Тьюринга, распознающая язык L_{nnn}. Положим

$$T = (S, I, f, s_0),$$

где

$S = \{s_0, \ldots, s_9\}$,
$I = \{0, 1, 2, M, E\}$,
s_0 — начальное состояние;
s_9 — конечное состояние.

Функция f представлена в таблице 9.5.

Таблица 9.5. Функция f

	Входная информация					
	λ	0	1	2	E	M
s_0	$s_9\ \lambda\ L$	$s_1\ 0\ L$	-	-	-	-
s_1	$s_2\ E\ R$	-	-	-	-	-
s_2	-	$s_3\ M\ R$	-	-	-	$s_2\ M\ R$
s_3	-	$s_3\ 0\ R$	$s_4\ M\ R$	-	-	$s_3\ M\ R$
s_4	-	-	$s_4\ 1\ R$	$s_5\ M\ R$	-	$s_4\ M\ R$
s_5	$s_6\ \lambda\ L$	-	-	$s_5\ 2\ R$	-	-
s_6	-	-	-	$s_7\ 2\ L$	-	$s_8\ M\ L$
s_7	-	$s_7\ 0\ L$	$s_7\ 1\ L$	$s_7\ 2\ L$	$s_2\ E\ R$	$s_7\ M\ L$
s_8	-	-	-	-	$s_9\ E\ L$	$s_8\ M\ L$
s_9	-	-	-	-	-	-

Машины Тьюринга могут моделировать не только распознавание строк языка, но и любые другие вычислительные алгоритмы. Очевидно, что для выполнения таких самых простых вычислений, как сложение или умножение, требуются довольно громоздкие машины Тьюринга. Несколько упрощает структуру этих машин добавление дополнительных параллельных входных и выходных лент. Но ясно, что машина Тьюринга не является оптимальным устройством для практического выполнения вычислений или распознавания языков. Основная цель их создания заключается в исследовании с их помощью вопросов сложности алгоритма и алгоритмической разрешимости. Если удается доказать, что для решения некоторой задачи невозможно построить машину Тьюринга, то это означает, что не существует и алгоритма для ее решения.

> **Определение**
> Недетерминированный распознаватель, памятью которого является первоначально пустая лента машины Тьюринга длиной не больше входной цепочки, называется ***линейно-ограниченным автоматом***.

Язык определяется линейно-ограниченным автоматом тогда и только тогда, когда он контекстно-зависимый.

Вопросы

1. Сформулируйте тэзис Черча — Тьюринга.
2. Какие языки можно распознавать с помощью машины Тьюринга?
3. Изобразите схему машины Тьюринга.
4. Дайте определение машины Тьюринга.
5. Какова основная цель создания машины Тьюринга?
6. Что представляет собой линейно-ограниченный автомат, и какие языки он определяет?

Задания

1. Исследуйте работу машины Тьюринга для распознавания языка L_{nnn}: $0^n1^n2^n$ (конструкция машины приведена в качестве примера в этом разделе). Используйте в качестве входной информации строки, принадлежащие и не принадлежащие данному языку.
2. Пусть машина Тьюринга задана таблицей.

	Входная информация		
	0	1	λ
s_0	$s_0 0\ R$	$s_1 1\ R$	$s_3 \lambda\ R$
s_1	$s_0 0\ R$	$s_2 0\ L$	$s_3 \lambda\ R$
s_2	$s_3 0\ R$	-	-
s_3	-	-	-

   Определите, какой вид будет иметь информация на ленте после останова машины, если в начале работы на вход подана строка «... λλ0̲10110λλ ...» (головка установлена на подчеркнутый символ).
3. Постройте машину Тьюринга (алфавит ленты {0, 1}), которая:
   а) изменяет первый 0 на ленте на 1 и остальные символы оставляет без изменения;
   б) изменяет все символы 1 на ленте на 0, кроме самой левой единицы;
   в) распознает строки, заканчивающиеся нулем;
   г) распознает строки, содержащие четное число единиц.
4. Составьте алгоритм, моделирующий работу машины Тьюринга. На вход алгоритма поступает таблица машины и входная строка.

раздел 10

Комбинаторика

10.1. Предисловие

Комбинаторика или комбинаторный анализ — область дискретной математики, которая изучает комбинации и перестановки предметов, взаимное расположение частей конечных множеств предметов произвольного происхождения, а также бесконечных множеств, которые удовлетворяют некоторым условиям подчиненности. Возникла комбинаторика в XVII ст. Но в самостоятельную научную дисциплину комбинаторный анализ сформировался лишь в XX ст. Комбинаторные методы применяются в теории вероятностей, случайных процессах, статистике, математическом программировании, планировании экспериментов. Рассматриваются задачи, в которых приходится избирать те или иные предметы, располагать их в определенном порядке и находить среди разнообразных комбинаций наилучшие. Комбинаторика тесно связана с теориями чисел, графов, конечных автоматов. Ее достижения используются при планировании и анализе научных экспериментов, в линейном и динамическом программировании, в математической экономике, в системах проектирования и управления, в компьютерных науках и других областях науки и техники.

Выделяют следующие проблемы комбинаторного анализа:

1) *Задачи на перечисления*, в которых необходимо определить количество размещений элементов конечного множества, удовлетворяющих определенным условиям. Для решения задач перечисления разработаны разнообразные методы, среди которых следует отметить метод производящих функций и метод перечисления Пойа.

2) *Задачи о существовании и построении*. В задачах такого класса рассматриваются вопросы: имеет ли место определенная конфигурация частей конечного множества с некоторыми свойствами; если такая конфигурация существует, то как ее построить. При этом важную роль играют численные и алгебраические методы.

3) *Задачи о выборе*. Задачи такого типа исследуют условия, при которых можно осуществить выбор подмножества или некоторой совокупности частей множества так, чтобы удовлетворить определенным требованиям. Для решения задач о выборе, кроме комбинаторных методов, надо применять алгебраический аппарат.

История развития комбинаторики

Китайские рукописи XII–XIII ст. до н. э. описывали окружающую действительность как объединение двух начал — мужского и женского. Восемь рисунков из трех рядов символов изображали землю, горы, воду и прочие стихии — сумма первых восьми натуральных чисел (36) воплощала Вселенную.

Легенда о китайском императоре Ию, который жил 4000 лет тому, рассказывает: существовала священная черепаха, на панцире которой был изображен рисунок, содержащий девять чисел (рис. 10.1).

4	9	2
3	5	7
8	1	6

Рис. 10.1. Магический квадрат

Сложив числа в строках, столбцах или диагоналях магического квадрата, изображенного на рис. 10.1, получим одно и то же число — 15.

Конкретные комбинаторные задачи, которые касались перечня предметов или небольших их групп, древние греки решали без ошибок.

Существуют смежные задачи между комбинаторикой и теорией чисел.

					0	0	0	
		0	0		0	0	0	
0		0	0		0	0	0	

Рис. 10.2. Представление квадратов чисел

Древнегреческие философы предложили понятие «квадрата числа» — квадраты натуральных чисел изображались камешками, как показано на рис. 10.2. С помощью комбинаций предметов и вычисления количества таких комбинаций были получены числа, которые равняются сумме своих делителей, например, $6 = 1 + 2 + 3$; «дружественные числа», любое из которых равняется сумме делителей некоторого другого числа.

Большое развитие в средние века в Европе и Азии получили разнообразные числовые суеверия и толкования, связанные с заменой букв соответствующими числами (греки обозначали числа с помощью букв — первые 9 литер алфавита обозначали цифры от 1 до 9, следующие за ними — от 10 до 90, а последние 9 литер — числа от 100 до 900). В средние века ученые, которые назывались кабалистами, поддавали такому «анализу» слова Библии и других священных книг и делали на основе своих исследований пророчества о будущем мира. Особо выделялись чертова дюжина — число 13, комбинации этого числа с днями недели и астрономическими явлениями — затмениями или появлением комет; число дьявола — 666.

Наряду с кабалистами и мистиками комбинаторикой в средневековье занимались астрологи.

Астрология — наука, которая изучает объединение планет и их взаиморасположение между собой, — также дала толчок для формулирования и решения классических комбинаторных задач. Астрологов интересовал вопрос о движении планет и их влиянии на судьбы людей. Особое значение уделяли они объединениям планет — встречам разных планет в одном знаке зодиака.

Астролог Бен Эзра (1140 г. н. э.) подсчитал количество сочетаний семи планет по две, по три и т.д. Оказалось

$$C_7^2 = C_7^5, \quad C_7^3 = C_7^4,$$

где C_n^k — число сочетаний из n разных предметов по k.

Окончательно формулу числа сочетаний получил Леви Бен Герман в XIV ст.:

$$C_n^k = \frac{n(n-1)\ldots(n-k-1)}{1 \cdot 2 \cdot \ldots \cdot k}.$$

В начале XVII ст. эту формулу заново вывел французский математик П. Эригон.

Большое внимание классификации видов суждений уделяла схоластическая наука. В схоластике удивительно переплетались исследования богословов с исследованиями проблем, которые

примыкают к комбинаторике, математической логике, теории множеств и других современных областей математики. Большими знатоками схоластических исследований были основатели теории множеств — *Бернард Больцано* и *Георг Кантор*. Споря о взаимоотношениях членов пресвятой троицы, соподчинении ангелов, архангелов, херувимов и серафимов, схоласты были вынуждены рассматривать разнообразные отношения порядков и иерархии. Например, загробный мир, описанный Данте в «Божественной комедии», с его кругами ада и разными частями чистилища и рая.

Схоласт *Раймонд Лулий* создал в XIII ст. машину, составленную из нескольких кругов, на которые были нанесены основные предикаты, субъекты, атрибуты и прочие понятия схоластической логики. Поворачивая эти круги, он получал разнообразные объединения понятий и надеялся найти с их помощью истину.

Решая вопросы о произведении корней любой степени, арабские алгебраисты дошли до формулы для степени суммы двух чисел, известной под названием «бином Ньютона». Наверное, эту формулу знал поэт и математик *Омар Хайям*, который жил в XI–XII ст. н. э. В XIII–XV ст. н. е. такую формулу приводят в своих работах некоторые арабские ученые.

Итальянец *Леонардо Пизанский*, по прозвищу *Фибоначчи*, в своей книге «Liber Abaci», изданной в 1202 г., систематизировал всю арифметику арабов, некоторые сведения из геометрии *Эвклида* и прибавил к ним результаты своих исследований. Работа Фибоначчи содержала и новые комбинаторные задачи, например, о нахождении наименьшего количества гирь, с помощью которых можно получить любой целый вес от 1 до 40 фунтов. Рассматривал Леонардо и нахождение целых решений уравнений. Далее аналогичные задачи привели к решению задачи о количестве натуральных решений систем уравнений и неравенств, которая может рассматриваться как одна из фундаментальных частей комбинаторики.

Но главной заслугой Леонардо перед комбинаторикой было то, что он сформулировал и решил «задачу о кроликах». Еще со времен греческих математиков были известны две последовательности, каждый член которых получался по определенным правилам из предшествующих членов — арифметическая и геометрическая прогрессии. В задачи Леонардо появилась новая последовательность, каждый член которой (начиная с третьего) равняется сумме двух предшествующих членов: $u_n = u_{n-1} + u_{n-2}$. Подобные формулы получили название рекуррентных (от латинского recurrere — возвращаться). Метод рекуррентных формул

оказался со временем одним из мощнейших для решения комбинаторных задач.

Нарды, шашки, шахматы, японские облавные шашки «го» — в этих играх нужно рассматривать разнообразные соединения передвижных фигур, и выиграет тот, кто их лучше выучит, просчитает выигрышные комбинации и избегнет проигрышных.

Немалый толчок к развитию комбинаторики дали азартные игры, которые существовали очень давно, но получили особое распространение после крестовых походов. Наибольшее распространение получила игра в кости.

Работы *Блеза Паскаля* и *Пьера Ферма* (Франция, XVII ст.) положили начало комбинаторной (дискретной) теории вероятностей.

В 1666 г. *Готфрид Вильгельм Лейбниц* предложил фундаментальные понятия комбинаторики. К области комбинаторики Лейбниц относил и «универсальную характеристику» — математику суждений, то есть прообраз математической логики.

После работ Паскаля и Ферма, Лейбница и Л. Эйлера можно было уже говорить о комбинаторике как о самостоятельной части математики, тесно связанной с другими областями науки, как теория вероятностей, учение о рядах и т.п. В конце XVIII ст. немецкий ученый Гинденбург и его ученики сделали даже попытку выстроить общую теорию комбинаторного анализа.

В XIX ст. во время исследований по комбинаторике начали прослеживаться связи этой теории с определителями, конечными геометриями, группами, математической логикой и т.д.

Одной из наиболее сложных загадок в биологии XX ст. было строение «нитей жизни» — молекул белка и нуклеиновых кислот. Оказалось, что молекулы белка — это объединения нескольких длинных цепей, составленных из 20 аминокислот. После открытия строения ДНК возник вопрос: каким образом молекулы ДНК передают организму инструкции о построении цепей аминокислот, из которых состоят белки. Исследования показали, что речь идет о 20 аминокислотах. Американский физик *Г. Гамов* сформулировал задачу так: как с помощью 4 видов нуклеотидов можно зашифровать 20 видов аминокислот? Эта задача, как и много других задач генетики, имеет комбинаторное содержание и, в частности, связана с таким разделом комбинаторики, как *кодирование*.

Криптография — наука о шифровании — была известна с давних времен; одна из областей криптографии — расшифровка письмен (текстов) древних цивилизаций. Ключ к чтению иероглифов, утерянный несколько тысячелетий назад, был снова найден именно

благодаря комбинаторным методам в чтении забытых письменностей, основанных на наблюдениях за текстом, на сопоставлении повторяемости комбинаций слов и грамматических форм. Также нужен анализ назначения надписи, времени и условий ее составления и т.д. Типичный пример из этой области дает история расшифровки клинописных надписей.

10.2. Первичные понятия комбинаторного анализа

Некоторые классические задачи комбинаторики

А. Магический квадрат. «Расположить числа 1, 2, 3, 4, 5, 6, 7, 8, 9 в виде квадрата так, чтобы сумма чисел любого из столбцов, строк и диагоналей была одной и той же».

Магический квадрат 3-го порядка был изображен на рис. 10.1. Более сложная задача — построение магических квадратов 4-го порядка; доказана возможность построения всего 880 типов таких квадратов. Существуют алгоритмы построения квадратов высших порядков и магических кубов.

Б. Шахматные задачи. Хорошо известна задача о ферзях: «Поставить на шахматную доску наибольшее количество ферзей таким образом, чтобы ни один из них не мог взять другого».

Решение здесь достаточно очевидное — больше 8 ферзей на доску поставить не удается.

Поскольку ферзь бьет по горизонтали, вертикали и диагонали шахматной доски размером 8×8 клеток, то больше 8 ферзей поставить невозможно. Задачу можно решить прямым перебором вариантов, и окажется, что восемь ферзей можно расположить таким образом, причем всего есть 92 варианта такого расположения. Один из вариантов в качестве примера приведен на рис. 10.3.

В общепринятых обозначениях шахматных вертикалей и горизонталей указанный вариант расположения ферзей записывается так:

$$(a, 6), (b, 3), (c, 5), (d, 8), (e, 1), (f, 4), (g, 2), (h, 7).$$

Приведем еще один допустимый вариант расположения:

$$(a, 6), (b, 1), (c, 5), (d, 2), (e, 8), (f, 3), (g, 7), (h, 4).$$

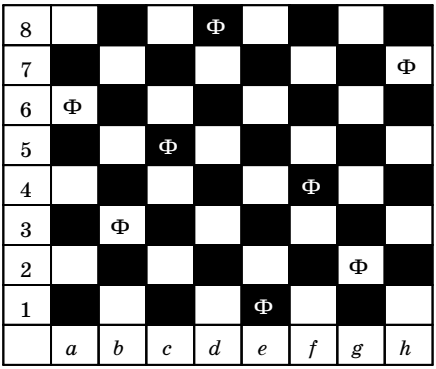


Рис. 10.3. Расположение ферзей таким образом, что ни один из них не может взять другого

На доске размером $n \times n$, где число n не должно делиться ни на 2, ни на 3, можно поставить по n ферзей n разнообразных цветов таким образом, чтобы ферзи одного цвета не били друг друга.

Если расставлять шахматных коней, то окажется, что любой из них на доске размером 2×4 может бить только одну клетку и на этой доске возможно разместить только четырех коней, которые не бьют друг друга. На доске 8×8 можно таким образом поставить только $8 \cdot 4 = 32$ коня, так как доска размером 8×8 клеток разбивается на восемь полей размера 2×4. На доске размером $n \times n$ можно поставить $(n \cdot n)/2$ не бьющих друг друга коней, если n четное, и $(n \cdot n + 1)/2$ коней, если n нечетное.

B. Латинские квадраты и блок-схемы. Квадрат размером n строк и n столбцов, составленный из n литер таким образом, чтобы каждая буква входила лишь однажды в каждый столбец и каждую строку, называют ***латинским***. На рис. 10.4 изображен латинский квадрат размером 4×4.

$$\begin{matrix} A & B & C & D \\ B & A & D & C \\ C & D & A & B \\ D & C & B & A \end{matrix}$$

Рис. 10.4. Латинский квадрат 4×4

Если два латинских квадрата «наложить» друг на друга таким образом, чтобы каждая пара букв (большая и маленькая) A, B, C, D

и a, b, c, d встретилась только один раз, то такие квадраты называют ***ортогональными***.

Ортогональный латинский квадрат размером 4×4 изображен на рис. 10.5.

Ab	*Db*	*Ba*	*Cc*
Bc	*Ca*	*Ad*	*Db*
Cd	*Bb*	*Dc*	*Aa*
Da	*Ac*	*Cb*	*Bd*

Рис. 10.5. Пара ортогональных латинских квадратов

Первым задачу об ортогональных латинских квадратах рассмотрел Л. Эйлер (задача об «офицерском каре»). Эта задача не имеет решения при $n = 2$ и $n = 6$.

Г. Теорема о представителях

Пусть в некотором множестве X выделены подмножества X_1, X_2, ..., X_n. *Для того, чтобы в X можно было избрать n разных элементов-представителей* a_1, a_2, ..., a_n, *таких, что* $a_1 \in X_1$, $a_2 \in X_2$, ..., $a_n \in X_n$, *необходимо и достаточно выполнение такого условия: для каждого значения* $k = 1, 2, ..., n - 1$ *объединение любых k выбранных подмножеств X должно содержать, по крайней мере, k неодинаковых элементов*. Сформулированная теорема о представителях была доказана Ф. Холлом. Сама проблема поиска представителей имеет другие названия или толкования. Например, название «*задача о сельских свадьбах*» возникло потому, что Герман Вейль впервые сформулировал подобную задачу в несколько шутливом тоне: «В селе относительно каждого парня и девушки известно, дружат они или нет. Если для любых k парней объединения множеств их подруг содержит, по крайней мере, k девушек, то каждый парень может избрать себе жену из числа своих подруг». Здесь X — множество всех девушек; X_1, X_2, ..., X_n — подмножества, которые состоят из подруг первого, второго, ..., n-го парня. Доказывают эту теорему методом математической индукции по количеству парней.

Заметим, что Ф. Холл доказал справедливость теоремы о различных представителях для любой системы подмножеств X_1, X_2, ..., X_n множества X, которое может содержать и бесконечное число элементов. Конечный вариант теоремы Ф. Холла эквивалентен *теореме Д. Кенига о матрицах* (любая из теорем легко выводится одна из одной). Теорема Кенига относится к булевым матрицам, которые встречаются в логике, целочисленном программировании,

в теории графов и сетей. Любой столбец или строка матрицы называются ее «***линией***».

Теорема Д. Кенига *(о матрицах)*

Пусть элементы прямоугольной матрицы состоят из нулей и единиц. Минимальное число линий, которые содержат все единицы матрицы, равняется максимальному числу q таких единиц в матрице, среди которых не существует двух расположенных на одной линии.

Каждое подмножество из q единиц матрицы с указанным свойством назовем ***базисным множеством единиц***. Рассмотрим примеры булевых матриц:

$$A=\begin{array}{c} 4\\3\\2\\1 \end{array}\begin{bmatrix} 0 & 1^* & 0 & 0 & 0\\ 0 & 0 & 1^* & 0 & 0\\ 0 & 0 & 0 & 0 & 0\\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}\begin{array}{c}\\\\\\\\ a\ \ b\ \ c\ \ d\ \ e \end{array},\qquad B=\begin{array}{c} 4\\3\\2\\1 \end{array}\begin{bmatrix} 1^* & 1 & 0 & 0\\ 1 & 1^* & 0 & 0\\ 0 & 0 & 1^* & 1\\ 1 & 0 & 0 & 0 \end{bmatrix}\begin{array}{c}\\\\\\\\ a\ \ b\ \ c\ \ d \end{array}.$$

У матрицы A базисное множество единиц обозначено звездочками, оно единственное. Число q равняется двум, а минимальное множество линий, которые содержат все единицы A, не одно. Собственно, существует 4 минимальные множества линий:

{строка 3, строка 4}, {столбец b, столбец c},
{строка 4, столбец c}, {строка 3, столбец b}.

У матрицы B число q равняется трем. Минимальное множество линий, которые содержат все единицы, одно: {столбец a, столбец b, строка 2}.

Базисное множество единиц обозначено звездочками, но оно не одно.

Задания

Пересчитать все базисные множества единиц у матрицы B.

Д. Правила суммы и произведения

Правило суммы. Если множество M — это объединение множеств M_1, M_2, M_3, ..., M_k, которые не пересекаются попарно, тогда количество элементов множеств связаны соотношением

$$|M| = |M_1| + ... + |M_k|.$$

Здесь $|M|$ — количество элементов множества M.

Пусть предмет a_1 можно избрать m_1 способами, предмет a_2 — m_2 способами, ..., предмет a_k — m_k способами. Тогда выбор предмета a_1, или a_2, ..., или a_k может быть сделан

$$m_1 + m_2 + \ldots + m_k \qquad (10.1)$$

способами.

Подчеркнем, что здесь союз «или» употребляется в разделительном содержании, то есть выбор a_i исключает выбор любого другого предмета.

Пример. Из города A в город B отправляются 10 поездов, 5 самолетов и 3 автобуса. Сколькими способами одному человеку можно добраться из A в B?

По правилу суммы всего существует $10 + 5 + 3 = 18$ способов.

Правило произведения. Если M_1, M_2, M_3, ..., M_k — конечные множества и $M = M_1 \times M_2 \times \ldots \times M_k$ — их декартово произведение, то

$$|M| = |\, M_1 \times M_2 \times \ldots \times M_k \,| = |M_1| \cdot |M_2| \cdot \ldots \cdot |M_k|.$$

Пусть предмет a_1 можно избрать m_1 способами, предмет a_2 — m_2 способами, ..., предмет a_k — m_k способами и пусть выбор предмета a_1 не влияет на количество способов выбора предметов a_2, ..., a_k; выбор предмета a_2 не влияет на количество способов выбора предметов a_1, a_3, ..., a_k и т.д. Тогда выбор *упорядоченного* множества предметов (a_1, a_2, ..., a_k) в указанном порядке можно осуществить

$$m_1 \cdot m_2 \cdot \ldots \cdot m_k \qquad (10.2)$$

способами.

Пример. Есть 17 парней и 21 девушка. Сколько танцевальных пар они могут образовать?

Сначала изберем парня — это можно сделать 17 способами, после этого каждый парень изберет себе партнершу (21 способ). По правилу произведения выбор упорядоченного множества танцевальных пар (парень, девушка) составляет $17 \cdot 21 = 375$ пар.

Е. Сложный выбор предметов

Часто в комбинаторных задачах выбор предметов осуществляется в несколько этапов, причем в некоторых из них выполняются условия, при которых действуют правила суммы, в других — правило произведения. Вообще «составной», «сложный» выбор предметов, который осуществляется путем последовательного выбора «простых» предметов, в практических задачах встречается довольно часто. Удобным средством анализа возможностей и перебора

случаев выбора является построение корневого дерева — каждой последовательности возможных выборов отвечает путь, который направляется от корня X_0 к концевым вершинам дерева. Конец любой из дуг первого уровня отвечает возможному результату выбора первого уровня и т.д. Таким образом, дерево строится уровень за уровнем, пока не будут исчерпаны все возможности выбора в последовательности, которая рассматривается. Конец каждой дуги k-уровня дерева отвечает единой последовательности результатов первых k выборов.

Пример. Код Морзе. Сколько разнообразных символов может быть записано кодом Морзе (·, –), если для их записи используется не больше пяти знаков?

Существует пять способов выбора длины символа: 1, 2, 3, 4 или 5 знаков. Из корня X_0 выйдут 5 дуг:

$$X_0X_1, \quad X_0X_2, \quad X_0X_3, \quad X_0X_4, \quad X_0X_5.$$

При длине кода 1 существуют 2 символа · и —, поэтому из вершины X_1 соответственно выбору длины символа, который равняется 1, проводим две дуги: X_1X_{11} и X_1X_{12}. При длине кода, которая равняется 2, существуют 4 символа: · ·, · –, – ·, ––; из вершины X_2 проводим 4 дуги:

$$X_2X_{21}, \quad X_2X_{22}, \quad X_2X_{23}, \quad X_2X_{24}.$$

При длине кода, который равняется 3, имеем 8 символов. Итак, из вершины X_3 проводим 8 дуг. Аналогично 16 символов длины 4 и 32 символа длины 5. Общее количество концевых вершин равняется $2 + 4 + 8 + 16 + 32 = 62$. Это и есть искомое число символов.

10.3. Перестановки, размещения, сочетания

Размещения и перестановки без повторений

Пусть $M = \{a_1, a_2, ..., a_n\}$ — фиксированное множество.

Определение

Упорядоченные подмножества из k элементов $(b_1, b_2, ..., b_k)$ множества M называются ***перестановками k*** элементов. Можно сказать, что k-перестановка $(b_1, b_2, ..., b_k)$ — это размещение в определенном порядке k элементов из множества M.

Определение

k-перестановки множества из n элементов называются ***размещениями из n по k*** элементов.

Если в перестановке принимают участие все элементы множества (n-перестановка), то используют термин ***перестановка,*** а количество всех *n-перестановок* обозначают через $\boldsymbol{P_n}$.

Два размещения из n по k различны, если они состоят из разных элементов или из одинаковых элементов, но расположенных в разном порядке.

Обозначение. $\boldsymbol{A_n^k}$ ***— количество k-размещений n-множества M*** (читается A из n по k). При образовании размещений первый элемент может быть избран n способами, так как существует возможность независимого выбора из n элементов, второй — $(n-1)$ различными способами, так как независимый выбор осуществляется для остатка из $(n-1)$ элемента, k-ий элемент — соответственно $n-k+1$ способами. Применим правило произведения:

$$A_n^k = n(n-1)\ldots(n-k+1) = \frac{n!}{(n-k)!}. \qquad (10.3)$$

Если $n = k$, тогда A_n^n — количество всех способов упорядочения этих элементов:

$$P_n = A_n^n = n!. \qquad (10.4)$$

Пример. $M = \{1, 2, 3\}$. Построить всяческие перестановки множества M по 2 и по 3 элемента.

Перестановки (размещения) по 2 элемента:

(1, 2), (2, 1), (1, 3), (3, 1), (2, 3), (3, 2).

Их количество $A_3^2 = 3 \cdot 2 = 6$.

Перестановки по 3 элемента, то есть просто перестановки элементов множества M:

(1, 2, 3), (1, 3, 2), (2, 1, 3), (2, 3, 1), (3, 1, 2), (3, 2, 1).

Их количество $P_3 = 3! = 1 \cdot 2 \cdot 3 = 6$.

Размещения и перестановки обязательно учитывают порядок элементов.

Размещения и перестановки с повторениями

Перестановки из n элементов в количестве (10.4) и размещения из n элементов по k в количестве (10.3) представляют ***упорядоченные выборки*** попарно различных элементов исходного множества M, $|M| = n$. Часто приходится делать упорядоченные выборки *с повторениями* некоторых элементов. Например, такими являются 3-выборки (1, 1, 2), (1, 2, 1), (2, 1, 1), (2, 3, 3), (3, 2, 3) из множества $M = \{1, 2, 3\}$. Такие общие упорядоченные k-выборки

(с повторениями либо без них) иногда называют ***кортежами*** (длины k). Два кортежа (то есть две общие выборки) считаются *одинаковыми*, если они имеют одинаковую длину и на местах с одинаковыми номерами стоят одинаковые элементы.

Определение

Размещением с повторениями из n элементов по k (***k-размещением*** с повторением из n) называется кортеж (выборка с повторением) длины k из n элементов.

Количество k-размещений с повторениями из n элементов вычисляется по формуле:

$$\overline{A_n^k} = n^k. \qquad (10.5)$$

Действительно, после заполнения первого места кортежа длиной k одним из n элементов (что возможно сделать n вариантами) заполнять второе место кортежа можно снова любым элементом из всего множества (повторяя в одном из вариантов элемент, который находится на первом месте), и так далее k раз. По правилу произведения получим, что $\overline{A_n^k} = n \cdot n \ldots n = n^k$.

Другое доказательство этого соотношения проводится методом математической индукции по k-количеству элементов в размещении при фиксированном значении n.

Пример. Если в виде исходного множества взять любое слово, например, БРАТ, тогда получим 24 слова из 4 разных букв (24 перестановки без повторений в соответствии с формулой (10.4) для $n = 4$). Если возможные повторения букв, например, БРРР, БАБА и др., тогда мы получим количество слов $\overline{A_4^4} = 4^4 = 256$. Множество 3-слов, использующих четыре различные буквы (таких, как БАР, РАБ, АРА и др.), состоит из $4^3 = 64$ слов.

Прежде чем рассматривать перестановки с повторениями, выясним, сколько неодинаковых кортежей длины 4 можно образовать, переставляя между собою буквы в слове ГАГА. Таких перестановок имеем только 6 (вместо числа перестановок $P_4 = 4! = 24$ из четырех разных букв без повторений):

ГАГА, ГААГ, ГГАА, АГАГ, ААГГ, АГГА.

Правило создания этих 6 кортежей можно переформулировать иначе, благоприятнее для обобщения.

Четырехэлементное множество M состоит из двух элементов множества $M_1 = \{A, A\}$ типа «A» и двух элементов множества $M_2 = \{Г, Г\}$ типа «$Г$». Сколько существует 4-выборок (перестановок) из

элементов M таких, которые содержат два элемента типа A и два элемента типа $Г$?

Обобщение этого примера обеспечивает удачное формулирование и быстрое решение задачи о *количестве упорядоченных перестановок с обозначенными индексами повторений*.

Формулировка. Пусть множество M из n предметов ($|M| = n$) состоит из множества M_1 предметов первого типа количеством $n_1(|M_1| = n_1)$, множества M_2 предметов второго типа количеством n_2 и т. д. до множества M_k предметов k-го типа количеством $n_k(|M_k| = n_k)$, причем

$$M = \bigcup_{i=1}^{k} M_i, \qquad n_1 + n_2 + \ldots + n_k = n, \qquad M_i \cap M_j = \varnothing,\ i \neq j.$$

Сколько существует таких n-перестановок с повторениями, которые содержат n_1 элементов первого типа, n_2 элементов второго типа, ..., n_k элементов k-го типа?

Элементы первого типа можно переставлять между собой $n_1!$ способами, но так как все элементы одинаковые, такие перестановки ничего не изменяют.

Перестановки элементов одного типа можно делать независимо от перестановок элементов другого типа, поэтому по правилу произведения элементы перестановки можно переставлять между собою $n_1!n_2!\ldots n_k!$ способами так, что общая n-перестановка не изменится. Итак, множество всех $n!$ перестановок из n элементов исходного множества распадается на части, которые состоят из $n_1!n_2!\ldots n_k!$ одинаковых перестановок. Количество разнообразных перестановок с указанными повторениями n_1, n_2, ... n_k равняется

$$P(n_1, n_2, \ldots, n_k) = \frac{n!}{n_1!n_2!\ldots n_k!}, \tag{10.6}$$

где $n = n_1 + n_2 + \ldots + n_k$. Список ($n_1$, n_2, ... n_k) называют ***спецификацией перестановки***.

Сочетания

В тех случаях, когда не интересен порядок элементов в комбинации, а интересен лишь состав комбинации, говорят о *сочетании* (иногда используется термин «*комбинация*»).

Сочетания без повторений

Определение

Сочетанием из n элементов по k (***k-сочетанием*** из ***n***) называется любое k-размещение (или *k-выборка*) из этих

элементов, в котором учитывается лишь состав элементов и не учитывается их порядок.

Количество k-сочетаний из n элементов обозначается C_n^k, или $\binom{n}{k}$, читается «количество сочетаний из n по k».

Из неупорядоченного множества элементов $a_{i_1}, a_{i_2}, \ldots, a_{i_k}$, можно образовать $k!$ упорядоченных k-размещений обозначенных элементов. Поэтому количество C_n^k всех k-сочетаний из n элементов в $k!$ раз меньше, чем количество A_n^k всех упорядоченных размещений из n элементов по k:

$$C_n^k = \frac{A_n^k}{k!} = \frac{n!}{(n-k)!k!}. \qquad (10.7)$$

Эта формула совпадает с формулой (10.6), если $n_1 = k$, $n_2 = n - k$:

$$P(k, n-k) = \frac{n!}{(n-k)!k!}, \qquad (10.8)$$

другими словами,

$$C_n^k = P(k,\ n-k). \qquad (10.9)$$

Пример. В футбольном чемпионате принимают участие 17 команд. При условии, что 3 последние команды оставляют высшую лигу, сколько вариантов такого завершения чемпионата?

Поскольку нас не интересует взаимный порядок команд, которые заняли последние места в лиге, применим формулу подсчета сочетаний — количества вариантов:

$$C_{17}^3 = \frac{17!}{3!(14)!} = \frac{17 \cdot 16 \cdot 15}{1 \cdot 2 \cdot 3} = 17 \cdot 8 \cdot 5 = 680.$$

Сочетания с повторениями

Имеем предметы n типов в таком количестве, что предметов каждого типа не меньше, чем k экземпляров. Сколько существует неупорядоченных *k-выборок* (комбинаций) предметов с возможными повторениями?

Эта проблема имеет остроумную «кондитерскую» иллюстрацию. В магазине продаются пирожные четырех сортов (типов): бизе, наполеоны, песочные и эклеры. Сколько существует комбинаций приобретения 7 пирожных? Здесь $n = 4$, $k = 7$.

Если идентифицировать типы пирожных номерами 1, 2, 3, 4 в той последовательности, как они записаны, то примеры приобретения пирожных имеют вид кортежей длины 4:

(1, 1, 1, 4), (2, 2, 2, 1), (2, 0, 0, 5), (0, 2, 3, 2), ... (10.10)

Второй кортеж означает, что мы приобрели 2 бизе, 2 наполеона, 2 песочника, 1 эклер.

Приведем эквивалентную формулировку этой задачи. Найти количество $\overline{C_n^k}$ ***сочетаний по k элементов с повторениями*** из n элементов множества $A = \{a_i\}_1^n$.

Обозначим через k_i количество повторений элемента a_i в определенном сочетании (комбинации)

$k_i \in \{0, 1, 2, ..., k\}$, $i = 1, 2, ..., n$. Тогда $k_1 + k_2 + ... + k_n = k$.

Задачу решают с помощью кодирования — зашифруем каждую комбинацию нулями и единицами: для каждого типа пишут столько единиц, сколько предметов этого типа входит в комбинацию и, если предметы некоторого типа не входят в комбинацию, тогда на соответствующем месте пишется 0. Соседние типы отделяют один от другого нулем, если оба соседних типа содержат хотя бы по одной единице. Так, четыре комбинации (10.10) из «кондитерского примера» кодируются соответственно таким образом:

(1, 0, 1, 0, 1, 0, 1, 1, 1, 1), (1, 1, 0, 1, 1, 0, 1, 1, 0, 1);
(1, 1, 0, 0, 0, 1, 1, 1, 1, 1), (0, 1, 1, 0, 1, 1, 1, 0, 1, 1).

При кодировании получим столько единиц, сколько предметов входит в комбинацию, то есть k единиц и $n - 1$ нулей. Разным комбинациям (или неупорядоченным выборкам) соответствуют разные перестановки с повторениями, а каждой перестановке соответствует комбинация нулей и единиц. Таким образом, количество ***k-сочетаний с повторениями из элементов n типов*** $\overline{C_n^k}$ равняется количеству $P(k, n - 1)$ перестановок с повторениями из k единиц и $n - 1$ нулей:

$$\overline{C_n^k} = P(k, n-1) = \frac{(k+n-1)!}{k!(n-1)!} = C_{n+k-1}^k. \qquad (10.11)$$

Прибавим условие — в комбинации должны входить элементы нескольких фиксированных типов (r типов) $r \le n$. Чтобы выполнить это условие, возьмем сразу же по одному элементу из данных r типов. Таким образом, в k-сочетании окажутся занятыми r мест, другие $k - r$ мест можно заполнить любыми элементами, поэтому комбинаций искомого вида будет

$$\overline{C_n^{k-r}} = C_{n+k-r-1}^{k-r} = \frac{(k+n-r-1)!}{(k-r)!(n-1)!}. \qquad (10.12)$$

Задания

1. В задаче с пирожными подсчитать количество вариантов покупки 7 пирожных.
2. При тех же самых условиях задачи подсчитать: сколько существует таких вариантов покупки 7 пирожных, которые содержат:
   а) хотя бы одно бизе;
   б) не менее чем 3 эклера.

Свойства сочетаний

Арифметический треугольник (треугольник Паскаля)

Последовательные значения C_n^k можно подсчитывать с помощью формул (10.13), которые проверяются непосредственно с помощью (10.7):

$$C_n^k = C_n^{n-k},\ \ C_n^k = C_{n-1}^{k-1} + C_{n-1}^k. \qquad (10.13)$$

Построим так называемый арифметический треугольник.

1. Положим $C_0^0 = 1$, запишем это значение в первую строку.

2. Во второй строке запишем значения $C_1^0 = 1$, $C_1^1 = 1$ таким образом, чтобы значение C_0^0 оказалось над промежутком между числами второй строки (рис. 10.6).

3. Следующая строка: первое и последнее (третье) числа есть $C_2^0 = C_2^2 = 1$. Между этими числами запишем значение $C_2^1 = 2$. По формуле (10.13) $C_2^1 = C_1^0 + C_1^1$, то есть C_2^1 равняется сумме чисел предшествующей строки, расположенных по левую сторону и по правую сторону от него: $C_2^1 = C_1^0 + C_1^1 = 1 + 1 = 2$.

4. По такому же правилу заполняем следующие строки — по краям пишем числа $C_n^0 = C_n^n = 1$, а все промежуточные значения получаем как суммы двух чисел предшествующей строки, расположенных по левую сторону и по правую сторону от искомого значения. В результате получим ***арифметический треугольник***, первые строки которого приведены на рис. 10.6.

```
          1
        1   1
      1   2   1
    1   3   3   1
  1   4   6   4   1
_   _   _   _   _   _
```

Рис. 10.6. Арифметический треугольник

При построении арифметического треугольника каждое число n-строки принимает участие в формировании двух чисел $(n+1)$-строки. Если сложить числа $(n+1)$-строки через одно, то полученная сумма равняется сумме всех чисел n-строки.

Через одно числа $(n+1)$-й строки можно складывать двумя способами — просуммировать числа, расположенные на нечетных позициях в строке, или те, что расположенны на четных позициях. В обоих случаях получим одинаковое число, которое равняется сумме чисел в строке с предшествующим номером n. Это свойство запишем в виде

$$C_n^0 + C_n^2 + C_n^4 + \ldots = C_n^1 + C_n^3 + C_n^5 + \ldots = C_{n-1}^0 + C_{n-1}^1 + C_{n-1}^2 + \ldots + C_{n-1}^{n-1}.$$

Избранные тождества

Из свойств арифметического треугольника вытекает, что сумма чисел $(n+1)$-й строки вдвое больше суммы чисел n-й строки. В первой строке стоит число 1, и сумма чисел первой строки 1. Поэтому в $(n+1)$-й строке сумма чисел равняется 2^n, то есть

$$C_n^0 + C_n^1 + \ldots + C_n^n = 2^n. \qquad (10.14)$$

Еще несколько свойств приведем без доказательства:

$$C_{n-1}^0 + C_n^1 + C_{n+1}^2 + \ldots + C_{n+m-1}^m = C_{n+m}^m; \qquad (10.15)$$

$$C_n^n + C_{n+1}^n + C_{n+2}^n + \ldots + C_{n+m-1}^n = C_{n+m}^{n+1}. \qquad (10.16)$$

Отсюда

$$1 + 2 + 3 + \ldots + m = \frac{m(m+1)}{2} = C_{m+1}^2; \qquad (10.17)$$

$$1^2 + 2^2 + 3^2 + \ldots + m^2 = \frac{m(m+1)(2m+1)}{6};$$

$$1^3 + 2^3 + 3^3 + \ldots + m^3 = \frac{m^2(m+1)^2}{4};$$

$$(C_p^0)^2 + (C_p^1)^2 + \ldots + (C_p^p)^2 = C_{2p}^p;$$

$$C_n^0 - C_n^1 + C_n^2 - \ldots + (-1)^n C_n^n = 0;$$

$$C_n^k \cdot C_{n-k}^{m-k} = C_m^k \cdot C_n^m.$$

10.4. Формула включений и исключений. Применение

Общая формула. Поставим задачу: «Пусть есть N предметов, некоторые из которых имеют свойства α_1, α_2, ..., α_n. При этом отдельный предмет может не иметь ни одного из этих свойств или иметь одно из свойств, или иметь свойства α_i, α_j, ..., α_h и, возможно, некоторые из других свойств. Обозначим через $N(\alpha_i, \alpha_j, ..., \alpha_h)$ количество предметов, которые имеют свойства α_i, α_j, ..., α_h. Если предметы не имеют свойства α_k, обозначим это штрихом α'_k. Например, $N(\alpha_1, \alpha_2, \alpha'_3)$ — количество предметов со свойствами α_1, α_2, которые не имеют свойства α_3. О других свойствах нам неизвестно, имеют ли их эти предметы или нет.

Общая формула включений и исключений подсчитывает количество предметов, лишенных всех без исключения свойств и имеет вид:

$$N(\alpha'_1,\alpha'_2,\ldots,\alpha'_n) = N - N(\alpha_1) - N(\alpha_2) - \ldots - N(\alpha_n) +$$
$$+N(\alpha_1,\alpha_2)+N(\alpha_1,\alpha_3)+\ldots+N(\alpha_1,\alpha_n)+\ldots+N(\alpha_{n-1},\alpha_n)-N(\alpha_1,\alpha_2,\alpha_3)-\ldots$$
$$\ldots - N(\alpha_{n-2},\alpha_{n-1},\alpha_n)+\ldots+(-1)^n N(\alpha_1,\alpha_2,\ldots,\alpha_n). \qquad (10.18)$$

Если ($\alpha_i\alpha_j$ ... $\alpha_k\alpha_n$) — комбинация свойств без учета их порядка, то знак «+» в формуле (10.18) ставится при условии, что количество свойств четно и «−» — если нечетно. Например, член $N(\alpha_1, \alpha_3, \alpha_8, \alpha_{11})$ входит с знаком «+», а член $N(\alpha_2, \alpha_4, \alpha_6)$ — со знаком «−».

Таким образом, в формуле включений и исключений сначала исключаются все предметы, которые имеют хотя бы одно из свойств α_1, α_2, ..., α_n, потом включаются предметы, которые имеют хотя бы два из этих свойств, дальше исключаются те, которые имеют, по крайней мере, три свойства и т.д.

Доказательство проводится методом индукции по количеству свойств.

Наложим следующие ограничения на количество элементов, которые обладают некоторыми из свойств α_1, α_2, ..., α_n:

$$N(\alpha_1) = ... = N(\alpha_n) = N^{(1)},$$
$$N(\alpha_1, \alpha_2) = N(\alpha_1, \alpha_3) = ... = N(\alpha_{n-1}, \alpha_n) = N^{(2)},$$
$$\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots,$$
$$N(\alpha_1, ..., \alpha_k) = ... = N(\alpha_{n-k+1}, ..., \alpha_n) = N^{(k)},$$
$$\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots,$$
$$N(\alpha_1, \alpha_2, ..., \alpha_n = N^{(n)}).$$

Таким образом, дополнительно предполагается, что *количество элементов* с любыми свойствами зависит только от количества этих свойств, а не от типа (или номеров) свойств, присущих предметам. Отсюда вытекает, что суммарное количество предметов, которые имеют ровно k свойств, равняется

$$N(\alpha_1,\ldots,\alpha_k)+\ldots+N(\alpha_{n-k+1},\ldots,\alpha_n)=C_n^kN^{(k)},k=1,2,\ldots,n.$$

При указанных ограничениях формула включений и исключений имеет вид (10.19):

$$N(\alpha'_1,\ldots,\alpha'_n)=N-C_n^1N^{(1)}+C_n^2N^{(2)}-\ldots+(-1)^nC_n^nN^{(n)}. \quad (10.19)$$

Применение формулы включений и исключений к перестановкам с ограничениями

Количество смещений D_n

Найти количество смещений D_n — перестановок из n элементов, из которых ни один не остается в первичном положении.

Задача о смещении известна давно: например, ею занимался еще Л. Эйлер. Значение D_n быстро получается с помощью формулы (10.19) включений и исключений с ограничениями. Пусть $N=P_n$ — количество всех перестановок из n элементов, $N^{(1)}=P_{n-1}$ — количество таких перестановок, в которых один фиксированный элемент α_i остается на своем месте. Вообще, $N^{(k)}=P_{n-k}$ — количество перестановок, в которых k фиксированных элементов (из этих n элементов) остаются на своем месте, $k = 1, \ldots, n$.

Тогда количество смещений D_n равняется $N(\alpha'_1, \ldots, \alpha'_n)$ — числу перестановок, в которых каждый элемент α_i не остается на своем месте. Согласно (10.19) имеем

$$D_n=P_n-C_n^1P_{n-1}+C_n^2P_{n-2}-\ldots+(-1)^nC_n^nP_0=n![1-\frac{1}{1!}+\frac{1}{2!}+\ldots+\frac{(-1)^n}{n!}].$$

Конечно, $D_1 = 0$.

Если $n = 0$, договоримся, что $D_0 = 1$.

Смещение неполного количества элементов

Количество перестановок $D_{n,r}$ из n элементов, при которых ровно r элементов остаются на первоначальных местах, а другие $n - r$ обязательно изменяют свое положение, равняется

$$D_{n,r}=C_n^rD_{n-r}.$$

□ Сначала следует выбрать, какие именно r элементов останутся на месте. Это можно сделать C_n^r способами. Другие $n - r$ элементов

можно переставлять любыми способами, лишь бы ни один из них не оказался на своем начальном месте. Это можно сделать D_{n-r} способами. По правилу произведения получим $C_n^r D_{n-r}$ искомых перестановок. ■

Разобьем все перестановки на классы в зависимости от того, сколько элементов остаются несмещенными при данной перестановке

$$\sum_{r=0}^{n} D_{n,r} = \sum_{r=0}^{n} C_n^r D_{n-r} = n!,$$

(так как общее количество перестановок равняется $n!$).

С помощью формулы включений и исключений можно решить такую задачу: «Найти количество перестановок из n элементов, при которых данные r элементов смещены (а другие могут быть или на своих местах, или смещеными)». Это количество равняется

$$n! - C_r^1(n-1)! + C_r^2(n-2)! - \ldots + (-1)^r(n-r)!.$$

Аргументация осуществляется по образцу вывода формулы для D_n: от первого шага — исключение из P_n количества перестановок $C_r^1 P_{n-1}$,в которых один из r элементов не изменяет своего места, до r-го шага — учет количества $C_r^r P_{n-r} = P_{n-r}$ перестановок, в которых r указанных элементов остается на своих местах:

$$P_n - C_r^1 P_{n-1} + C_r^2 P_{n-2} - \ldots + (-1)^r C_r^r P_{n-r}.$$

Задачи о размещении предметов

Решая задачи о размещении (распределении предметов по урнам), необходимо проанализировать содержание задачи, физические свойства предметов, чтобы правильно решить вопрос о том, считать ли предметы одинаковыми или разными, учитывать или нет порядок выборки предметов, различать или нет урны одну от другой. Если задача формулируется так, что невозможно однозначно ответить на подобные вопросы, необходимо принять дополнительные предположения.

Распределение n разных предметов по k урнам. Количество размещений по k урнам n разных предметов при условии, чтобы в первую урну попало n_1 предметов, во вторую — n_2, ..., в последнюю — n_k предметов, равняется:

$$P(n_1, n_2, \ldots, n_k) = \frac{n!}{n_1! n_2! \ldots n_k!}, \qquad (10.20)$$

$$n = n_1 + n_2 + \ldots + n_k.$$

Задания

Исходя из совпадения формул (10.20) и (10.6), установить связь между задачами о n-перестановках с повторениями (п. 10.3) и распределение предметов между урнами.

Указание. Номера любого из n_i предметов i-ой урны отвечают номерам мест, которые занимают в перестановках повторяемые элементы i-го типа.

Распределение n одинаковых предметов по k урнам

Установим, что n одинаковых предметов между k лицами можно разделить

$$P(n, k-1) = C_{n+k-1}^{n} = C_{n+k-1}^{k-1}$$

способами.

□ Выстроим в одну строку n одинаковых предметов и создадим дополнительно множество из $(k-1)$ предметов второго типа («разделителей»). Вставляя все дополнительные предметы на разные места в строке между исходными предметами, можно разделить последние на «k» последовательных цепочек, которые вручаются соответствующим лицам. Понятно, что количество искомых распределений n одинаковых предметов между k лицами равняется количеству перестановок из $n + (k-1)$ предметов с повторениями n предметов первого типа и $(k-1)$ предметов второго типа, то есть равняется $P(n, k-1)$. Если при этом m разделителей предшествуют всем n предметам первого типа, тогда лица $1, 2, \ldots, m$ не получают ни единого предмета при разделе, $m \in \{1, \ldots, k-1\}$. ■

Если делить справедливо, с уровнем справедливости r, то каждый участник распределения должен получить минимум r предметов. Начнем с того, что каждому раздадим по r предметов. После этого останется $n - kr$ предметов, которые распределим произвольно. Это можно сделать

$$C_{n-kr+k-1}^{k-1} = C_{n-k(r-1)-1}^{k-1}$$

способами.

Распределение разных предметов без учета порядка предметов в урнах

Необходимо распределить n различных предметов между k урнами, вместительность которых не ограничена (каждый человек, который принимает участие в распределении, может забрать себе все). Первый предмет можно положить в любую из k урн, второй (независимо от первого) также в k урн и т.д. По правилу произведения предметы можно распределить k^n способами.

Распределение разных предметов между одинаковыми урнами при условии, что урны не пустые

Обозначим S_n^k количество способов распределения n разных предметов между k одинаковыми урнами так, чтобы каждая урна была не пустой. Из каждого такого распределения можно получить $k!$ аналогичных распределений по разным урнам (пометить k урн можно $k!$ способами). Таким образом, *количество $M(n, k)$ распределений n разных предметов между k разными урнами с использованием каждой урны в каждом распределении* («не пустые урны») равняется $M(n, k) = k!S_n^k$.

Количества $M(n, k)$ носят название ***чисел Моргана***. Формула

$$S_n^k = \frac{1}{k!}\left[k^n - C_k^1(k-1)^n + C_k^2(k-2)^n + \ldots + (-1)^{k-1}C_k^{k-1}1^n\right] \quad (10.21)$$

вытекает из формулы включений и исключений.

Числа S_n^k называются ***числами Стирлинга второго рода***. Определим:

$$S_n^k = 0, \text{ если } k > n \text{ или } k = 0. \quad \text{Также } S_0^0 = 1,\ S_n^n = 1.$$

Числа Стирлинга S_n^k удовлетворяют рекурсивному соотношению:

$$S_{n+1}^k = kS_n^k + S_n^{k-1}.$$

Числа Белла

Числа Белла B_n — это количество всех способов разбиения множества из n элементов в объединение непустых подмножеств, которые не пересекаются. Понятно, что

$$B_n = \sum_{k=0}^{n} S_n^k,$$

где $B_0 = 1$, $S_n^0 = 0$.

Проверяется, что

$$B_{n+1} = \sum_{k=0}^{n} C_n^k B_k.$$

Распределение разных предметов с учетом их порядка в урнах

Если не ограничивать количество предметов в урнах и если предметы не различать между собой, то количество таких распределений равняется C_{n+k-1}^{k-1}.

Каждому способу распределения одинаковых предметов между урнами отвечает $n!$ способов распределения разных предметов

с учетом их порядка, которые получаются с помощью перестановок предметов между собой без изменения количества предметов в урнах. По правилу произведения получаем искомое количество распределений:

$$C_{n+k-1}^{k-1} \cdot n! = \frac{(n+k-1)!}{(k-1)!} = A_{n+k-1}^{n}.$$

10.5. Биномиальная и полиномиальная формулы

Бином Ньютона. Формулу

$$(a+b)^n = C_n^0 a^0 b^n + C_n^1 a^1 b^{n-1} + \ldots + C_n^k a^k b^{n-k} + \ldots + C_n^n a^n b^0 \quad (10.22)$$

докажем комбинаторными соображениями. Если выражение $(a+b)^n$ записать в виде произведения множителей $(a+b)(a+b)\ldots(a+b)$, то после их перемножения без изменения порядка мы получим подобные слагаемые, у любого из которых в разном порядке входят k множителей a и $(n-k)$ множителей $b (k = n, n-1, \ldots, 0)$. Количество подобных членов $a^k b^{n-k}$ с фиксированными степенями k и $n-k$ равняется количеству вариантов записи буквы a на k позициях из n возможных позиций, то есть равняется C_n^k. Отсюда вытекает формула (10.22).

Полиномиальная формула

$$(a_1 + a_2 + \ldots + a_m)^n = \sum_{k_i=0}^{n} P(k_1, k_2, \ldots, k_m) \cdot a_1^{k_1} a_2^{k_2} \ldots a_m^{k_m}, \quad (10.23)$$

где $k_1 + k_2 + \ldots + k_m = n$ есть обобщение формулы бинома Ньютона (10.22). Доказательство сразу же вытекает из того, что количество подобных членов $a_1^{k_1} \cdot a_2^{k_2} \cdot \ldots \cdot a_m^{k_m}$, полученных после перемножения n множителей

$$(a_1 + a_2 + \ldots + a_m)(a_1 + a_2 + \ldots + a_m) \ldots (a_1 + a_2 + \ldots + a_m),$$

равняется числу разбиений n мест на m групп для k_1 множителей a_1, k_2 множителей a_2, ..., k_m множителей a_m. Общее количество таких разбиений равняется $P(k_1, k_2, \ldots, k_m)$ (см. (10.20)).

10.6 Комбинаторные задачи и теория чисел

Решето Эратосфена

Одна из больших загадок математики — размещение простых чисел в натуральном ряде. *Эратосфен* предложил решения

задачи о нахождении всех простых чисел среди натуральных от 1 до N (сейчас 1 считается особым числом). Способ Эратосфена: сначала вычеркивают все числа, которые делятся на 2, оставляя само число 2, потом берут следующее число — 3, понятно, что оно простое, и вычеркивают все следующие числа, которые делятся на 3 и т.д. Числа, которые остались после вычеркивания, и есть простые.

Подсчитаем, сколько простых чисел в первой сотне. Обозначим это количество через $\pi(100)$. Простые числа, меньшие, чем $\sqrt{100}=10$, есть числа 2, 3, 5, 7. Применим алгоритм решета Эратосфена к последовательности

$$2, 3, 4, 5, 6, 7, 8, \ldots, 100$$

и вычеркнем все числа, которые имеют хотя бы один делитель 2, 3, 5, 7, а самые эти делители оставляем в последовательности. Тогда остается $\pi(100)$ простых чисел. Обозначим свойства:

α_1 — делимость на 2;
α_2 — делимость на 3;
α_3 — делимость на 5;
α_4 — делимость на 7.

Понятно, что в последовательности 1, 2, 3, ..., 100 количество чисел $N(\alpha_1', \alpha_2', \alpha_3', \alpha_4')$, которые не имеют ни единого свойства $\alpha_i (i = 1, 2, 3, 4)$, равняется $N(\alpha_1', \alpha_2', \alpha_3', \alpha_4') = \pi(100) + 1 - 4 = \pi(100) - 3$. Здесь добавлена 1 — количество особых чисел (единица) и вычтено 4 — количество первых четырех простых чисел (2, 3, 5, 7), каждое из которых обладает в точности одним из свойств α_i.

Напомним, что $N(\alpha_1) = [\frac{100}{2}] = 50$ — количество чисел от 1-го до 100, которые делятся на 2; $N(\alpha_1\alpha_2) = [\frac{100}{2\cdot 3}] = 16$ — делятся на 2 и 3 одновременно; $N(\alpha_2, \alpha_3, \alpha_4) = [\frac{100}{3\cdot 5\cdot 7}] = 0$ — делятся на 3, 5, 7 одновременно; $N(\alpha_1, \alpha_2, \alpha_3, \alpha_4) = [\frac{100}{2\cdot 3\cdot 5\cdot 7}] = 0$ — делятся на любое из чисел 2, 3, 5, 7. Здесь $[k]$ есть обозначение целой части числа k.

По формуле включений и исключений имеем:

$$\pi(100) - 3 = N(\alpha_1', \alpha_2', \alpha_3', \alpha_4') = 100 - N(\alpha_1) - N(\alpha_2) -$$
$$- N(\alpha_3) - N(\alpha_4) + N(\alpha_1, \alpha_2) + N(\alpha_1, \alpha_3) + N(\alpha_1, \alpha_4) + N(\alpha_2, \alpha_3) +$$
$$+ N(\alpha_2, \alpha_4) + N(\alpha_3, \alpha_4) - N(\alpha_1, \alpha_2, \alpha_3) - N(\alpha_1, \alpha_2, \alpha_4) - N(\alpha_1, \alpha_3, \alpha_4) -$$
$$- N(\alpha_2, \alpha_3, \alpha_4) + N(\alpha_1, \alpha_2, \alpha_3, \alpha_4) = 100 - 50 - 33 - 20 - 14 + 16 +$$
$$+ 10 + 7 + 6 + 4 + 2 - 3 - 2 - 1 - 0 + 0 = 22.$$

Окончательно получаем, что в первой сотне помещается $\pi(100) = 25$ простых чисел.

Задания

Привести все 25 простых чисел от 2 до 100. Сколько среди них насчитывается «пар близнецов» — пар простых чисел, которые отличаются на 2? Сколько простых чисел попадает в каждый десяток — первый, второй, ..., десятый?

Рекуррентные соотношения

Метод рекуррентных соотношений состоит в том, что решение комбинаторной задачи с n предметами выражается через решение аналогичной задачи с меньшим количеством предметов с помощью некоторого соотношения, которое называется рекуррентным («обратным»). Пользуясь этим соотношением, искомую величину можно вычислить, исходя из того, что для небольшого количества предметов (одного, двух) решение задачи легко находится.

Проиллюстрируем метод рекуррентных соотношений на примере сочетаний с повторениями.

Обозначим количество сочетаний из n предметов $\{a_1, ..., a_n\}$ по k с повторениями через f_n^k (в разделе 10.3 было применено *обозначение* $\overline{C}_n^k$). Любое из сочетаний или содержит, или не содержит a_1. Количество сочетаний, которые не содержат a_1, равняется f_{n-1}^k (это сочетания из предметов $a_2, ..., a_n$ по k). Каждое сочетание, которое содержит a_1 как минимум один раз, может быть получено присоединением к a_1 некоторого сочетания из n предметов по $k-1$ (количество таких сочетаний равняется f_n^{k-1}). Итак,

$$f_n^k = f_{n-1}^k + f_n^{k-1}. \qquad (10.24)$$

Последовательно применяя данное рекуррентное соотношение, получим

$$f_n^k = f_n^{k-1} + f_{n-1}^k = f_n^{k-1} + (f_{n-1}^{k-1} + f_{n-2}^k) = \ldots = f_n^{k-1} + f_{n-1}^{k-1} + \ldots + f_2^{k-1} + f_1^k. \qquad (10.25)$$

Понятно, что $f_n^1 = n$, $f_1^k = 1$.

Считая $k = 2$, имеем (см. (10.17)):

$$f_n^2 = n + (n-1) + \ldots + 2 + 1 = n(n+1)/2 = C_{n+1}^2. \qquad (10.26)$$

Если $k = 3$, тогда с учетом (10.16) имеем:

$$f_n^3 = C_{n+1}^2 + C_n^2 + \ldots + C_3^2 + C_2^2 = C_{n+2}^3.$$

На $(k-1)$-м шаге получим:

$$f_n^k = C_{n+k-1}^k = P(k, n-1), \qquad (10.27)$$

что согласовывается с предшествующим результатом (10.11).

Числа Фибоначчи

Леонардо Фибоначчи в 1202 г. решал задачи с помощью рекуррентного соотношения, в котором n-ый член выражался с помощью двух предшествующих. Задача, которая рассматривалась, имеет следующий вид:

«Каждая пара взрослых кролей приносит ежемесячно еще пару кроликов (самку и самца), которые, в свою очередь, начинают давать такой самый приплод через два месяца после своего рождения. Сколько пар кроликов будет через год, если в начале года была одна пара новорожденных кроликов и ни один из них за год не погиб?»

Обозначим через u_n количество пар кроликов в начале n-го месяца. Тогда по условию $u_1 = 1$, $u_2 = 1$, $u_3 = 2$, так как к началу 3-го месяца появляется приплод. Далее $u_4 = 3$, $u_5 = 5$, так как приплод дает как первоначальная пара, так и пара, родившаяся в конце второго месяца. Легко видеть, что данные числа связаны между собой соотношением:

$$u_n = u_{n-1} + u_{n-2}, \qquad (10.28)$$

так как в начале n-го месяца имеем все пары, которые были в начале предшествующего, и, кроме того, приплод принесут все пары, родившиеся за два месяца до данного и раньше.

Легко подсчитать, что $u_{13} = 233$, то есть через год будет 233 пары кроликов.

Числа $u_1, u_2, \ldots, u_n$ называют ***числами Фибоначчи***. Эти числа встречаются в разнообразных разделах математики, например, при оптимальном нахождении точек екстремума методом проб.

Приведем 14 членов последовательности $\{u_n\}$ с начальными членами $u_1 = 1$, $u_2 = 1$: $\{u_n\}_{n=1}^{\infty} = 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, 233, 377, \ldots$

Иногда последовательность чисел Фибоначчи начинают с чисел $u_0 = 0$, $u_1 = 1$, дополняя натуральный индекс значением $n = 0$:

$$\{u_n\}_{n=0}^{\infty} = 0, 1; 1, 2, 3, 5, 8, 13, 21, 34, 55, \ldots \qquad (10.29)$$

Тогда рекуррентное соотношение (10.28) выполняется для значений индекса $n \geq 2$.

В других случаях рассматривается последовательность чисел Фибоначчи f_n с начальными членами

$f_0 = 1$, $f_1 = 2$: $\{f_n\}_{n=0}^{\infty} = 1, 2; 3, 5, 8, 13, 21, 34, 55, 89, \ldots$ (10.30)

Сам Фибоначчи предложил именно эту последовательность, исходя из одной пары взрослых кроликов в преддверии первого месяца ($f_0 = 1$) и получая после первого рождения две пары кроликов на первом месяце ($f_1 = 2$). Рекуррентное соотношение (10.28) также выполняется для последовательности (10.30), если $n \geq 2$:

$$f_n = f_{n-1} + f_{n-2}.$$

Понятно, что последовательность чисел Фибоначчи (10.30) можно сдвинуть влево, например, выбирая начальные элементы $f_0 = 13$, $f_1 = 21$, или сдвинуть вправо, например, выбирая

$$f_0 = -3, \quad f_1 = 2.$$

Тогда удовлетворяя рекуррентному соотношению (10.28), будем иметь соответственно последовательности

$$13, 21; 34, 55, 89, 144, 233, 377, \ldots$$

$$-3, 2; -1, 1, 0, 1, 1, 2, 3, 5, 8, 13, 21, 34, \ldots \quad (10.31)$$

Первая из этих последовательностей является частью последовательности Фибоначчи (10.30), вторая — наоборот, содержит последовательность (10.30) как свою часть.

Тем не менее, не всякая последовательность $\{f_n\}_0^{\infty}$, которая удовлетворяет рекуррентному условию $f_n = f_{n-1} + f_{n-2}$, содержит в своем составе какую-нибудь часть последовательных чисел Фибоначчи (10.30), например,

$$2, 5; 7, 12, 19, 31, 50, 91, \ldots$$

$$-2, 6, 4, 10, 14, 24, 38, 62, \ldots$$

Приведем явную формулу для вычисления произвольного числа Фибоначчи, которую впервые получил Бине. Для последовательности (10.29) формула имеет вид

$$u_n = \frac{1}{\sqrt{5}}\left(\frac{1+\sqrt{5}}{2}\right)^n - \frac{1}{\sqrt{5}}\left(\frac{1-\sqrt{5}}{2}\right)^n, \quad n = 0, 1, 2, \ldots \quad (10.32)$$

Действительно, для последовательности (10.32)

$$u_{n+2} - u_n = \frac{1}{\sqrt{5}}\frac{(1+\sqrt{5})^n}{2^n}\left[\frac{(1+\sqrt{5})^2}{2^2} - 1\right] - \frac{1}{\sqrt{5}}\frac{(1-\sqrt{5})^n}{2^n}\left[\frac{(1-\sqrt{5})^2}{2^2} - 1\right] =$$

$$= \frac{1}{\sqrt{5}}\frac{(1+\sqrt{5})^n}{2^n}\cdot\frac{1+\sqrt{5}}{2} - \frac{1}{\sqrt{5}}\frac{(1-\sqrt{5})^n}{2^n}\cdot\frac{1-\sqrt{5}}{2} = u_{n+1}$$

так что рекуррентное соотношение (10.28) выполняется. Начальные значения $u_0 = 0$, $u_1 = 1$ получаются из (10.32), если $n = 0$, $n = 1$ соответственно.

Учитывая сдвиг индекса на 2, для последовательности чисел Фибоначчи f_n (10.30) имеем такой вид формулы Бине:

$$f_n = \frac{1}{\sqrt{5}}\left(\frac{1+\sqrt{5}}{2}\right)^{n+2} - \frac{1}{\sqrt{5}}\left(\frac{1-\sqrt{5}}{2}\right)^{n+2}, \quad n = 0, 1, 2, \ldots$$

Особая роль чисел $\frac{1+\sqrt{5}}{2}$, $\frac{1-\sqrt{5}}{2}$ в формуле Бине поясняется тем, что эти числа являются корнями уравнения

$$\lambda^2 - \lambda - 1 = 0: \quad \lambda_1 = \frac{1+\sqrt{5}}{2}, \ \lambda_2 = \frac{1-\sqrt{5}}{2}. \qquad (10.33)$$

Это уравнение носит название ***характеристического*** для рекуррентного соотношения Фибоначчи (10.28) $u_n - u_{n-1} - u_{n-2} = 0$ и возникает, если искать решение (10.28) в виде $u_n = \lambda^n$:

$$\lambda^n - \lambda^{n-1} - \lambda^{n-2} = 0 \sim \lambda^{n-2}(\lambda^2 - \lambda - 1) = 0.$$

Докажем, что общее решение рекуррентного соотношения (10.28) — последовательность $\{u_n\}_{n=0}^{\infty}$ с любыми начальными значениями u_0, u_1 — имеет вид

$$u_n = c_1\lambda_1^n + c_2\lambda_2^n, \quad n = 0, 1, 2, \ldots, \qquad (10.34)$$

где постоянные c_1, c_2 зависят только от выбора начальных значений u_0, u_1.

Действительно, первые два соотношения в (10.34) образуют систему двух линейных уравнений относительно постоянных c_1, c_2:

$$\begin{cases} 1c_1 + 1c_2 = u_0, \\ \lambda_1 c_1 + \lambda_2 c_2 = u_1. \end{cases}$$

Детерминант этой системы равняется $\Delta = \lambda_2 - \lambda_1 = -\sqrt{5} \neq 0$.

Система имеет единственное решение

$$c_1 = -\frac{\lambda_2}{\sqrt{5}}u_0 + \frac{1}{\sqrt{5}}u_1; \quad c_2 = \frac{\lambda_1}{\sqrt{5}}u_0 - \frac{1}{\sqrt{5}}u_1.$$

В частности, последовательности (10.29) отвечают в формуле (10.34) постоянные

$$c_1 = \frac{1}{\sqrt{5}}, \ c_2 = -\frac{1}{\sqrt{5}} \qquad (u_0 = 0, u_1 = 1),$$

последовательности (10.30) — постоянные

$$c_1 = \frac{1}{2} + \frac{3}{2\sqrt{5}}, \quad c_2 = \frac{1}{2} - \frac{3}{2\sqrt{5}} \qquad (u_0 = f_0 = 1, u_1 = f_1 = 2).$$

И наконец, последовательности с начальными значениями (−2), 6 отвечают постоянные

$$c_1 = \frac{7}{\sqrt{5}} - 1, \quad c_2 = -\frac{7}{\sqrt{5}} - 1.$$

Асимптотическое поведение общего решения $\{u_n\}$ (10.34) рекуррентного соотношения (10.28) и чисел Фибоначчи u_n (10.29) при условии $n \to \infty$ выясняется после замечания $|\lambda_2| = \left|\frac{1-\sqrt{5}}{2}\right| < 1$, из которого вытекает сходимость $\lambda_2^n \to 0$, $n \to \infty$. Таким образом, $u_n \sim c_1\lambda_1^n (n \to \infty)$. В частности, для чисел Фибоначчи (10.29)

$$u_n \sim \frac{1}{\sqrt{5}}\left(\frac{1+\sqrt{5}}{2}\right)^n, \qquad n \to \infty. \tag{10.35}$$

Более того, число Фибоначчи u_n есть ближайшее к $\frac{1}{\sqrt{5}}\left(\frac{1+\sqrt{5}}{2}\right)^n$ целое число.

К числам Фибоначчи приводит, например, следующая задача теории информации. Обозначим через f_n количество сообщений, которые можно передать с помощью n сигналов 0 и 1 так, чтобы при этом нигде не оказались рядом два сигнала 0. Понятно, что $f_0 = 1$ (отсутствие сигнала рассматривается как одно «пустое» сообщение), $f_1 = 2$. Все сообщения «длиной» $n + 2$ количеством f_{n+2} можно поделить на два класса:

а) те, в которых первый сигнал есть 1; их количество равняется f_{n+1}, так как второй сигнал может быть как 0, так и 1;

б) те, в которых первый сигнал есть 0, и, следовательно, второй сигнал обязательно есть 1; их количество равняется f_n.

Итак, $f_{n+2} = f_{n+1} + f_n$, поэтому f_n являются числами Фибоначчи (10.30).

10.7. Композиции и разбиения

Композиции

При решении задач о распределении n одинаковых предметов по k непустым ячейкам можно говорить о разложении числа n в сумму натуральных слагаемых n_i:

$$n = n_1 + n_2 + \ldots + n_k, \quad \text{где } k \geq 1,\ n_i \geq 1. \qquad (10.36)$$

Два таких разложения числа n считаются разными, если они отличаются хотя бы одним слагаемым. В таком случае говорят о композициях числа n. Иначе, ***композиции числа n*** — это его разложения в виде (10.36), где учитывается как *величины слагаемых* («частей») n_i, так и *порядок их расположения в сумме* (10.36).

Выпишем, например, все композиции числа 3:

$$3 = 3,\ 3 = 2 + 1,\ 3 = 1 + 2,\ 3 = 1 + 1 + 1.$$

Композиции с ограничениями на количество слагаемых. Число композиций $\Psi(k, n)$ числа n с k слагаемыми равняется числу распределений n одинаковых предметов по k различным ячейкам при условии отсутствия пустых ячеек. Число предметов, которые попали в ячейку с номером i, дает слагаемое n_i. Отсюда вытекает, что $\Psi(k, n) = C_{n-1}^{k-1}$. Действительно, в соответствии с п. 10.4 количество распределений n одинаковых предметов между k лицами (в нашем случае — между k ячейками) с уровнем справедливости r равняется $C_{n-k(r-1)-1}^{k-1}$. При наших условиях каждая ячейка получает не меньше одного предмета, так как $n_i > 0$. Таким образом, следует установить «уровень справедливости» $r = 1$, и мы получаем число распределений $\psi(k, n) = C_{n-1}^{k-1}$. В конце концов остается объединить числа композиций $\psi(k, n)$ по всевозможным количествам слагаемых $k = 1, 2, \ldots, n$. ***Число всех композиций числа n*** составляет:

$$\sum_{k=1}^{n} \Psi(k, n) = \sum_{k=1}^{n} C_{n-1}^{k-1} = 2^{n-1}, \qquad (10.37)$$

где последнее равенство обосновывается с помощью тождества (10.14).

Разбиения

Определение

Разбиением числа n *называется его представление в виде* (10.36), *если порядок слагаемых* n_i *не учитывается.*

Подсчет количества разбиений эквивалентен задаче о количестве распределений n одинаковых предметов по одинаковым непустым ячейкам. Как и для композиций, слагаемые n_i в разбиениях называются ***частями***; количество вхождений каждой части в сумму (10.36) называется ее ***кратностью***. Разбиение числа n на k слагаемых $n_1, n_2, \ldots, n_k$ можно рассматривать как k-сочетание (с повторениями) этих чисел, которое удовлетворяет условию (10.36). Иногда для поиска композиций числа n

(или вычисления их количества) удобно решить эту же задачу для его разбиений, а после этого перейти к композициям.

Поскольку в разбиениях (в отличие от композиций) порядок частей (слагаемых) не учитывается, договорились в разбиении (10.36) располагать слагаемые так, чтобы они не возрастали:

$$n_1 \geq n_2 \geq \ldots \geq n_k \quad (k \geq 1). \tag{10.38}$$

Количество всех разбиений числа n на k частей обозначим как $\boldsymbol{p_k(n)}$, а количество всех разбиений числа n со всевозможными значениями числа частей $k(1 \leq k \leq n)$ — через $p(n)$. Понятно, что по правилу суммы

$$p(n) = \sum_{k=1}^{n} p_k(n), \tag{10.39}$$

причем $p_1(n) = 1$, $p_n(n) = 1$, $p_k(n) = 0$ для $n < k$.

Из равенства $n = \sum_{i=1}^{k} n_i$ (10.36) формально получаем

$$\sum_{i=1}^{k} (n_i - 1) = n - k, \quad (n_i - 1) \geq 0. \tag{10.40}$$

Равенство (10.40) можно считать разбиением числа $(n - k)$ на k_0 слагаемых, где $k_0 = \min\{i: n_i = 1\}$ (см. (10.38)). Если в разбиении (10.36) $n_i > 1$ для всех частей, тогда $k_0 = k$. По правилу суммы количество разбиений числа $(n - k)$ на части, которых не более чем k штук, равняется $\sum_{i=1}^{k} p_i(n-k)$. Каждому такому разбиению числа $(n - k)$ на k_0 частей

$$n - k = m_1 + m_2 + \ldots + m_{k_0}, \; 1 \leq k_0 \leq k$$

однозначно отвечает разбиение n на k частей вида

$$n = (m_1 + 1) + (m_2 + 1) + \ldots + (m_{k_0} + 1) + \underbrace{1 + 1 + \ldots + 1}_{(k - k_0) \text{ раз}}.$$

Поэтому

$$p_k(n) = \sum_{i=1}^{k} p_i(n-k). \tag{10.41}$$

Это рекуррентное соотношение определяет числа $p_k(n)$ при начальных условиях $p_k(k) = 1$, $p_k(n) = 0$, если $n < k$. Для использования удобнее переписать (10.41) в развернутом виде

$$p_k(n) = p_k(n - k) + p_{k-1}(n - k) + \ldots + p_1(n - k). \tag{10.42}$$

Последовательно имеем: $p_1(n) = 1$, $\forall n \geq 1$;

$$p_2(n) = p_2(n - 2) + p_1(n - 2) = p_2(n - 2) + 1, \; n \geq 2.$$

Таким образом,

$$p_2(1) = 0,\ p_2(2) = 1,\ p_2(3) = p_2(1) + 1 = 1;$$
$$p_2(4) = p_2(2) + 1 = 1 + 1 = 2,\ p_2(5) = p_2(3) + 1 = 1 + 1 = 2;$$
$$p_2(6) = p_2(4) + 1 = 2 + 1 = 3;\ p_2(7) = 3,\ p_2(8) = 4;$$
$$p_2(9) = 4,\ p_2(10) = 5 \text{ и т.д.}$$

Далее

$$p_3(n) = p_3(n-3) + p_2(n-3) + p_1(n-3);$$
$$p_3(n) = p_3(n-3) + p_2(n-3) + 1,\ n \geq 3.$$

Итак,

$$p_3(3) = 1,\ p_3(4) = p_3(1) + p_2(1) + 1 = 0 + 0 + 1;$$
$$p_3(5) = p_3(2) + p_2(2) + 1 = 0 + 1 + 1 = 2;$$
$$p_3(6) = p_3(3) + p_2(3) + 1 = 1 + 1 + 1 = 3 \text{ и т.д.}$$

Значения числа разбиений $p_k(n)$ для $1 \leq k,\ n \leq 10$ приведены в таблице 10.1.

Таблица 10.1. Значения $p_k(n)$, $p(n)$

k \ n	1	2	3	4	5	6	7	8	9	10
1	1	1	1	1	1	1	1	1	1	1
2	0	1	1	2	2	3	3	4	4	5
3	0	0	1	1	2	3	4	5	7	8
4	0	0	0	1	1	2	3	5	6	9
5	0	0	0	0	1	1	2	3	5	7
6	0	0	0	0	0	1	1	2	3	5
7	0	0	0	0	0	0	1	1	2	3
8	0	0	0	0	0	0	0	1	1	2
9	0	0	0	0	0	0	0	0	1	1
10	0	0	0	0	0	0	0	0	0	1
$p(n)=\sum_k p_k(n)$	1	2	3	5	7	11	15	22	30	42

Суммируя числа $p_k(n)$ в n-ой колонке, мы получаем количество разбиений $p(n)$ числа n, или количество распределений n одинаковых предметов по n одинаковым ячейкам, часть которых может оказаться пустыми.

Свойства чисел разбиений $p(n)$, $p_k(n)$ изучались со средних веков; они важны как для комбинаторики, так и для теории чисел. Первые глубокие результаты для них получил Л. Эйлер, и дальше вплоть до последнего времени — К. Гаусс, Феррер, Дж. Сильвестр, П. Макмагон, Г. Харди, С. Рамануджан, И. Шур, Г. Радемахер, Г. Эндрюс и др. Одной из нетривиальных задач является характеристика возрастания числа $p(n)$ при больших $n \to \infty$.

В нижней строке таблицы 10.1 приведены значения $p(1)$, $p(2)$, ..., $p(10)$. Заметим, что скорость возрастания $p(n)$ тем выше, чем больше n: если $p(1) = 1$, $p(5) = 7$, $p(10) = 42$, то $p(20) = 627$, $p(30) = 5604$, $p(40) = 37338$, $p(50) = 204226$, $p(60) = 966467$, $p(70) = 4087968$, $p(80) = 15796476$, $p(90) = 56634173$, $p(100) = 190569292$, $p(200) = 3972999029388$.

Асимптотически при $n \to \infty$ последовательность чисел $p(n)$ ($n = 1, 2, 3$ ...) эквивалентна следующей последовательности чисел g_n:

$$g_n = \frac{1}{4\sqrt{3}} e^{\pi\sqrt{\frac{2}{3}}\sqrt{n}} \sim p(n): \lim_{n\to\infty} \frac{p(n)}{g_n} = 1. \qquad (10.42)$$

Таким образом, число $\frac{1}{4\sqrt{3}}(e^{\pi\sqrt{\frac{2}{3}}})^{\sqrt{n}}$ есть приближенное значение натурального числа $p(n)$ для большого n. Более точные приближения для $p(n)$ нашли Г. Харди и С. Рамануджан (индийский математический гений-самоучка): учет всего 8 членов в их формуле приближенного вычисления значения $p(200) = 3972999029388$ — очень большого числа — имеет ошибку всего 0,004.

Большинство качественных заключений относительно чисел $p_k(n)$, $p(n)$ возможно получить с помощью наглядных представлений разбиений, названных диаграммами Ферре (или Феррера).

Диаграмма Ферре разбиения (10.36) есть n точек, расположенных строками одна под другой так, что первая строка содержит n_1 точек, вторая строка — n_2 точек, ..., и в конце концов k-ая строка содержит n_k точек. Например, разбиение

$$15 = 5 + 5 + 3 + 2 \qquad (10.43)$$

отображается диаграммой Ферре на рис. 10.7.

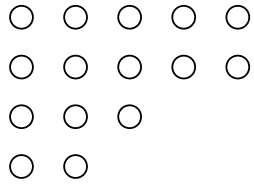


Рис. 10.7. Диаграмма Ферре разбиения (10.43)

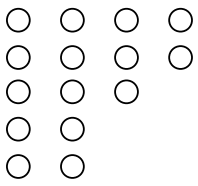


Рис. 10.8. Диаграмма Ферре разбиения (10.44)

Каждому разбиению числа n однозначно отвечает сопряженное разбиение этого числа, которое определяется транспозицией диаграммы Ферре (преобразованием строк в колонки и колонок в строки). Например, транспозиция диаграммы рис. 10.7 есть диаграмма рис. 10.8, которая определяет разбивания

$$15 = 4 + 4 + 3 + 2 + 2. \qquad (10.44)$$

Понятно, что для разбиения на k частей сопряженное разбиение состоит из частей, которые не превышают число k, и хотя бы одна часть равняется k (для разбиения (10.43) $k = 4$).

Это наблюдение доказывает такое утверждение.

Теорема 10.1

Количество разбиений числа n на k частей равняется количеству разбиений числа n с наибольшей частью, которая равняется k.

Задания

Доказать, что количество разбиений числа n, в которых части не превышают k (возможно, не равняясь k ни разу), совпадает с количеством разбиений числа $n + k$ на k частей (любых), то есть совпадает с $p_k^{(n+k)}$.

С помощью диаграмм Ферре доказываются следующие интересные утверждения.

Теорема 10.2

Количество разбиений числа n на части, из которых каждые две разные, равняется количеству разбиений числа n на нечетные части.

Теорема 10.3

*Количество самосопряжених разбиений числа n (**разбиение самосопряженное**, если оно совпадает со своим сопряженным) равняется количеству таких разбиений числа n на нечетные части, в которых каждые две части разные.*

Теорема 10.4 *(Л. Эйлер)*

Количество разбиений числа n на не более чем k частей равняется количеству разбиений числа $n + \dfrac{k(k+1)}{2}$ *на k неравных частей.*

Теорема 10.5

Количество разбиений числа n на нечетные части равняется количеству разбиений числа n, в которых каждое слагаемое, кроме наибольшего, присутствует четное количество раз, а наибольшее слагаемое — нечетное количество раз.

10.8. Производящие функции

Понятие о производящих функциях

Пусть дана некоторая последовательность чисел $a_0, a_1, a_2, \ldots, a_n, \ldots$. Если степенной ряд

$$a_0 + a_1x + a_2x^2 + \ldots + a_nx^n + \ldots \qquad (10.45)$$

совпадает при некоторых x с функцией $f(x)$, то $f(x)$ называется ***производящей функцией*** для последовательности $a_0, a_1, a_2, \ldots, a_n, \ldots$.

С последовательностью $a_0, a_1, a_2, \ldots, a_n, \ldots$ можно связать еще один степенной ряд:

$$a_0 + a_1x + a_2\frac{x^2}{2!} + \ldots + a_n\frac{x^n}{n!} + \ldots \qquad (10.46)$$

Если этот ряд совпадает с функцией $\varphi(x)$, то $\varphi(x)$ называется ***экспоненциальной производящей функцией*** для последовательности $a_0, a_1, a_2, \ldots, a_n, \ldots$. Это название поясняется тем, что для последовательности $1, a, a^2, \ldots a^n, \ldots$ экспоненциальной производящей функцией (10.46) является экспонента e^{ax}:

$$e^{ax} = 1 + ax + \frac{a^2x^2}{2!} + \ldots + \frac{a^nx^n}{n!} + \ldots \qquad (10.47)$$

Для той же последовательности обычная производящая функция есть $\dfrac{1}{1-ax}$:

$$(1 - ax)^{-1} = 1 + ax + a^2x^2 + \ldots + a^nx^n + \ldots \qquad (10.48)$$

Действительно, если $q = ax$, $|q| < 1$, то есть $|x| < \dfrac{1}{|a|}$, тогда по формуле суммы бесконечно убывающей геометрической прогрессии получаем (10.48):

$$\frac{1}{1-q} = 1 + q + q^2 + \ldots = 1 + ax + a^2x^2 + \ldots$$

В частности, если $a = 1$, тогда

$$\frac{1}{1-x} = 1 + x + x^2 + \ldots + x^n + \ldots, \quad |x| < 1. \qquad (10.49)$$

Дифференцирование (10.49) приводит к равенству

$$\frac{1}{(1-x)^2} = 1 + 2x + 3x^2 + \ldots + nx^{n-1} + \ldots, \quad |x| < 1, \qquad (10.50)$$

а умножение (10.50) на x дает

$$\frac{x}{(1-x)^2} = x + 2x^2 + 3x^3 + \ldots + nx^n + \ldots, \quad |x| < 1. \quad (10.51).$$

Таким образом, $\frac{1}{1-x}$ есть производящая функция для последовательности 1, 1, 1, 1, ... (10.49), $\frac{1}{(1-x)^2}$ — производящая функция для последовательности чисел натурального ряда 1, 2, 3, 4, ... (10.50), $\frac{x}{(1-x)^2}$ — производящая функция для последовательности 0, 1, 2, 3, ... (10.51).

Функцию $f(x)$, которая имеет производные сколь угодно высокого порядка при $t = 0$, можно считать экспоненциальной производящей функцией для последовательности своих производных $f(0)$, $f'(0)$, $f''(0)$, ..., $f^{(n)}(0)$.... Действительно, мы имеем разложение в ряд Тейлора:

$$f(x) = f(0) + f'(0)x + f''(0)\frac{x^2}{2!} + \ldots + f^{(n)}(0)\frac{x^n}{n!} + \ldots$$

Рассмотрим производящие функции для последовательностей, связанных с комбинаторными задачами.

Операции над последовательностями и производящими функциями

Благодаря последним примерам мы можем предположить, что операциям над производящими функциями должны отвечать определенные операции над соответствующими последовательностями. Действительно, пусть $f(x)$ есть производящая функция последовательности a_0, a_1, a_2, ..., и далее $g(x)$ — производящая функция последовательности b_0, b_1, b_2, ..., то есть

$$f(x) = \sum_{k=0}^{\infty} a_k x^k, \quad g(x) = \sum_{k=0}^{\infty} b_k x^k.$$

Тогда линейная комбинация функций $\alpha f(x) + \beta g(x) = \varphi(x)$ есть производящая функция последовательности $\{c_k = \alpha a_k + \beta b_k\}_{k=0}^{\infty}$, которая получается как соответствующая линейная комбинация последовательностей $\{a_k\}$, $\{b_k\}$:

$$\varphi(x) = \sum_{k=0}^{\infty} c_k x^k.$$

Обозначенная связь между операциями в классе производящих функций и в классе последовательностей символически можно записать так: если $f(x) \sim \{a_k\}$, $g(x) \sim \{b_k\}$, тогда

$$\alpha f(x) + \beta g(x) \sim \alpha\{a_k\} + \beta\{b_k\}. \qquad (10.52).$$

Введем для последовательностей бинарную операцию свертки по правилу:

$$\{a_k\} * \{b_k\}_0^\infty = \{d_k\}_{k=0}^\infty, \qquad d_k = a_0 b_k + a_1 b_{k-1} + \ldots + a_k b_0.$$

Из правила умножения степенных рядов имеем, что свертке двух последовательностей отвечает произведение их производящих функций:

$$f(x)\, g(x) \sim \{a_k\} * \{b_k\}. \qquad (10.53).$$

Соответствия (10.52), (10.53) между последовательностями и их производящими функциями оказываются эффективными и полезными в комбинаторных вычислениях. Как образец приведем формулу для суммы квадратов $1 + 2^2 + \ldots + n^n$ первых n чисел натурального ряда.

Бесконечная последовательность квадратов $\{n^2\}_{n=1}^\infty$ имеет производящую функцию $\dfrac{d}{dx}\dfrac{x}{(1-x)^2}$; в этом можно убедиться с помощью дифференцирования равенства (10.51). Поэтому функция $x\dfrac{d}{dx}\dfrac{x}{(1-x)^2}$ отвечает последовательности $\{n^2\}_{n=0}^\infty$ с первым членом 0, так как, если $f(x) \sim \{a_0, a_1, a_2, \ldots\}$, то $xf(x) \sim \{0, a_0, a_1, a_2, \ldots\}$. Поскольку $\dfrac{1}{1-x} \sim \{1, 1, 1, \ldots\}$ (см. (10.49)), то

$$\frac{1}{1-x} x \frac{d}{dx}\frac{1}{(1-x)^2} \sim \{1,1,1,\ldots\} * \{n^2\}_{n=0}^\infty = \left\{\alpha_n = \sum_{k=0}^n k^2\right\}_{n=0}^\infty.$$

Таким образом, искомые числа $\alpha_n = 1 + 2^2 + \ldots + n^2$ должны быть коэффициентами степенного ряда для функции

$$\frac{x}{1-x}\frac{d}{dx}\frac{x}{(1-x)^2} = \frac{x(x+1)}{(1-x)^4} = \sum_{n=0}^\infty \alpha_n x^n, \quad \alpha_0 = 0.$$

Сначала получим разложение функции $\dfrac{1}{(1-x)^4}$ в степенной ряд. Коэффициент ряда равняется

$$a_k = \frac{1\cdot d^k}{k!\,dx^k}(1-x)^{-4}\bigg|_{x=0} = \frac{1}{k!}4(4+1)\cdot\ldots\cdot(4+k-1)(1-x)^{-4-k}\bigg|_{x=0} =$$

$$= \frac{1}{k!}[4]^k(1-x)^{-4-k}\big|_{x=0} = \frac{[4]^k}{k!},$$

где использованы обозначения

$$[m]^k = m(m+1)\ldots(m+k-1). \qquad (10.54)$$

Отсюда получаем последовательно

$$\frac{x(x+1)}{(1-x)^4} = (x^2+x)(1-x)^{-4} = (x+x^2)\sum_{k=0}^{\infty}\frac{[4]^k}{k!}x^k =$$

$$= \sum_{n=1}^{\infty}\frac{[4]^{n-1}}{(n-1)!}x^n + \sum_{n=2}^{\infty}\frac{[4]^{n-2}}{(n-2)!}x^n = \sum_{n=1}^{\infty}\alpha_n x^n,$$

где

$$\alpha_1 = \frac{[4]^0}{0!} = 1; \quad \alpha_n = \frac{[4]^{n-1}}{(n-1)!} + \frac{[4]^{n-2}}{(n-2)!},\ n \geq 2.$$

Учитывая значения $[4]^{n-1}$, $[4]^{n-2}$ согласно (10.54), сразу получаем, что

$$\alpha_n = \frac{n(n+1)(2n+1)}{1\cdot 2\cdot 3}.$$

Поскольку α_n есть сумма квадратов, искомая формула имеет вид:

$$1^2 + 2^2 + \ldots + n^2 = \frac{1}{6}n(n+1)(2n+1). \qquad (10.55)$$

Замечание. Свертка произвольной последовательности $\{a_n\}_0^\infty$ с последовательностью $\{1, 1, 1, \ldots\}$ равняется $\{a_n\}_0^\infty * \{1, 1, 1, \ldots\} = \{a_0, a_0 + a_1, a_0 + a_1 + a_2, \ldots\}$, поэтому последовательность $\{1, 1, 1, \ldots\}$ носит название ***сумматора***. Напомним, что сумматор имеет производящую функцию $\frac{1}{1-x}$ (10.49) и что свойствами сумматора мы воспользовались выше для суммирования квадратов $1^2 + 2^2 + \ldots + n^2$. При суммировании конечных последовательностей $\{b_0, b_1, b_2, \ldots, b_n, 0, 0, 0 \ldots\}$ элементы свертки стабилизируются, начиная с определенного номера. В частности, для функции

$$(1 - 2x^3 + 3x^7 + 4x^{25} + 5x^{45})\frac{1}{1-x}$$

коэффициенты степенного ряда при x^{45}, x^{46}, ... равняются числу 11(= 1 − 2 + 3 + 4 + 5), коэффициенты при x^7, x^8, ..., x^{24} равняются 2(= 1 − 2 + 3).

Производящие функции сочетаний

Найдем производящую функцию для последовательности

$$\{C_n^0, C_n^1, C_n^2, \ldots, C_n^n, 0, 0, 0, \ldots\}. \qquad (10.56)$$

Образуем для переменных x, x_1, x_2, ..., x_n многочлен

$$(1 + x_1x)(1 + x_2x)...(1 + x_nx) = 1 + (x_1 + x_2 + ... + x_n)x + \\ + (x_1x_2 + x_1x_3 + ... + x_{n-1}x_n)x^2 + ... + (x_1x_2 ... x_n)x^n. \quad (10.57)$$

Понятно, что коэффициент перед $x^k (0 \le k \le n)$ имеет вид суммы всех k-произведений из элементов n-множества $X = \{x_1, x_2, ..., x_n\}$. Поскольку любое k-произведение в сумме взаимно однозначно отвечает определенному k-сочетанию из n-множества X, то количество слагаемых (k-произведений) в сумме перед x^k равняется именно количеству k-сочетаний (без повторений) из n-множества, то есть C_n^k.

Выражение (10.57) содержит явный перечень всех индивидуальных k-сочетаний. Но если нам нужно зафиксировать их общее количество C_n^k, положим в (10.57) $x_1 = x_2 = ... x_n = 1$ и получим производящую функцию для последовательности чисел сочетаний (10.7):

$$(1 + x)^n = \sum_{k=0}^{n} C_n^k x^k. \quad (10.58)$$

Конечно, это другой взгляд на хорошо известную биномиальную формулу. Оставаясь на новом пути, дифференцируем равенство (10.58) k раз, положим после этого $x = 0$ и получим:

$n(n - 1) ... (n - k + 1) = C_n^k k(k - 1)... \cdot 2 \cdot 1$, откуда

$$C_n^k = \frac{n(n-1)\ldots(n-k+1)}{k!} = \frac{n!}{k!(n-k)!} \text{ (см. (10.7) п. 10.3).}$$

Получим теперь производящую функцию для последовательности

$$\left\{\overline{C_n^0}, \overline{C_n^1}, \overline{C_n^2}, \ldots, \overline{C_n^k}, \ldots, \overline{C_n^{n+1}}, \ldots\right\}, \quad (10.59)$$

где $\overline{C_n^k}$ — количество k-сочетаний с повторениями без ограничений из элементов n типов. Обобщая многочлен (10.57), для изображения индивидуальных k-сочетаний из n элементов с повторениями образуем n-произведение формальных рядов с такими же переменными x, x_1, x_2, ..., x_n:

$$(1 + x_1x + x_1^2x^2 + \ldots)(1 + x_2x + x_2^2x^2 + \ldots)\ldots(1 + x_nx + x_n^2x^2 + \ldots) = \\ = 1 + (x_1 + x_2 + \ldots + x_n)x + (x_1x_2 + \ldots + x_{n-1}x_n + x_1^2 + x_2^2 + \ldots + x_n^2)x^2 + \ldots + \\ + (x_1x_2\ldots x_k + x_1^2x_2\ldots x_{k-1} + \ldots + x_n^k)x^k + \ldots \quad (10.60)$$

В сумме, которая выделена скобками перед x^k, каждое слагаемое (k-произведение) изображает соответствующее индивидуальное

k-сочетание с повторениями. Чтобы получить общее количество таких k-сочетаний $\overline{C_n^k}$, положим в (10.60) $x_1 = x_2 = \ldots = x_n = 1$:

$$(1 + x + x^2 + \ldots)^n = \sum_{k=0}^{\infty} \overline{C_n^k} x^k, \quad \overline{C_n^0} = 1.$$

Заменим ряд $\sum x^k$ его суммой (10.49) и получим:

$$\frac{1}{(1-x)^n} = \sum_{k=0}^{\infty} \overline{C_n^k} x^k, \quad |x| < 1. \tag{10.61}$$

Таким образом, производящей для последовательности (10.59) есть функция $(1 - x)^{-n}$. По общему правилу k-й коэффициент $\overline{C_n^k}$ степенного ряда (10.61) может быть вычислен дифференцированием левой части:

$$\overline{C_n^k} = \frac{1}{k!} \frac{d^k}{dx^k} (1-x)^{-n} \Big|_{x=0} = \frac{(-n)(-n-1)\ldots(-n-k+1)(-1)^k}{k!},$$

$$\overline{C_n^k} = \frac{n(n+1)\ldots(n+k-1)}{k!} = \frac{(n+k-1)!}{(n-1)!k!} = C_{n+k-1}^k.$$

Вторично получено выражение $\overline{C_n^k} = C_{n+k-1}^k$ для количества k-сочетаний с повторениями через количество k-сочетаний без повторений, создаваемых на расширенном множестве элементов.

Аналогично строятся производящие функции для сочетаний, которые удовлетворяют разнообразным ограничениям на использование любого из элементов — индивидуальным или общим.

Например, если любой из n элементов может повторяться в каждом сочетании не более чем m раз, тогда рассматривается n-произведение многочленов

$$(1 + x_1 x + \ldots + x_1^m x^m)(1 + x_2 x + \ldots + x_2^m x^m)\ldots(1 + x_n x + \ldots + x_n^m x^m) =$$

$$= \sum_{k=0}^{mn} (x_1^{k_1} x_2^{k_2} \ldots x^{k_n} + \ldots) x^k;$$

$$0 \le k_i \le m, \quad k_1 + k_2 + \ldots + k_n = k.$$

Положим здесь $x_1 = x_2 = \ldots = x_n = 1$:

$$(1 + x + x^2 + \ldots + x^m)^n = \sum_{k \ge 0}^{mn} C(n, k; m) x^k, \tag{10.62}$$

где через $C(n; k; m)$ обозначено количество таких k-сочетаний с повторениями из элементов n типов, в которых каждый элемент может быть использован (повторен) от 0 до m раз. Производящая функция таких сочетаний есть левая часть (10.62).

Производящие функции размещений и перестановок

Для индивидуального перечня *упорядоченных выборок* (перестановок, размещений) более удобной чаще оказывается *экспоненциальная производящая функция*, так как обычная (степенная) функция трактует произведения x_1x_2 и x_2x_1 как два одинаковых.

Биномиальную формулу (10.58), где $C_n^k = \frac{A_n^k}{k!}$, то есть

$$(1+x)^n = \sum_{k=0}^{n} A_n^k \cdot \frac{x^k}{k!}, \qquad (10.63)$$

одновременно можно считать разложением в ряд экспоненциальной производящей функции $(1 + x)^n$ для последовательности $\{A_n^0, A_n^1, \ldots, A_n^n; 0, 0, 0, \ldots\}$ чисел k-размещений (или k-перестановок) n-множества. Здесь имеются в виду k-перестановки без повторений.

Если любой из n элементов в k-перестановке может повторяться от 0 до m раз, тогда экспоненциальная производящая функция для чисел $C_e(n, k; m)$ таких k-перестановок имеет вид:

$$(1 + x + \frac{x^2}{2!} + \ldots + \frac{x^m}{m!})^n = \sum_{k=0}^{mn} C_e(n, k; m)\frac{x^k}{k!}.$$

Предположим, в k-перестановке можно использовать (повторять) элемент i-го типа не более чем m_i раз, $i = 1, 2, \ldots, n$. Экспоненциальная производящая функция таких перестановок

$$\prod_{i=1}^{n}\left(1 + x + \frac{x^2}{2!} + \ldots + \frac{x^{m_i}}{m_i!}\right) —$$

многочлен степени $(m_1 + m_2 + \ldots + m_n)$.

Задания

1. Рассмотреть k-перестановки с неограниченными повторениями из n разных типов элементов. Правильна ли следующая формула для экспоненциальной производящей функции таких перестановок, и как интерпретируются коэффициенты n^k в правой части:

$$\left(1 + x + \frac{x^2}{2!} + \ldots\right)^n = \sum_{k=0}^{\infty} n^k \cdot \frac{x^k}{k!}.$$

   Иначе говоря, является ли левая часть формулы экспоненциальной производящей функцией для последовательности $\{n^k\}_{k=0}^{\infty}$?

2. Проверить, что функция $\frac{1}{\ln(1-x)}$ есть производящая (обычная) для последовательности

$$\left\{\frac{1}{k+1}\right\}_{k=0}^{\infty} = \left\{1, \frac{1}{2}, \frac{1}{3}, \ldots\right\}.$$

3. Проверить, что

$$\frac{1}{k!}(e^x - 1)^k = \sum_{n=k}^{\infty} S_n^k \cdot \frac{x^n}{n!},$$

то есть $\dfrac{(e^x-1)^k}{k!}$ — экспоненциальная производящая функция для чисел Стирлинга второго рода $\{S_n^k\}_{k=0}^{\infty}$, где $S_n^k = 0$, если $n < k$ (см. п. 10.4).

Производящие функции для разбиений чисел

Разбиение (10.36) числа n на части n_i

$$n = n_1 + n_2 + \ldots + n_k$$

иногда задается с помощью ***векторной спецификации***

$$\delta = (\delta_1, \delta_2, \ldots, \delta_n), \quad 0 \le \delta_i \le n, \tag{10.64}$$

компонента которой δ_i равняется *количеству вхождений натурального числа i в разбиение* (10.36), $i = 1, 2, \ldots, n$. Иначе говоря, неотрицательные целые числа δ_i образуют спецификацию (10.64) некоторого разбиения числа n, если

$$n = 1 \cdot \delta_1 + 2 \cdot \delta_2 + \ldots + n \cdot \delta_n = \sum_{i=1}^{n} i \cdot \delta_i. \tag{10.65}$$

Понятно, что количество k частей разбиения (10.36) равняется сумме компонент спецификации:

$$\delta_1 + \delta_2 + \ldots + \delta_n = k, \quad 1 \le k \le n. \tag{10.66}$$

Часть компонент δ_i состоит из нулей. Используется также ***мультипликативная запись спецификации***:

$$(1^{\delta_1}, 2^{\delta_2}, \ldots, n^{\delta_n}). \tag{10.67}$$

Она хорошо согласуется с записью разбиения в виде «приведенной» суммы (10.65). В отличие от векторной спецификации (10.64), мультипликативную запись можно сокращать за счет пар i^{δ_i}, в которых $\delta_i = 0$. Например, для разбиения $7 = 2 + 2 + 1 + 1 + 1$ приведенное разбиение (10.65) есть $7 = 1 \cdot 3 + 2 \cdot 2 = 1 \cdot \delta_1 + 2 \cdot \delta_2$, спецификация (10.64) имеет вид $\delta = (3, 2, 0, 0, 0, 0, 0)$, а мультипликативная запись спецификации более удобна: $(1^3, 2^2)$.

Соответственно для разбиения $7 = 5 + 1 + 1$ имеем $\delta = (2, 0, 0, 0, 1, 0, 0)$ и $(1^2, 5^1)$. Если $7 = 7$, тогда $\delta = (0, 0, 0, 0, 0, 0, 1)$ и мультипликативная запись есть (7^1).

Напомним, что $p_k(n)$ обозначает количество k-разбиений числа n (разбиений (10.36) с k членами), $p(n)$ — количество любых разбиений числа n. Через $p(n, n_i \le r)$ обозначим количество разбиений числа n на части (слагаемые) n_i, которые *не превышают* r. Имея ввиду тривиальное «разбиение» $0 = 0$ с тривиальной «спецификацией» $\delta_i = 0$ $(\forall i)$, договоримся, что $p(0, n_i \le r) = 1$. Рассмотрим формальное произведение двух рядов

$$(1 + x^{1\cdot 1} + x^{1\cdot 2} + x^{1\cdot 3} + x^{1\cdot 4} + \ldots)(1 + x^{2\cdot 1} + x^{2\cdot 2} + x^{2\cdot 3} + \ldots) =$$
$$= 1 + 1 \cdot (x^{1\cdot 1}) + (x^{1\cdot 2} + x^{2\cdot 1}) + (x^{1\cdot 3} + x^{1\cdot 1 + 2\cdot 1}) + (x^{1\cdot 4} + x^{2\cdot 2} +$$
$$+ x^{1\cdot 2 + 2\cdot 1}) + \ldots = 1 + 1 \cdot x + 2x^2 + 2x^3 + 3x^4 + 3x^5 + \ldots =$$
$$= p(0, n_i \le 2) + p(1, n_i \le 2) \cdot x + p(2, n_i \le 2) \cdot x^2 +$$
$$+ p(3, n_i \le 2) \cdot x^3 + p(4, n_i \le 2) \cdot x^4 + \ldots + p(n, n_i \le 2) \cdot x^n + \ldots$$

Действительно, (1^1) — единственная мультипликативная спецификация разбиения числа 1; (1^2) и (2^1) — все мультипликативные спецификации разбиений числа 2; (1^3) и $(1^1, 2^1)$ — спецификации всех разбиений числа 3 вида $3 = 1 + 1 + 1$, $3 = 2 + 1$, такие, что их части не превышают числа $r = 2$; (1^4), (2^2), $(1^2, 2^1)$ — спецификации всех тех разбиений числа 4 вида $4 = 1 + 1 + 1 + 1$, $4 = 2 + 2$, $4 = 2 + 1 + 1$, части которых не превышают числа 2. Поэтому сумма $f(x)$ ряда

$$1 + 1 \cdot x + 2x^2 + 2x^3 + 3x^4 + 3x^5 + \ldots = \sum_{n=0}^{\infty} p(n, n_i \le 2) \cdot x^n = f(x)$$

должна быть производящей функцией последовательности

$$(1, 1, 2, 2, 3, 3, \ldots) = \{p(0, n_i \le 2),\ p(1, n_i \le 2),$$
$$p(2, n_i \le 2), \ldots, p(n, n_i \le 2), \ldots\}. \qquad (10.68)$$

Заметим, что

$$\frac{1}{1-x^2} = 1 + x^{2\cdot 1} + x^{2\cdot 2} + x^{2\cdot 3} + \ldots, \ |x| < 1$$

есть производящая функция последовательности

$$(1, 0, 1, 0, 1, 0, 1, 0, \ldots).$$

Поскольку свертка этой последовательности с сумматором

$$(1, 1, 1, 1, \ldots) * (1, 0, 1, 0, 1, 0, \ldots)$$

совпадает с последовательностью (10.68), то последовательность (10.68) имеет производящую функцию

$$f(x) = \frac{1}{(1-x)(1-x^2)} = \sum_{n=0}^{\infty} p(n,\, n_i \le 2) \cdot x^n.$$

Для произвольного натурального r, которое ограничивает части разбиений, соответственно имеем производящую функцию

$$\frac{1}{(1-x^1)(1-x^2)\ldots(1-x^r)} = \sum_{n=0}^{\infty} p(n,\, n_i \le r) \cdot x^n. \qquad (10.69)$$

С помощью символа произведения производящую функцию (10.69) записывают как

$$\prod_{k=1}^{r} \frac{1}{1-x^k}.$$

Если в (10.69) $r \to \infty$, тогда ограничение на части разбиений n_i исчезает, и следует ожидать, что

$$\prod_{k=1}^{\infty} \frac{1}{1-x^k} = \sum_{n=0}^{\infty} p(n) \cdot x^n \qquad (10.70)$$

есть производящая функция для последовательности чисел разбиений $\{p(n)\}_{n=1}^{\infty}$. Это утверждение составляет содержание известной теоремы комбинаторного анализа. Непосредственное доказательство формулы (10.70) получается путем последовательной записи членов степенного ряда, который формально равняется бесконечному произведению

$$\prod_{k=1}^{\infty} (1 + x^{k\cdot 1} + x^{k\cdot 2} + x^{k\cdot 3} + \ldots) = 1 + a_1 \cdot x + a_2 \cdot x^2 + \ldots$$

рядов $\frac{1}{1-x^k} = 1 + x^{k\cdot 1} + x^{k\cdot 2} + x^{k\cdot 3} + \ldots$, $|x| < 1$, аналогично тому, как это было сделано выше для произведения двух рядов.

Задания

1. Через $p(n, dif)$ обозначим количество разбиений числа n на попарно разные части.
   Проверить, что производящая функция последовательности $\{p(n, dif)\}_{n=0}^{\infty}$ есть:
   $$R(x) \equiv (1+x)(1+x^2)(1+x^3)\ldots(1+x^k)\ldots = \sum_{n=0}^{\infty} p(n, dif) x^n.$$
2. Производящая функция для чисел разбиений на нечетные части равняется:
   $$N(x) \equiv \frac{1}{1-x} \cdot \frac{1}{1-x^3} \cdot \frac{1}{1-x^5} \cdot \ldots \cdot \frac{1}{(1-x^{2k-1})} \cdots$$

Замечание. Производящие функции $R(x)$ и $N(x)$ совпадают. Действительно, воспользуемся соотношениями:

$$1 + x^k = \frac{1 - x^{2k}}{1 - x^k}, \quad k = 1, 2, \ldots$$

Имеем

$$R(x) = \frac{1 - x^2}{1 - x} \cdot \frac{1 - x^4}{1 - x^2} \cdot \frac{1 - x^6}{1 - x^3} \cdot \ldots = \frac{1}{1 - x} \cdot \frac{1}{1 - x^3} \cdot \ldots = N(x).$$

Таким образом, мы получаем доказательство теоремы 10.2 с помощью производящих функций.

10.9. Асимптотичные оценки и формулы

Вообще комбинаторные характеристики (функции) зависят от числа n элементов исходного множества, из которого осуществляются комбинаторные выборки. Асимптотические оценки и формулы характеризуют значения соответствующей комбинаторной функции при очень больших значениях n, как, например, соотношение (10.35) для чисел Фибоначчи u_n.

При записи асимптотических оценок и формул используются такие обозначения:

1) $f(x) = o(g(x))$ при $x \to \infty$ (читается «есть $f(x)$ o-малое от $g(x)$») тогда и только тогда, если $\lim_{x \to \infty} |f(x)/g(x)| = 0$;
2) $f(x) \sim g(x)$ при $x \to \infty$ (читается «$f(x)$ и $g(x)$ асимптотически равны» или эквивалентны) тогда и только тогда, когда $\lim_{x \to \infty} |f(x)/g(x)| = 1$;
3) $f(x) = O(g(x))$ при $x \to \infty$ (читается «есть $f(x)$ O-большое от $g(x)$») тогда и только тогда, если существуют константы $C > 0$, $R > 0$, такие, что $|f(x)| \le C|g(x)|$ при $|x| > R$.

Определением эквивалентности ~ мы уже пользовались для асимптотической оценки (10.35) чисел Фибоначчи u_n.

Обратимся к формуле количества D_n смещений — перестановок из n элементов, в которых ни один элемент не остается в первичном положении:

$$D_n = n!\left(1 - \frac{1}{1!} + \frac{1}{2!} + \ldots + \frac{(-1)^n}{n!}\right).$$

Сравним D_n с величиной $n!e^{-1}$, воспользовавшись разложением в ряд

$$e^{-1} = 1 - \frac{1}{1!} + \frac{1}{2!} + \ldots + \frac{(-1)^n}{n!} + \ldots$$

и оценкой $|a_1 - a_2 + a_3 - a_4 + ...| < |a_1|$ для суммы сходящегося знакопеременного ряда:

$$\left|D_n - n!e^{-1}\right| = \left|(-1)^{n+1} n!\left(\frac{1}{(n+1)!} - \frac{1}{(n+2)!} + \ldots\right)\right| =$$

$$= \left|\frac{1}{n+1} - \frac{1}{(n+1)(n+2)} + \ldots\right| \le \frac{1}{n+1} < \frac{1}{n}.$$

Таким образом, справедливы асимптотические формулы

$$D_n = n!e^{-1} + O(\tfrac{1}{n}), \quad \text{или } D_n \sim \frac{n!}{e}, \ n \to \infty.$$

Для справки дальше приводятся некоторые комбинаторные асимптотические формулы:

$$\sum_{k=1}^{n} \frac{1}{k} = \ln n + O(1), \quad n \to \infty. \tag{10.71}$$

$$\ln(n!) = \sum_{k=1}^{n} \ln k = n \ln n - n + O(\ln n), \quad n \to \infty. \tag{10.72}$$

$$\ln(n!) = (n + \frac{1}{2}) \ln n - n + \ln\sqrt{2\pi} + \frac{\theta}{12n}, \ 0 < \theta < 1, \ n \to \infty. \tag{10.73}$$

Формула (10.73) точнее формулы (10.72). Воспользовавшись тождеством $n! = e^{\ln n!}$, получаем эквивалентное формуле (10.73) соотношение (10.74):

$$n! = \sqrt{2\pi n} \cdot (\tfrac{n}{e})^n \cdot e^{\frac{\theta}{12n}}, \quad 0 < \theta < 1, \ n \to \infty. \tag{10.74}$$

Факториальная функция $\Gamma(n + 1) = n!$ очень важна, поэтому для нее получено много аппроксимаций и формул приближения. Формула (10.74) часто носит название ***формулы Стирлинга***. Приведем еще более точную ***«аппроксимацию Стирлинга»***:

$$n! = \sqrt{2\pi n} \cdot (\tfrac{n}{e})^n \cdot e^{\frac{1}{12n} - \frac{1}{360n^3} + \frac{q}{1260n^5}}, \quad 0 < q < 1. \tag{10.75}$$

Если в формуле (10.75) отвергнуть соответствующие прибавления в показателе степени экспоненты, сразу получим довольно тонкие оценки значений факториальной функции при $n \to \infty$:

$$\sqrt{2\pi n} \cdot (\tfrac{n}{e})^n \cdot e^{\frac{1}{12n} - \frac{1}{360n^3}} \le n! \le \sqrt{2\pi n} \cdot (\tfrac{n}{e})^n \cdot e^{\frac{1}{12n}}. \tag{10.76}$$

Для чисел Стирлинга второго рода S_n^k (10.21) выполняется асимптотическая формула:

$$S_n^k \sim \frac{k^n}{k!}, \quad n \to \infty. \tag{10.77}$$

В заключение напомним, что асимптотическое поведение числа $p(n)$ всех разбиений числа n при условии $n \to \infty$ рассматривалось выше в п. 10.7 (см. (10.42)).

Задания

1. Получить формулу (10.74), исходя из формулы (10.75) или оценок (10.76).
2. Получить формулу (10.72) как следствие формулы (10.73).
3. Доказать, что из любой формулы (10.72), (10.73), (10.74), (10.75) или оценок (10.76) вытекает эквивалентность факториальной функции:
$$n! \sim \sqrt{2\pi n}\left(\frac{n}{e}\right)^n, \quad n \to \infty.$$
4. Доказать (либо проверить для каких-нибудь значений n) оценку
$$n^{n/2} \le n! \le \frac{(n+1)^n}{2^n}, \; n > 2.$$
5. Доказать, что
$$\sum_{k=1}^{n} k^2 = O(n^3), \quad n \to \infty,$$
точнее
$$\sum_{k=1}^{n} k^2 \approx \frac{1}{3}n^3, \; n \to \infty.$$
6. Сравнить возрастание последовательностей $\{a_n\}_{n=1}^{\infty}$, $\{b_n\}_{n=1}^{\infty}$, где $a_n = n^{\ln n}$, $b_n = (\ln n)^n$, $n \to \infty$.
7. Доказать, что
$$1 + \frac{2}{n} + O(n^{-2}) = (1 + \frac{2}{n})(1 + O(n^{-2})), \; n \to \infty.$$
8. С помощью формулы Стирлинга доказать:
$$C_{2n}^n \sim \frac{1}{\sqrt{\pi n}} 4^n, \quad n \to \infty.$$
9. Доказать неравенства $(2n)! < (n(n+1))^n$, $\left(\frac{n}{3}\right)^n < n!$.

Комбинаторные задания для контроля знаний и самостоятельной работы

1. Из Киева к Чернигову можно добраться пароходом, поездом, автобусом, самолетом; из Чернигова в Новгород-Северскиий — пароходом или автобусом. Сколькими способами можно осуществить путешествие по маршруту Киев — Чернигов — Новгород-Северский?
2. Сколько четырехзначных чисел можно составить из цифр 1, 2, 3, 4, 5, если:
   а) ни одна из цифр не повторяется больше одного раза;
   б) цифры могут повторяться;

в) числа должны быть нечетными (цифры могут повторяться)?

3. В корзине 12 яблок и 10 апельсинов. Брат выбирает яблоко или апельсин, после чего сестра выбирает из фруктов, которые остались, и яблоко, и апельсин. Сколько возможностей таких выборов? При каком выборе брата сестра имеет больше возможностей выбора?
4. Есть 5 видов конвертов без марок и 4 вида марок. Сколькими способами можно выбрать конверт и марку для письма?
5. Сколькими способами можно выбрать гласную и согласную в слове «*паркет*»?
   Ответ: $n = 8$.
6. Сколькими способами можно указать на шахматной доске два квадрата — белый и черный? Решите эту задачу, если нет ограничений на цвет квадрата. Решите ее, если нужно выбрать 2 белых квадрата.
7. Сколькими способами можно выбрать на шахматной доске белый и черный квадраты, которые не лежат на одной горизонтали или на одной вертикали?
8. Из 3 экземпляров учебников алгебры, 7 геометрии и 6 физики нужно выбрать комплект, который содержит по одному учебнику каждого предмета. Сколькими способами это можно сделать?
   Ответ: $n = 3 \cdot 7 \cdot 6 = 126$.
9. Докажите, что 77 телефонов нельзя связать между собой так, чтобы каждый из них был связан ровно с 15 другими.
10. Сколько надо взять элементов, чтобы число перестановок, образованных из них, равнялось 5040?
    Ответ: $n = 7$ ($7! = 5040$).
11. В одиннадцатом классе 35 учеников. Они обменялись фотографиями. Сколько всего фотографий было роздано?
    Решение. Каждый ученик получит 34 фотографии. Поэтому $n = 35 \cdot 34$.
12. Сколько разных прямых можно провести через 10 точек плоскости, из которых никакие 3 не лежат на одной прямой?
13. Доказать, что среди любых 5 грибов в лесу, из которых никакие 3 гриба не расположены на одной прямой, всегда можно найти 4 таких, которые являются вершинами выпуклого четырехугольника.
14. На плоскости взяты 9 точек, расположенных в виде квадрата 3×3. Сколько существует треугольников, у которых одна вершина находится в фиксированной точке A, а две другие — среди других 8 точек?
15. Некоторая комиссия собиралась 40 раз. Каждый раз на заседаниях было по 10 человек, причем 2 ее членов не были присутствующими более одного раза. Доказать, что количество членов комиссии больше 60.
16. В некотором учреждении 25 сотрудников. Доказать, что из них нельзя составить больше 30 комиссий по 5 человек в каждой так, чтобы любые 2 комиссии не имели более одного общего члена.

17. Двое играют в такую игру: не увидев номера автомашины, которая приближается, первый «берет» две любых цифры номера (например, первую и третью или вторую и четвертую), а второму достанутся две другие; если номер становится известным, оба игрока составляют цифры и выиграет тот, у которого в сумме цифр единиц больше. Сколько среди номеров от 0001 до 9999 таких чисел, что игра заканчивается вничью независимо от выбора, сделанного первым игроком?
18. Для окраски одной грани кубика надо 5 с. За какое наименьшее время 3 человека могут покрасить 188 кубиков? (Предполагается, что два человека не могут вместе красить один кубик).
19. В соревнованиях по гимнастике две команды имели одинаковое количество участников. В итоге общая сумма баллов, полученных всеми участниками, равнялась 156. Сколько было участников, если каждый из них получил оценку 8 или 9 баллов?
20. Каких чисел больше среди первого миллиона: тех, в записи которых встречается 1, или тех, в записи которых ее нет?
    Решение. Подсчитаем количество чисел от 0 до 999999, в записи которых нет единиц, то есть сколько можно составить чисел шести знаков из цифр 0, 2, 3, 4, ..., 9 (если число имеет меньше шести цифр, договоримся дописывать слева нули). На первом месте в таком числе может стоять любая из девяти цифр, к любой из них можно приписать по правую сторону любую из тех же девяти: 0, 2, 3, 4, ..., 9; итак, получим 81 двузначное число с 0, 2, 3, 4, ..., 9. Продолжая, получим 9^6 шестизначных чисел, из них надо исключить 000000. Таким образом, показано, что среди первого миллиона существуют $9^6 - 1$ чисел, в записи которых нет единиц, то есть $9^6 - 1 = 531371$, что больше остальных чисел в первом миллионе.
21. За пересылку бандероли следует уплатить 18 грн. Сколькими способами можно уплатить ее марками стоимостью в 4, 6, 10 грн., если два способа, которые отличаются порядком марок, считаются разными?
    Ответ: марки можно наклеивать 8-ю способами:

    1) (10, 4, 4); 3) (4, 4, 10); 5) (4, 6, 4, 4); 7) (4, 4, 4, 6);

    2) (4, 10, 4); 4) (6, 4, 4, 4); 6) (4, 4, 6, 4); 8) (6, 6, 6).

22. За круглый стол садится $n (n > 2)$ человек. Два расположения за столом будем считать совпадающими, если каждый человек имеет одних и тех же соседей в обоих случаях. Сколько существует способов сесть за стол?
23. Сколькими способами можно посадить за круглый стол n мужчин и n женщин таким образом, чтобы никакие два лица одного пола не сидели рядом?
24. Сколько можно составить перестановок из n элементов, в которых заданные m элементы не расположены рядом в любом порядке?
25. Имеем набор из 16 карточек. На четырех написана буква A, на четырех — $Б$, на четырех — B, на четырех — $Г$. Сколько разных слов

можно получить, выбирая из набора 4 карточки и располагая их в некотором порядке?

26. Сколькими способами можно обтянуть 6 стульев тканью, если она имеется шести разных цветов, и все стулья должны быть разноцветными?
Ответ: $n = P_6 = 6! = 720$.
27. Сколькими способами могут расположиться в турнирной таблице 10 футбольных команд, если известно, что никакие 2 не набрали одинакового количества очков?
Ответ: $n = P_{10} = 10!$.
28. Сколько четырехзначных чисел можно образовать из цифр 0, 1, 2, 3, не повторяя их?
29. На собрании должны выступить 5 человек: *А*, *Б*, *В*, *Г*, *Д*. Сколькими способами можно их расположить в списке ораторов, если:
1) *Б* не должен выступать перед *А*;
2) *Б* должен выступить сразу за *А*?
30. Сколькими способами можно составить трехцветный полосатый флаг, если есть ткани пяти разных цветов? Решите ту же самую задачу при условии, когда одна полоса должна быть красной.
31. Есть 8 токарей. Сколькими способами можно поручить 3 из них изготовление трех разных деталей (по одному виду на каждого)?
Решение. Речь идет о выборе 3 токарей из 8 с дальнейшим размещением их около трех станков, которые изготовляют разные детали. Поэтому $n = A_8^3 = 336$.
32. В профактив факультета избрано 9 человек. Из них надо избрать председателя, его заместителя, секретаря и культработника. Сколькими способами это можно сделать?
Ответ: $n = A_9^4 = \dfrac{9!}{4!} = 3024$.
33. Сколькими способами можно опустить 5 писем в 11 почтовых ящиков, если в каждом из них бросается не больше одного письма?
Ответ: $n = A_{11}^5 = ?$
34. Сколько разных натуральных чисел можно образовать из цифр 0, 1, 2, 3, 4, если каждое число содержит любую из данных цифр не больше одного раза?
35. Из колоды в 52 карты вынули 10 карт. В скольких случаях среди этих карт окажется:
а) хотя бы один туз;
б) ровно один туз;
в) не меньше двух тузов;
г) ровно два туза?
36. Строительная фирма формирует бригаду из 5 рабочих. В организации 20 рабочих, в том числе 5 маляров, 4 плотника и 2 штукатура. Сколькими числом способов можно укомплектовать бригаду, чтобы она состояла из рабочих всех специальностей по одному?

37. Найдите сумму четырехзначных чисел, которые получаются с помощью всяческих перестановок цифр 1, 1, 4, 4. То же самое для 0, 0, 4, 4.
38. Сколькими способами можно составить три пары из n шахматистов?
39. Сколькими способами можно выбрать 6 карт из колоды в 52 карты таким образом, чтобы среди них были карты каждой масти?
40. Сколькими способами в игре «Спортлото» можно:
    а) выбрать пять шаров из 36;
    б) угадать 5, 4, 3 номера.
41. Сколькими способами можно разместить 12 разных деталей в 3-х ящиках?
42. Сколько существует автомобильных пятизначных номеров,
    а) составленных из цифр 2, 3, 5, 7;
    б) не содержащих цифру 8;
    в) не содержащих цифр 0 и 8?
43. Найти количество способов распределения n одинаковых шаров по: а) двум; б) трем; в) i неразличным урнам.
44. Сколькими способами можно разместить n_1 белых, n_2 черных и n_3 голубых шаров по m разным урнам?
45. В лаборатории научно-исследовательского института работает несколько человек, причем каждый из них знает хотя бы один иностранный язык, 6 — английский, 6 — немецкий, 7 — французский, 4 — английский и немецкий, 3 — немецкий и французский, 2 — французский и английский, одно лицо — все три языка. Сколько человек работает в лаборатории? Сколько из них знает лишь английский язык? Сколько лиц знает лишь один язык?
46. Сколько чисел среди первой тысячи натуральных чисел не делятся ни на 2, ни на 3, ни на 5, ни на 7?
47. Сколько шестизначных чисел можно составить из цифр 1, 2, 3, ..., 9, если любое число должно состоять из трех четных и трех нечетных цифр, причем никакие 2 цифры в числе не повторяются?
48. 20 пассажиров собираются путешествовать поездом. В кассе есть 12 билетов на нижние полки и 8 — на верхние. При этом 4 пассажира не желают ехать внизу, а 5 пассажиров — вверху. Сколькими способами их можно разместить в поезде, если:
    а) порядок размещения пассажиров как внизу, так и вверху не учитывается;
    б) порядок размещения учитывается как внизу, так и вверху;
    в) учитывается порядок размещения лишь внизу?
49. Сколько слов можно получить, переставляя буквы слов:
    а) «*парабола*»; б) «*метаморфоза*»?
50. Для премий на математической олимпиаде выделено 3 экземпляра одной книги, 4 — второй и 8 — третьей. Сколькими способами можно разделить эти премии между 30 участниками, если любому вручают не больше одной книги?

51. Абитуриент должен сдать 4 экзамена. Он считает, что для зачисления достаточно набрать 17 очков. Сколькими способами он сможет сдать экзамены, набрав не меньше 17 очков и не получив ни одной двойки?
52. Сколько слов, которые содержат не меньше одной буквы, можно составить из двух букв *А*, пяти букв *Б* и девяти букв *В*?
53. Сколько слов из пяти букв, любое из которых состоит из трех согласных и двух гласных, можно составить из букв слова «уравнение»?
54. Участники «Генуэзской лотереи» покупают билеты, на которых размещены числа от 1 до 90. На некоторых билетах указаны сразу 2, 3, 4 или 5 чисел. В день розыгрыша произвольно избирают 5 жетонов с номерами от 1 до 90. Выиграют те участники, в которых все номера на билетах будут среди выбранных на жетонах. Какая вероятность выигрыша в случае покупки билета с одним числом? С k числами $(1 \le k \le 5)$?
55. Сколькими способами можно переставить буквы слова «*обороноспособность*», чтобы две буквы «о» не шли подряд?
56. Найдите наибольший коэффициент в разложении:
    а) $(a + b + c)^{10}$;
    б) $(a + b + c + d)^{11}$.
57. Сколько существует треугольников, длины сторон которых принимают одно из таких значений: 4, 5, 6, 7 см?
58. Сколько можно построить прямоугольных параллелепипедов, длины ребер которых выражаются натуральными числами от 1 до 10?
59. Сколькими способами 9 пассажиров можно разместить в трех вагонах? Для скольких размещений в первый вагон сядут 3 из них? Для скольких размещений в каждый вагон сядут 3 из них? Для скольких размещений в один вагон сядет 4, во второй — 3, а в третий — 2 пассажиров?
60. Сколькими способами можно отобрать несколько фруктов из 7 яблок, 4 лимонов, 9 апельсинов (считаем, что фрукты одного вида не отличаются один от другого)?
61. Две команды *А* и *Б* играют серию матчей в баскетбол до тех пор, пока одна из них не получит четыре победы (ничьих нет). Сколько разных серий таких матчей может быть?
62. Сейф открывается с помощью цифрового кода, циферблат которого состоит из 100 клавиш с числами, расположенными по кругу. Для того, чтобы открыть сейф, нужно нажать на какие-то 3 клавиши, причем известно, что между любыми двумя искомыми клавишами располагается не меньше чем 10 других. Сколько комбинаций из 3 клавиш надо перепробовать, чтобы открыть сейф, если:
    а) порядок нажатия не имеет значения;
    б) порядок нажатия имеет значение?
63. Для каждого натурального n найти сумму

$$S_n = 1 \cdot 1! + 2 \cdot 2! + 3 \cdot 3! + \ldots + n \cdot n!.$$

Указание. Проверить $S_n = (n + 1)! - 1$.

64. Докажите, что $(C_n^0)^2 + (C_n^1)^2 + \ldots + (C_n^n)^2 = C_{2n}^n$.

Решение. C_{2n}^n — количество n-элементных подмножеств $2n$-элементного множества B можно подсчитать так. Разобьем множество B на два подмножества (B_1 и B_2) по n элементов в каждом. Любое из искомых n-элементных подмножеств состоит из k элементов множества B_1 и $(n - k)$ элементов множества B_2 ($k = 0, 1, 2, \ldots, n$). Для любой из C_n^k выборок k элементов множества B_1 остаток $(n - k)$ элементов из множества B_2 можно выбрать $C_n^{n-k} = C_n^k$ способами. По правилу произведения количество искомых подмножеств с k элементами множества B_1 будет $(C_n^k)^2$. Поскольку

$$k = 0, 1, 2, \ldots, n, \quad \text{то } C_{2n}^n = (C_n^0)^2 + (C_n^1)^2 + \ldots + (C_n^n)^2.$$

65. Докажите такое тождество для чисел C_m^k:

$$C_p^0 \cdot C_{n-p}^m + C_p^1 \cdot C_{n-p}^{m-1} + \ldots + C_p^k \cdot C_{n-p}^{m-k} \ldots + C_p^m \cdot C_{n-p}^0 = C_n^m.$$

Решение. C_n^m — количество m-элементных подмножеств Y множества X из n элементов. Разобьем множество X на два подмножества X_p из p элементов и X_{n-p} из остатка $n - p$ элементов. Множества Y можно строить, выбирая сначала 0 элементов множества X_p и m элементов множества X_{n-p} (что можно сделать $C_p^0 \cdot C_{n-p}^m$ способами), и так далее, выбирая k элементов множества X_p и $(m - k)$ элементов множества X_{n-p}, что можно сделать $C_p^k \cdot C_{n-p}^{m-k}$ способами. Поскольку для разных значений k будем иметь разные подмножества, а других подмножеств быть не может, то отсюда вытекает справедливость записанной выше формулы.

Тождество из задачи 65 чаще встречается в одной из следующих форм:

$$\sum_{k=-m}^{k} C_r^{m+k} C_s^{n-k} = C_{r+s}^{m+n}; \quad \sum_{p=0}^{n-k} C_n^{k+p} C_m^p = C_{m+n}^{n-k}.$$

Это тождество носит название *свертки А. Вандермонда* (конец XVIII ст.), однако в 1303 г. оно было известно Чжу Ши-дзе в Китае.

ПОСЛЕСЛОВИЕ

В одной книге трудно дать развернутое изложение всех разделов дискретной математики, рассмотренных в данном учебнике. Здесь подробно представлены отношения, булевы алгебры и функции, графы, материал которых может быть использован при выполнении индивидуальных заданий и изучении спецкурсов по соответствующим темам. Достаточно емко изложены множества, математическая логика и комбинаторика. В то же время для более углубленного изучения материала этих разделов и использования их в спецкурсах мы рекомендуем пользоваться дополнительной литературой: Куратовский К., Мостовский А. (23), Натансон И. П. (26) — по теории множеств; Клини С. (16), Мендельсон Э. (28), Колмогоров А. Н., Драгалин А. Г. (17), Чень Ч., Ли Р. (37) — по математической логике; Андерсон Дж. (1), Виленкин Н. Я. (8, 9), Липский В. (24), Риордан Дж. (30), Грэхем Р., Кнут Д., Поташник О. (13), Холл М. (36) — по комбинаторике.

Авторы вынуждены были более конспективно изложить такие разделы, как языки и грамматики, алгоритмы и автоматы. Обычно эти разделы читаются в виде отдельных дисциплин на многих специальностях в высших учебных заведениях. Можно порекомендовать целый ряд книг с прекрасным изложеним этих разделов: Ахо А., Ульман Дж. (2, 3), Капитонова Ю. В., Кривий С. Л., Летичевский О. А., Луцкий Г. М., Печорин М. К. (15), Глушков В. М. (12), Корман Т., Лейзерсон Ч., Ривест Р. (18), Кук Д., Бейз Г. (22).

В названной литературе имеется много модельных и содержательных примеров, которые полезны и поучительны. Некоторыми из них авторы с благодарностью воспользовались.

СПИСОК ЛИТЕРАТУРЫ

1. Андерсон Дж. Дискретная математика и комбинаторика.— Москва — С. Петербург — Киев.: Издат. дом «Вильямс», 2003.— 958 с.
2. Ахо А., Ульман Дж. Теория синтаксического анализа, перевода и компиляции. т. 1. Синтаксический анализ.— М.: Мир, 1978.— 612 с.
3. Ахо А., Хопкфорт Дж., Ульман Дж. Построение и анализ вычислительных алгоритмов.— М.: Мир, 1979.— 536 с.
4. Бардачев Ю. Н., Соколова Н. А., Ходаков В. Е. Основы дискретной математики.— Херсон, ХГТУ, 2000.— 356 с.
5. Белоусов А. И., Ткачев С. Б. Дискретная математика.— М.: Издательство МГТУ им. Н. Э. Баумана, 2001.— 744 с.
6. Білоус Н. В., Дудар З. В., Лєсна Н. С., Шубін І. Ю. Основи комбінаторного аналізу.— Харків: ХТУРЕ, 1999.— 96 с.
7. Бондаренко М. Ф., Білоус Н. В., Шубін І. Ю. Збірник тестових завдань з дискретної математики.— Харків: ХТУРЕ, 2000.— 156 с.
8. Виленкин Н. Я. Комбинаторика.— М.: Наука, 1969.— 328 с.
9. Виленкин Н. Я. Популярная комбинаторика.— М.: Наука, 1975.— 208 с.
10. Гаврилов Г. П., Сапоженко А. А. Сборник задач по дискретной математике.— М.: Наука, 1977.— 368 с.
11. Горбатов В. А. Основы дискретной математики.— М.: Высшая школа, 1986.— 312 с.
12. Глушков В. М. Синтез цифровых автоматов.— М.: Физматгиз, 1962.— 476 с.
13. Грэхем Р., Кнут Д., Поташник О. Конкретная математика, основание информатики.— М.: Мир, 1998.— 704 с.
14. Иванов Б. Н. Дискретная математика (Алгоритмы и программы).— М.: Лаборатория базовых знаний, 2001.— 288 с.
15. Капітонова Ю. В., Кривий С. Л., Летичевский О. А., Луцький Г. М., Печорін М. К. Основи дискретної математики.— Київ: Наукова думка, 2002.— 578 с.
16. Клини С. Математическая логика.— М.: Мир, 1973.— 480 с.
17. Колмогоров А. Н., Драгалин А. Г. Введение в математическую логику.— М.: МГУ, 1982.— 120 с.
18. Корман Т., Лейзерсон Ч., Ривест Р. Алгоритмы: построение и анализ.— М.: МЦНМО, 2001.— 960 с.
19. Криницкий Н. А. Аналитическая теория алгоритмов.— М.: Физматлит, 1994.— 352 с.
20. Кристофидес Н. Теория графов (Алгоритмический подход).— М.: Мир, 1978.— 432 с.
21. Кузнецов О. П., Адельсон-Вельский Г. М. Дискретная математика для инженера.— М.: Энергия, 1980.— 344 с.
22. Кук Д., Бейз Г. Компьютерная математика.— М.: Наука, 1990.— 384 с.

23. Куратовский К., Мостовский А. Теория множеств.— М.: Мир, 1970.— 416 с.
24. Липский В. Комбинаторика для программистов.— М.: Мир, 1998.— 214 с.
25. Мальцев А. И. Алгебраические системы.— М.: Наука, 1970.— 370 с.
26. Натансон И. П. Теория функций вещественной переменной.— М.: Наука, 1974.— 480 с.
27. Новиков Ф. А. Дискретная математика для программистов.— СПб: Питер, 2001.— 304 с.
28. Мендельсон Э. Введение в математическую логику.— М.: Наука, 1976.— 320 с.
29. Оре О. Теория графов.— М.: Наука, 1968.— 352 с.
30. Риордан Дж. Введение в комбинаторный анализ.— М.: ИИЛ, 1963.— 288 с.
31. Руткас А. Г. Введение в теорию графов.— Харьков.: «Принтал», 1993.— 63 с.
32. Уилсон Р. Введение в теорию графов.— М.: Мир, 1977.— 205 с.
33. Филлипс Д., Гарсиа-Диас А. Методы анализа сетей.— М.: Мир, 1964.— 496 с.
34. Форд Л., Фалкерсон Д. Потоки в сетях.— М.: Мир, 1966.— 276 с.
35. Харари Ф. Теория графов.— М., Мир, 1973.— 304 с.
36. Холл М. Комбинаторика.— М.: Мир, 1970.— 424 с.
37. Чень Ч., Ли Р. Математическая логика и автоматическое доказательство теорем.— М.: Наука, 1983,— 256 с.
38. Яблонский С. В. Введение в дискретную математику.— М.: Наука, 1986.— 384 с.

ПРЕДМЕТНЫЙ УКАЗАТЕЛЬ

А

Абелева группа 87
Автомат 386
– конечный 391–392
– – , диаграмма состояний 392
– – , таблица 392
– – без выхода 392
– – с выходом 392
– линейно-ограниченный 406
– Мили 394
– Мура 394, 395
– с магазинной памятью 400
Аксиомы исчисления высказываний 202
– – предикатов 227
Алгебраическая операция 73
– структура 79
Алгебра 23
– булева 97, 106
– – , двухэлементная 107
– Жегалкина 138
– логики 107
– множеств 22
– реляционная 62
Алгоритм Гомори — Ху 331–334
– Данцига 299–303
– Дейкстры 293–298
– Йоу, Даниэльсона, Дхавана 317–321
– Краскала 282
– Маркова 362–363
– – , нормальный 362–363
– перехода от произвольной формулы алгебры логики к СДНФ 134
– перехода от произвольной формулы алгебры логики к СКНФ 134
– перехода от таблицы истинности булевой функции к СДНФ 131–132
– перехода от таблицы истинности булевой функции к СКНФ 132
– «поиска в глубину» 276–280
– построения остовного дерева путем произвольного просмотра ребер 281–282
– Прима 283
– расстановки пометок 326–328
– Флойда 299–303
– Форда 293, 298–299
– Форда — Фалкерсона 326–330
Алфавит 339
– логики 235
Антирефлексивность отношения 43
Антисимметричность отношения 44
Антитранзитивность отношения 44–45
Асимметричность отношения 44
Ассоциативность 23, 106, 116, 138
Атом (элементарные высказывания) 217
Атрибуты таблицы 63

Б

Биективное отображение 57–58, 60
Биекция 27
Бинарное отношение 32
Бинарные функции 236–237
Булеан 16
Булева алгебра 97, 106, 107
– решетка 97

В

Взаимно однозначное соответствие 27
Включение двухстороннее 14
– нестрогое 15
– строгое 15
Внутреннее произведение вершин пути 318
Входная головка 386
– лента 386
Входной алфавит 386
Выводимые строки в грамматике 344
Выровненная влево грамматика 347
– вправо грамматика 347
Высказывание 186

Г

Геометрическая интерпретация множеств 18–20
Головоломка о кенигсберских мостах 239, 246
Гомоморфизм 80–81

Грамматика 342
– выровненная влево 347
– выровненная вправо 347
– контекстно-зависимая 347
– контекстно-свободная 347
– леволинейная 347
– леворекурсивная 350
– общего вида 347
– порождающая 342
– праволинейная 347
– праворекурсивная 350
– распознающая 342
– регулярная 347
– самовключающая 350
Граф 242–243
– , вершины 243
– – , степень 244
– , грань, внешняя, внутренняя 244
– – (ячейка) 244
– , диаметр 244
– , дополнение 244
– , индекс 262
– , каркас 275
– , остов 275
– , параллельные ребра 243
– , радиус 245
– , ребра 243
– , сечение (разрез) 287
– , смежные вершины 243
– – ребра 243
– , точка сочленения 244
– , хроматический класс 262
– , центр 245
– , центральные вершины 245
– – – , эксцентриситет 245
– , эйлеров 246
– абстрактный 249–250
– бихроматический 262, 263
– взвешенный 282
– гамильтонов 305
– геометрический 242, 248, 250
– геометрически двойственный 261
– дуальный (реберный) 262
– дуги 248
– конечный 251
– неориентированный 242–245, 248
– ориентированный 248–249
– отношения 34
– планарный 242
– плоский 241, 242, 244
– полный 244
– помеченный 271
– Понтрягина — Куратовского 253
– простой 244
– регулярный (однородный) 244
– связный 243
– симметрический 252
– цепи 240
– *r*-раскрашиваемый 261
– *r*-хроматический 261
График отображения 56
– функции 56
Графическое задание бинарного отношения 33
Группа 87
– абелева 87

Д

Двойственность 111–115
Двоичное слово 100
Двузначная логика 231
Дедуктивный вывод 197
Декартова степень 32
Декартово произведение 31
Декартов квадрат 32
– куб 32
Деление отношений 69–70
Дерево 240, 244, 269
– , высота 284
– – вершины 284
– , дуги 269
– , корень 274
– , лист 284
– , потомок вершины 284
– , ребра 269
– , уровень вершины 284
– бинарное (двоичное) 285
– вывода 354, 355
– корневое 274
– ориентированное 269, 283
– разбора 354
– свободное 273
– упорядоченное 285
Дефицит простого графа 314
Диаграмма Вейча 164
– Венна 18–19
– коммутативная 81–82
– Ферре разбиения 441
– Хассе 48, 50
Дизъюнктивная нормальная форма (ДНФ) 124
– – – , минимальная (МДНФ) 156

– – – , совершенная (СДНФ) 125
– – – , сокращенная 156
– – – , тупиковая 156
Дизъюнктивное поглощение 157
– склеивание неполное 157
– – полное 157
– ядро 156, 173
Дизъюнкция 103, 107, 187, 190
– элементарная 128
– *k*-значная 237
Дистрибутивность 24, 106, 117, 139
Доказательство от противного 206
Домен 63
Дополнение (отрицание) 21, 41
Достижимость вершины графа 256

Е

Единица по отношению к операции 77, 86, 87, 88
– решетки 96, 97

З

Задание булевой функции 102
– – – аналитически 102, 107–108
– – – порядковым номером 102, 104–105
– – – таблицей истинности 102–104
– множеств 11–13
– – определяющим свойством 11
– – перечислением элементов 11
– – рекурсивное 12
Задача загрузки (размещения) 268
– китайского почтальона 247
– коммивояжера 240–241, 314–317
– – гамильтонова 241, 314
– о гамильтоновом цикле 307–308
– о книгах 309
– о максимальном потоке 324–326
– о многополюсном максимальном потоке 331
– о трех колодцах 252
– раскраски стран географической карты 241, 261
– теории расписаний 268
Законы алгебры множеств 23–24
– ассоциативные 23, 96, 106, 117, 221
– дистрибутивные 24, 106, 117, 221–222
– для нуля, единицы и отрицания 106
– идемпотентности 24, 106, 117
– инволюции 24, 106
– коммутативные 23, 96, 106, 116, 221
– поглощения 96, 106
– де Моргана 24, 106, 119, 222
– для кванторов 222
Закон двойного отрицания 106, 118–119
– инволюции 24, 106
– исключенного третьего 24, 118
– противоречия 24, 118
– элиминации 24, 118
Замена связанной переменной 221, 222
Замкнутый класс 145
Замыкание множества Σ булевых функций 145
Записи infix 74, 75
– postfix 74, 75
– prefix 74
Значение истинности высказывания 186, 232

И

Идемпотентность 24, 106, 117
Иерархия Хомского 347
Изоморфизм графов 250, 254
Изоморфные структуры 82–83
Импликанта 155, 161, 166
– простая 156, 167, 171, 173
Импликация 104, 107, 108, 187, 190–191
– , обратная 104
Инверсия 102, 107
Индивидуальный символ 209
Интерпретация булевой функции 100, 104
– высказывания 189
– формулы логики предикатов 218
Инцидентность ребер и вершин графа 243
Инцидентор 249, 257
Инъективное отображение 57, 59
Истинностное значение высказывания 186
Источник 322, 330
Исчисление высказываний 201
– предикатов 227

К

Каноническая задача минимизации 155
Карта Карно 158–164
Квантор всеобщности 212, 215

– существования 212, 215
Классы эквивалентности 47–48
Класс Поста 151
Клика 264
Кликовое число (плотность) 264
Кодерево 289
Количество смещений D_n 427
– неполных смещений 427
Кольцо 91, 92, 140
– с единицей 91, 92, 140
– коммутативное 92
Коммутативная диаграмма 81–82
Коммутативность 23, 96, 106, 116, 221
Композиции числа n 438
Композиция отношений 38
Компонента (связности) 243
Конечные языки 350
Конкатенация 86, 340
Константа булева 100
– нуля (единицы) 102, 114, 138
– предметная 210
Конституента единицы 124
– нуля 128
Контекстно-зависимая грамматика (КЗ) 347
Контекстно-свободная грамматика (КС) 347
Континуум 29
Контур 249
– гамильтонов 308, 312, 314, 320, 321
– простой 249
– эйлеров 252
Конфигурация автомата 387
– – , заключительная 388
– – , начальная 387
Конъюктивная нормальная форма (КНФ) 128
– – – , совершенная (СКНФ) 129
Конъюктивное поглощение 158
– склеивание неполное 158
– – полное 158
Конъюнкция 103, 107, 187, 189
– , собственная часть 156
– , элементарная 124
– k-значная 237
Кортеж 63
Критерий минимизации 155, 156
Круги Эйлера 19
Куб булевый n-мерный 100

Л

Лексемы языка 357
Лемма о рукопожатиях 251
Линейно упорядоченное множество 50
Логика высказываний 187
– двузначная 231
– многозначная 231
– предикатов 207
– формальная 183–185
Логическое следствие 197, 219

М

Макстерм 128
Маршрут 243
– , длина 243
– замкнутый 243
– неориентированный 249, 315
– ориентированный 248, 315
Матрица инциденций 257–259
– Кирхгофа 275
– сечений графа 288
– смежности 253–256
– – модифицированная 317
– стоимостей 256
– циклов графа 287
Машина Тьюринга 402–403
Метод Квайна 166
– Квайна—Мак-Класки 168–174
– Порецкого—Блейка 174–175
Минтерм 124
Модель данных реляционная 62–72
Многозначная логика 231
Множества равномощные 27
– эквивалентные 27
Множество-степень 16
Множество 9
– бесконечное 10, 26–29
– конечное 10
– континуальное 29
– линейно упорядоченное 50
– независимое 264
– несчетное 29
– пустое 15, 16
– счетное 28
– универсальное 15
– упорядоченное 10–11
– частично упорядоченное 94
Моноид 86
Мощность 28, 29
Мультиграф 244

Н

Набор булевый 100
– – функционально полный 145, 152
– – – – в слабом смысле 153
Независимая система аксиом 203
Независимое множество 264
Непротиворечивость исчисления высказываний 203
Несократимая система булевых функций 152
Нетерминальный символ 342
Носитель алгебраической структуры 79
Нуль решетки 96

О

Область значений отношения 56
– – отображения 56
– определения отношения 49, 55
Обобщенное склеивание 174, 175
Образ отображения 56
Обратная импликация 104
– функция 57
Обратное отношение 37, 57
Обратный элемент 77, 88
Операнды 63, 74
Операторы 74, 362, 367
Оператор минимизации 368
– подстановки 367
– рекурсии 368, 369
Операции над отношениями 36–42
– алгебраические 73
– на множествах 20–22
Орграф 248, 249
Ортогональности соотношение 291
Ортогональные латинские квадраты 415
Остовный лес графа 280
Отношение
– бинарное 32
– нестрогого порядка 51
– обратное 37, 57
– полное 34
– порядка 11, 147
– – для наборов булевых констант 147
– пустое 34
– строгого порядка 51
– тождественное 34
– толерантности 51
– функциональное 54
– частичного порядка 48, 93, 95
– эквивалентности 47, 48, 83, 250
– , прямое произведение 65
– , разность 64
Отношения-операнды 63
Отображение множества 56
– биективное 57, 59
– инъективное 57, 59
– сюръективное 57, 59
Отрицание 21, 102, 107, 108, 189
– дизъюнкции 103
– импликации 103
– конъюнкции 104
– Лукашевича 235, 237
– обобщенное 236
– обратной импликации 103
– циклическое 235

П

Память автомата 387
Парадокс логический 13
Переменная булева (логическая) 100
– несущественная 101
– предметная 210
– свободная 213
– связанная 213
– фиктивная 101
Пересечение (произведение) 21
– отношений 64
Перестановки 418
– с повторениями 419
Петля 243
Подалфавит 339
Подграф 243
– , вес 282
Поддерево вершины левое, правое 285
Подмножество 14, 15, 16
Подстрока 340
Подструктура алгебраической структуры 80
Покрытие 155, 156, 161
Поле 92
Полином Жегалкина 140–143
– – , длина 141
Полная решетка 96
– система импликант 155, 156
– – функций 151–152, 236
Полное отношение 34
Полностью упорядоченное множество 50
Полнота исчисления высказывания 202
Полугруппа 86
Порождающая грамматика 342

Порядок предиката 208
Посылка (условие, антцедент) 190, 200
Поток в сети 322
– – – , аргументальная цепь 325
– – – по дуге 322
– – – чистый 322
Правило ∃-введения 228
– введения квантора всеобщности 225
– – – существования 225
– обобщения (∀-введения) 228
– отделения (Modus Ponens) 200
– переименования свободных переменных 228–229
– – связанных переменных 229
– подстановки 203–204, 205
– произведения 417
– суммы 416
– удаления квантора всеобщности 225
– – – существования 225
Правила вывода 197, 199, 202, 225, 228
Правильное рассуждение 195
Правильно построенная формула 188
Праволинейная грамматика 347
Праворекурсивные грамматики 350
Предваренная нормальная форма (ПНФ) 224
Предикат 208, 209
Предикатные аксиомы 227–228
Предметная область 208, 210
Предметные константы 210
– переменные 210
Префикс 224
Признак Дирака 310
Признаки гамильтоновости 310–314
Принцип двойственности 114
Приоритет операций в алгебре множеств 23
Проблема принадлежности 349
– пустоты 350
– четырех красок 241, 261
– эквивалентности 350
Продукция 342
Проекция отношения 67
Производящая функция 443
– – для разбиений числа n 450–452
– – перестановок 449
– – размещений 449
– – сочетаний 446
– – экспоненциальная 443
Промежуточный узел 322
Прообраз множества 56
Пропускная способность сечения 324
Прямое произведение отношений 65
Пустая строка 340, 401
Пустое отношение 34
Путь 249
– гамильтонов 308
– простой 249

Р

Равенство множеств 14, 15
Разбиение числа n 438–442
Разложение булевой функции 120–127
Размещение предметов по урнам 428–430
Размещения 418
– с повторениями 420
Разность отношений 64
Раскраски графа 260–269
Распознаватель 390, 406
Распознающая грамматка 342
Рассуждение правильное 195
Расширение алфавита 339
Регулярное выражение в алфавите 351
Реляционная алгебра 62
– модель данных 62–72
Рефлексивность отношения 42–43
Решетка 94
– дистрибутивная 96, 97
Решетка с дополнением 96
Решето Эратосфена 432

С

Свертка последовательностей 445
Свободная переменная 213
Свойства единицы 106
– нуля 106
– пустого и универсального множеств 24
Связанная переменная 213
Связывание переменной 213
Сеть 322
Сечение относительно подмножества 40
– отношения 40
Символическая логика 184
Символ предметных переменных 209
– индивидуальный 209
– нетерминальный 342
– предикатный 209
– терминальный 342
– функциональный 209
Симметричность отношения 43–44

Система S_1 204
– S_2 204–205
Склеивание обобщенное 174
Следствие (заключение, консеквент) 190
Слово двоичное 100
Сложение по модулю 77, 138, 237
Собственная часть конъюнкции 156
Совершенная нормальная форма 120
Сопряженное разбиение числа n 438–441
Сортировка вставками 371–372, 381
– слиянием 372–374, 379
Сочетания 421
– с повторениями 422
Сравнимые элементы в отношении частичного порядка 50
Сток 322
Стрелка Пирса 103, 108
Строка 339, 340
– , пустая 340
– , выводимая 344
Стэк 399, 400, 401
Сужение операции 80
Сумма графов 255
– по модулю 2 103, 138, 139
Суперпозиция 107
– булевой функции 107
Схема логическая 177–180
Сюръективное отображение 57, 59

Т

Таблица истинности булевой функции 102, 109
– Кэли 75, 88, 89, 92
Тезис Черча — Тьюринга 403
Теорема
– Брукса 265
– дедукции 205
– Дирака 310
– Ейлера 442
– Караганиса 311
– Кенига 263, 312
– – о матрицах 416
– Кирхгофа 275
– об одновременной полноте 145
– Оре 310
– о дизъюнктивном разложении булевой функции 120
– о конъюнктивном разложении булевой функции 125
– о максимальном потоке и минимальном разрезе 324
– о целочисленности 328
Теория
– Понтрягина — Куратовского 253
– Поста о функциональной полноте 151–155
– Поша 310
– Пуанкаре 289
– Флейшнера 311
– Харари, Нэш-Вильямса 311
– Хватала 310
Терм 209
Терминальный символ 349
Тождественное отношение 34
Тождественно истинная формула (тавтология) 195, 197, 202, 229
– ложная формула (противоречивая, невыполнимая) 195, 198
Транзитивность отношения 44
Треугольник Паскаля, арифметический треугольник 424
Тэта-граф 306

У

Умножение по модулю 77, 237
Унарное отношение 32
Унарные операции 74
– функции 235–236
Управляющее устройство 387

Ф

Фактор-множество 40
Формальная логика 183
Форма Бекуса — Наура (БНФ) 345
Формула 107
– биноминальная (бином Ньютона) 431
– включений и исключений 426
– нейтральная 195
– необщезначимая 195
– непротиворечивая 195
– общезначимая 195
– полиноминальная 431
– тождественно истинная 195
Формула Кэли 273
Формулы (молекулы) 188
– равносильные 108
– эквивалентные 108
Фундаментальная матрица сечений 290
– – циклов 290
– система циклов и сечений 289
Фундаментальное сечение 289

Фундаментальный цикл графа 290
Функциональное отношение 54–61
Функционально полная система 145, 146
– – – в слабом смысле 153
Функциональный символ 210
Функция
– биективная 27
– булева 100
– двойственная 111
– линейная 141
– логическая 100
– монотонные 148–149
– обратная 57
– переключательная 100
– самодвойственная 111
– сохраняющая 0 147
– сохраняющая 1 147
– унарная 235–236
– характеристическая 236
– частично-определенная 165

Х

Характеристическая функция 236

Ц

Цепь 243
– гамильтонова 305
– простая 243, 305
– эйлерова 247
Цикл 243
– гамильтонов 241, 304
– логический 178
– – , анализ 179
– – , синтез 180
– простой 243, 304
– эйлеров 239, 246
Циклическое отрицание 235

Ч

Частично упорядоченное множество 94
Числа Белла 430
– Моргана 430
– Стирлинга 430
– Фибоначчи 434–437
Число независимости 264

Ш

Штрих Шеффера 104, 108, 151

Э

Эквивалентность (эквиваленция) 103, 107, 108, 187, 191–192
Эквивалентные грамматики 344
Элементарная дизъюнкция 128
– конъюнкция 124
– – , ранг 141
– формула логики предикатов 217
Элементы множества 10, 11
– сравнимые в отношении частичного порядка 50
– обратный 77, 88

Я

Ядро дизъюнктивное 156, 167
Язык 341
– , определяемый грамматикой G 344
– исчисления высказываний 202
– конечный 350
– регулярный 350

N

n-арное отношение 32
n-ая степень отношения 39
n-местный предикат 217
n-местный функциональный символ 209

СОДЕРЖАНИЕ

Введение 3
Список обозначений 5

1. МНОЖЕСТВА 9

1.1. Множество. Способы задания множеств 9
Способы задания множеств 11
1.2. Основные понятия теории множеств 14
1.3. Геометрическая интерпретация множеств 18
1.4. Операции на множествах 20
1.5. Алгебра множеств 22
1.6. Бесконечные множества 26

2. ОТНОШЕНИЯ 30

2.1. Понятие отношения. Задание отношений 30
Способы задания бинарных отношений 33
2.2. Операции над отношениями 36
2.3. Свойства бинарных отношений 42
2.4. Отношения эквивалентности, порядка, толерантности 47
2.5. Функциональные отношения 54
Виды отображений 57
2.6. Реляционная модель данных 62

3. АЛГЕБРАИЧЕСКИЕ СТРУКТУРЫ 73

3.1. Алгебраические операции и их свойства 73
3.2. Понятие алгебраической структуры 79
3.3. Простейшие алгебраические структуры 85
3.4. Кольца и поля 90
3.5. Решетки 93

4. БУЛЕВЫ ФУНКЦИИ И ПРЕОБРАЗОВАНИЯ . . . 99

4.1. Булевы переменные и функции 99
4.2. Способы задания булевых функций 102
4.2.1. Таблицы истинности 102

4.2.2. Номера булевых функций и интерпретаций . . 104
4.2.3. Булевы алгебры: общая, двухэлементная и логическая 106
4.2.4. Булевы формулы и приоритет операций 107
4.2.5. Переход от формулы к таблице истинности функции 109
4.3. Двойственность 111
4.4. Законы булевой алгебры 115
4.5. Дизъюнктивные и конъюктивные разложения булевых функций 120
4.6. Нормальные формы представления булевых функций 130
4.7. Алгебра Жегалкина. Линейные функции 138
4.7.1. Тождества алгебры Жегалкина 138
4.7.2. Полином Жегалкина 140
4.8. Полнота и замкнутость 145
4.9. Функции, сохраняющие ноль и единицу. Монотонные функции 147
4.10. Теорема Поста о полноте 151
4.11. Минимизация булевых функций 155
4.11.1. Основные понятия 155
4.11.2. Минимизация булевых функций методом карт Карно (диаграмм Вейча) 158
4.11.3. Минимизация функций методом Квайна — Мак-Класки 166
4.11.4 Минимизация функций методом Порецкого — Блейка 174
4.12. Логические схемы 177

5. МАТЕМАТИЧЕСКАЯ ЛОГИКА 183

5.1. История и задачи математической логики 183
5.2. Понятия логики высказываний 185
5.3. Дедуктивные выводы в логике высказываний . . . 197
5.4. Исчисление высказываний 201
5.5. Логика предикатов 207
5.6. Кванторы 212
5.7. Формулы в логике предикатов 217
5.8. Законы и тождества в логике первого порядка . . . 220
5.9. Предваренные нормальные формы и логический вывод в логике предикатов 223

5.10. Исчисление предикатов 227
5.11. Многозначная логика 231

6. ТЕОРИЯ ГРАФОВ 239
6.1. Исторические замечания. Типичные задачи 239
6.2. Неориентированные графы и терминология 242
6.3. Эйлеровы циклы 246
6.4. Абстрактные графы и геометрические реализации. Ориентированные графы 248
6.5. Матрица смежности, изоморфизмы и операции над графами 253
6.6. Матрица инциденций 257
6.7. Раскраски 260
6.8. Деревья 269
6.9. Теорема Пуанкаре. Фундаментальные матрицы сечений и циклов 286
6.10. Кратчайшие расстояния и пути в сетях 292
6.11. Гамильтоновы циклы и пути. Задача коммивояжера 304
6.12. Потоки в сетях 322

7. ЯЗЫКИ И ГРАММАТИКИ 338
7.1. Задача формализации языков и перевода 338
7.2. Преобразование строк символов 339
7.3. Задание языков с помощью грамматик 341
7.4. Форма Бекуса — Наура 345
7.5. Типы грамматик 347
7.6. Регулярные выражения и языки 351
7.7. Деревья выводов. Стратегии выводов 354
7.8. Построение грамматики языка программирования . . 356

8. АЛГОРИТМЫ 360
8.1. Понятие алгоритма 360
8.2. Нормальные алгоритмы Маркова 362
8.3. Алгоритмы и рекурсивные функции 365
8.5. Примеры построения алгоритмов 369
8.6. Сложность алгоритмов 374
8.7. Использование быстрых алгоритмов 381

9. АВТОМАТЫ 385
9.1. Общая характеристика автоматов 385
9.2. Распознаватели 390
9.3. Конечные автоматы 391
9.4. Автоматы с магазинной памятью 399
9.5. Машина Тьюринга.
Линейно-ограниченные автоматы 402

10. КОМБИНАТОРИКА 408
10.1. Предисловие 408
10.2. Первичные понятия комбинаторного анализа . . . 413
10.3. Перестановки, размещения, сочетания 418
10.4. Формула включений и исключений. Применение . . 426
10.5. Биномиальная и полиномиальная формулы 431
10.6 Комбинаторные задачи и теория чисел 431
10.7. Композиции и разбиения 437
10.8. Производящие функции 443
10.9. Асимптотичные оценки и формулы 453

Послесловие 462

Список литературы 463

Предметный указатель 465

МАТЕМАТИКА ДЛЯ ВСТУПНИКІВ ДО ВУЗІВ

(українською та російською мовами)

Бондаренко М. Ф.
Дікарєв В. А.
Мельников О. Ф.
Семенець В. В.
Шкляров Л. Й.

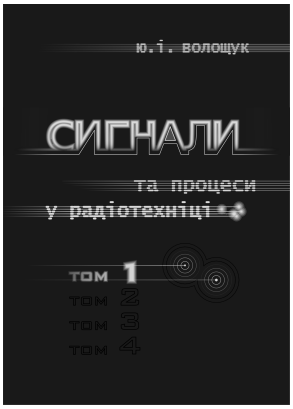
Навчальний посібник містить повну інформацію (довідковий матеріал, класифікацію задач, методи розв'язання) з усіх розділів математики, що включені до програми вступних екзаменів до вузів, включаючи задачі з параметрами. Методи розв'язання задач проілюстровані прикладами. В частині 1 (Алгебра) детально описано розв'язання 587 прикладів і задач. Для більш досконалого вивчення матеріалу в навчальному посібнику наведено завдання для самостійного виконання (519 завдань) та відповіді до них.

У частині 2 (Геометрія. Початки аналізу) наведено розв'язання 382 задач. Крім того, посібник містить 278 задач для самостійного розв'язання та відповіді до них.

Автори даного навчального посібника поставили собі за мету дати абітурієнтам повну інформацію про способи розв'язання достатньо повного «набору» різного роду прикладів і задач, що охоплюють усі розділи програми вступних екзаменів у вузах.

Навчальний посібник призначений для учнів і вчителів середніх шкіл, учнів і викладачів усіх форм довузівської підготовки.

Навчальний посібник. 1120 с. 1/32, 7БЦ.
Українська мова. 1999 р.
Російська мова. 2001 р.



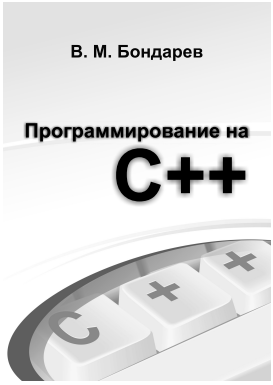
СИГНАЛИ ТА ПРОЦЕСИ У РАДІОТЕХНІЦІ

Волощук Ю. І.
1, 2 т.

У першому томі підручника докладно розглянуто сучасні методи аналізу детермінованих сигналів, математичні моделі керувальних, вузькосмугових, складних сигналів. Особливу увагу приділено сигналам з амплітудною, частотною, фазовою модуляцією.

У другому томі підручника розглянуто сигнали з імпульсною модуляцією, сигнали цифрових систем зв'язку; основи теорії випадкових сигналів, моделі випадкових сигналів, зокрема шумів, завад, зосереджених за часом і спектром, процесів Маркова та ін.; методи аналізу проходження детермінованих сигналів. Подання теоретичного матеріалу супроводжено великою кількістю прикладів, вправ і задач для самостійного розв'язання.

Підручник. 1120 с. Том 1 та 2. 1/16, 7БЦ. Українська мова. 2003 р.



ПРОГРАММИРОВАНИЕ НА C++

Бондарев В. Т.

Книга содержит компактное изложение основ алгоритмического языка C++ и состоит из трех частей. В первой части дано введение в программирование на C++, основной темой второй части является объектно-ориентированное программирование, третья часть вводит читателя в обобщенное программирование и знакомит со стандартной библиотекой шаблонов. Для тех, кто только пробует писать программы, в книге есть приложение, с которого им стоит начать.

Для студентов вузов специальностей «Программное обеспечение автоматизированных систем», «Информатика», «Прикладная математика» и всех, кто хочет научиться программировать на C++.

Учебное пособие. 284 с. 1/16, мягкая обложка. Русский язык. 2004 г.

МОДЕЛИРОВАНИЕ И ПРОЕКТИРОВАНИЕ БИЗНЕС-СИСТЕМ

Бондаренко М. Ф., Маторин С. И., Соловьева Е. А.

Рассмотрены глубинные причины происходящих в настоящее время явлений в области бизнеса и управления организационными системами. Показано, что развитие природы, человека, общества, информационных и организационных систем (бизнес-систем) имеет всеобщую знаниеориентированную направленность.

В рамках одной книги собраны и систематизированы все основные методы и технологии анализа и моделирования бизнес-структур и бизнес-процессов. Описаны теоретические основы предпринимательской деятельности, практические средства повышения ее эффективности. Уделяется внимание современным перспективным информационным технологиям и международным стандартам по методам разработки и повышения качества бизнес-процессов.

Учебное пособие. 272 с. 1/8, мягкая обложка. Русский язык. 2004 г.

КОНТРОЛЬ И УПРАВЛЕНИЕ КОРПОРАТИВНЫМИ КОМПЬЮТЕРНЫМИ СЕТЯМИ: ИНСТРУМЕНТАЛЬНЫЕ СРЕДСТВА И ТЕХНОЛОГИИ

Гуржій А. М., Коряк С.Ф., Самсонов В. В., Скляров О. Я.

В навчально-методичному комплексі (НМК) докладно висвітлено функції та методи застосування апаратно-програмних платформ моніторингу складних комп'ютерних мереж; розглянуто технології збору даних про стан обладнання, програмного забезпечення та мережної активності; наведено методику виявлення проблемних ситуацій, їх причин. НМК містить повний обсяг інформації, необхідний для успішного засвоєння матеріалу в умовах дистанційного навчання (самостійного вивчення).

Для студентів, які вивчають дисципліни «Комп'ютерні мережі», «Апаратне та системне програмне забезпечення комп'ютерних мереж» та ін.

Учебное пособие. 620 с. 1/16, мягкая обложка. Русский язык. 2004 г.

Підручник

Бондаренко М. Ф., Білоус Н. В., Руткас А. Г.

Дискретна математика

Редактор: Драган Н. О.

Комп'ютерна верстка: Кризський А. В.
Дизайн та обкладинка: Денисенко А. О.

Підп. до друку	16.09.2004	Формат	60x90 $^{1}/_{16}$
Друк	офсетний	Папір	офсетний
Умов. друк. арк.	30,0	Зам.	№ 36/09
Ціна договірна.		Тираж	1000 прим.

Видавництво «Компанія СМІТ».
61166, м. Харків, просп. Леніна, 14.
Тел.: 8-(057)-717-54-94, 702-08-16
Факс: 8-(057)-702-13-07
E-mail: book@smit.kharkov.ua

Свідоцтво про внесення суб'єкта видавничої справи до державного реєстру видавців, виготівників і розповсюджувачів видавничої продукції ДК № 435 від 26.04.2001.

Надруковано в друкарні ТОВ «СІМ».
61012, м. Харків, пров. Лопанський, 3-а.
Тел. 8-(057)-719-19-30.
E-mail: tipa_graf@pisem.net