Н. П. Редькин

ДИСКРЕТНАЯ МАТЕМАТИКА

УДК 519.95 (075.8)
ББК 22.18
Р 33

Редькин Н. П. **Дискретная математика.** — М.: ФИЗМАТЛИТ, 2009. — 264 с. — ISBN 978-5-9221-1093-8.

В учебнике представлен основной материал обязательного курса «Дискретная математика», читающегося на механико-математическом факультете МГУ с 1998 г. В сжатой форме он содержит для первоначального ознакомления ряд важных разделов дискретной математики: комбинаторный анализ, графы и сети, важнейшие классы управляющих систем, тесты, алгоритмы, кодирование, дискретные экстремальные задачи. К каждой главе приведены задачи, самостоятельное решение которых будет способствовать более глубокому усвоению теоретического материала и лучшей подготовке к экзамену.

Для студентов и аспирантов.

Рекомендовано УМО по классическому университетскому образованию в качестве учебника для студентов высших учебных заведений, обучающихся по направлениям подготовки 010100 «Математика», 010200 «Математика. Прикладная математика», 011000 «Механика. Прикладная математика».

ISBN 978-5-9221-1093-8

ОГЛАВЛЕНИЕ

Предисловие 7

Глава I. **Элементы комбинаторики** 9
§ 1. Комбинаторные объекты и комбинаторные числа 9
§ 2. Формула включения-исключения. Производящие функции и возвратные последовательности 12

Глава II. **Графы и сети** 19
§ 1. Элементы графа. Подграфы. Способы задания графов 19
§ 2. Геометрическая реализация графов. Верхняя оценка числа неизоморфных графов с m рёбрами 22
§ 3. Деревья. Характеристические свойства деревьев 23
§ 4. Верхняя оценка числа неизоморфных корневых деревьев с m рёбрами 26
§ 5. Теорема Кэли о числе деревьев с занумерованными вершинами 28
§ 6. Двудольные графы. Паросочетания и трансверсали. Теорема Холла 29
§ 7. Сети. Потоки в сетях. Теорема Форда–Фалкерсона 32

Глава III. **Булевы функции и формулы** 40
§ 1. Булевы функции. Элементарные булевы функции 40
§ 2. Формулы и функции, реализуемые формулами. Простейшие эквивалентности 42
§ 3. Разложение булевых функций. Дизъюнктивные нормальные формы 45
§ 4. Полнота систем булевых функций. Представление булевых функций полиномами Жегалкина 47
§ 5. Функции k-значной логики 49

Глава IV. **Предикаты** 51
§1. Высказывания, предикаты, кванторы. Геометрический смысл кванторов 51
§2. Модель, сигнатура модели, формулы в модели. Свободные и связанные переменные 53
§3. Истинность формулы в модели, на множестве. Тождественно истинные формулы 56
§4. Эквивалентность формул. Правила преобразования формул с кванторами 57
§5. Приведённые формулы 60
§6. Нормальные формулы 63

Глава V. **Схемы из функциональных элементов. Синтез и оценки сложности схем** 65
§1. Схемы из функциональных элементов в базисе $\{\&, \vee, ^-\}$ . . 65
§2. Синтез схем с использованием совершенных д.н.ф. 69
§3. Метод Шеннона 70
§4. Асимптотически оптимальный метод синтеза схем (метод Лупанова) 72
§5. Мощностной метод получения нижней оценки для сложности схем 75

Глава VI. **Тесты** 80
§1. Полные диагностические тесты для таблиц. Оценки длины тестов 80
§2. Тесты для схем. Построение минимальных тестов методом Яблонского 82
§3. Верхние оценки длины единичных тестов для схем 88
§4. Синтез легкотестируемых схем 89

Глава VII. **Ограниченно-детерминированные функции и реализация их автоматами** 92
§1. Детерминированные и ограниченно-детерминированные функции 92
§2. Способы задания ограниченно-детерминированных функций 97
§3. Схемы автоматов из функциональных элементов и элементов задержки 99

Глава VIII. **Алгоритмы** 101
§ 1. Алгоритмы. Машины Тьюринга. Задание машины системой команд 101
§ 2. Композиции машин. Тезис Тьюринга 105
§ 3. Проблема самоприменимости. Теорема о самоприменимости 106

Глава IX. **Кодирование** 109
§ 1. Алфавитное кодирование. Разделимые коды. Свойство префикса 109
§ 2. Неравенство Крафта–Макмиллана 112
§ 3. Коды с минимальной избыточностью. Оптимальное кодирование Хаффмена 115
§ 4. Самокорректирующиеся коды. Коды Хэмминга. 120
§ 5. Геометрические свойства самокорректирующихся кодов. Оценки Хэмминга и Гильберта 123

Глава X. **Дискретные экстремальные задачи** 127
§ 1. Задача на покрытие. Точное решение задачи на покрытие 127
§ 2. Градиентный алгоритм поиска приближённого решения. Оценка сложности градиентного покрытия. 129
§ 3. Задача о минимальном остовном дереве. 133
§ 4. Поиск кратчайшего и надёжного путей в графе 135
§ 5. Точное решение задачи на покрытие методом динамического программирования 138
§ 6. Приближённое решение задачи об упаковке в контейнеры 141
§ 7. Классы P и NP. Полиномиальная сводимость задач 143

Задачи 151
К главе I 151
К главе II 153
К главе III. 155
К главе IV 158
К главе V 160
К главе VI 163
К главе VII 164
К главе VIII 168
К главе IX 170
К главе X 171

Ответы, указания, решения 175
К главе I 175
К главе II 179
К главе III 186
К главе IV 200
К главе V 203
К главе VI 218
К главе VII 224
К главе VIII 238
К главе IX 252
К главе X 254

Литература 261

Светлой памяти моего учителя
Олега Борисовича Лупанова
посвящается

Предисловие

Эта книга возникла на основе годового обязательного курса лекций по дискретной математике, читающегося автором на механико-математическом факультете Московского государственного университета им. М. В. Ломоносова для студентов четвёртого курса отделения механики. Главная задача курса — обучение характерным для дискретной математики методам решения основных задач и соответствующему мышлению. Необходимость такого обучения диктуется научно-техническим прогрессом, поставившим перед математиками две крупные проблемы, которые часто оказываются взаимосвязанными. Одна из них — изучение сложных управляющих систем, разработка методов анализа и синтеза таких систем. Другая проблема — необходимость решения ряда новых задач, главной спецификой которых является дискретность. Такие задачи в настоящее время сплошь и рядом возникают как в теории, так и на практике — в экономике, технике, исследовании операций и т. п., а решение их даже с использованием мощной вычислительной техники часто наталкивается на принципиальные, порой непреодолимые затруднения, связанные с неприемлемо большими затратами машинного времени и памяти.

Вошедший в курс материал достаточно условно можно разбить на две части. Одну из них составляют основные понятия, описания изучаемых объектов и доказательства ключевых фактов, теорем, ставших большей частью уже классическими. Сюда относятся: элементы комбинаторики, графы и сети, булевы функции и формулы, предикаты, алгоритмы, кодирование, частично — дискретные экстремальные задачи. Эта часть составляет основу того математического аппарата, владение которым в настоящее

время представляется совершенно необходимым для выпускников математических факультетов.

В другой части представлены важнейшие классы управляющих систем — схемы из функциональных элементов и автоматы, а также типовые задачи, связанные с синтезом и диагностикой состояния этих систем, и основные способы решения таких задач; здесь часто используются понятия, математические модели и теоремы из первой части. Приведены и типичные, наиболее известные дискретные экстремальные задачи, точные и приближенные решения этих задач, примеры получения оценок качества приближенных решений.

Успешное, пусть даже начальное изучение дискретной математики вряд ли возможно без приобретения навыков решения подходящих задач, связанных с основными математическими моделями и теоремами. Для приобретения таких навыков освоение теоретического материала должно сопровождаться практическими занятиями. Для занятий к каждой главе книги прилагаются задачи различной степени трудности. Решение этих, а также, возможно, и других подходящих задач необходимо для глубокого и прочного усвоения теоретического материала.

Содержание данной книги, стиль изложения материала в значительной степени определяются научно-педагогической школой кафедры дискретной математики механико-математического факультета МГУ им. М. В. Ломоносова, основателем и бессменным руководителем которой на протяжении четверти века был выдающийся математик и педагог, академик РАН Олег Борисович Лупанов.

Глава I

ЭЛЕМЕНТЫ КОМБИНАТОРИКИ

§ 1. Комбинаторные объекты и комбинаторные числа

В комбинаторном анализе изучаются различные объекты, порождаемые элементами из конечного множества $A = \{a_1, \dots, a_n\}$, и числовые характеристики этих объектов. Часто рассматриваются, например, упорядоченные или неупорядоченные подмножества множества A, подмножества с повторяющимися элементами из множества A и т. п. Вместе с классами таких комбинаторных объектов естественным образом вводятся и так называемые комбинаторные числа, задающие число объектов в том или ином классе и зависящие от некоторых параметров, например, от мощностей исходного множества A и рассматриваемых подмножеств множества A.

Размещения элементов. Пусть $A = \{a_1, \dots, a_n\}$. *Размещением* элементов из A по k (или *размещением* из n элементов по k) называется упорядоченное подмножество из k элементов множества A.

Для $A = \{a_1, a_2, a_3, a_4\}$ размещениями из A по 3 будут, например, $\{a_1, a_3, a_4\}$, $\{a_3, a_4, a_1\}$, $\{a_2, a_3, a_4\}$.

Обозначим число размещений из n элементов по k через $(n)_k$. При построении конкретного размещения первым элементом в нём можно взять любой из n элементов множества A, вторым элементом — любой из $n-1$ оставшихся в A элементов и т. д. Поэтому

$$(n)_k = n(n-1)\dots(n-k+1) \text{ при } 1 \leqslant k \leqslant n.$$

При $k > n$ не существует размещений из n по k, следовательно, $(n)_k = 0$ при $k > n$; при $k = 0$ полагаем $(0)_0 = (n)_0 = 1$. Нетрудно заметить, что для чисел $(n)_k$ выполняются тождества:

$$\begin{aligned}(n)_k &= n(n-1)_{k-1},\\ (n)_k &= (n)_{k-1} \cdot (n-k+1).\end{aligned}$$

Перестановки элементов. *Перестановками* элементов множества $A = \{a_1, \ldots, a_n\}$ (или *перестановками* из n элементов) называются всевозможные упорядоченные множества из n элементов $a_1, \ldots, a_n$.

Для $A = \{a_1, a_2, a_3\}$ перестановками будут, например, $\{a_1, a_3, a_2\}$, $\{a_2, a_1, a_3\}$, $\{a_2, a_3, a_1\}$.

Перестановки из n элементов — частный случай размещений из n элементов по n. Поэтому

$$(n)_n = n(n-1)\ldots 2 \cdot 1 = n!$$

Как обычно, полагаем $0! = 1$.

Сочетания элементов. *Сочетанием* элементов из $A = \{a_1, \ldots, a_n\}$ по k (или *сочетанием* из n элементов по k) называется неупорядоченное подмножество из k элементов множества A.

Для $A = \{a_1, a_2, a_3\}$ всевозможными сочетаниями по 2 элемента будут $\{a_1, a_2\}$, $\{a_1, a_3\}$, $\{a_2, a_3\}$.

В сочетании, в отличие от размещения, порядок следования элементов не учитывается. Поэтому из одного сочетания (из n элементов по k) получается $k!$ размещений. Отсюда для числа $\binom{n}{k}$ сочетаний из n элементов по k получается формула

$$\binom{n}{k} = \frac{(n)_k}{k!} = \frac{n(n-1)\ldots(n-k+1)}{k!} = \frac{n!}{k!(n-k)!}$$

($0 \leqslant k \leqslant n$; вместо $\binom{n}{k}$ часто используется также обозначение C_n^k). Из последней формулы следует

$$\binom{0}{0} = \binom{n}{0} = \binom{n}{n} = 1 \text{ и } \binom{n}{k} = \binom{n}{n-k}.$$

Для случая $k > n$ полагаем $\binom{n}{k} = 0$, поскольку при $k > n$ не существует сочетаний из n элементов по k.

Числа $\binom{n}{k}$ фигурируют в функциональном тождестве, называемом формулой для бинома Ньютона:

$$(1+x)^n = \binom{n}{0} + \binom{n}{1}x + \ldots + \binom{n}{k}x^k + \ldots + \binom{n}{n}x^n. \qquad (1)$$

В правой части данного тождества коэффициент при x^k есть количество способов, которыми можно выбрать из k скобок $(1+x)$ переменную x, а из остальных скобок 1, что даёт ровно $\binom{n}{k}$ слагаемых.

Полагая в (1) $x = 1$, получим тождество

$$\binom{n}{0} + \binom{n}{1} + \ldots + \binom{n}{k} + \ldots + \binom{n}{n} = 2^n; \tag{2}$$

при $x = -1$ получим

$$\binom{n}{0} - \binom{n}{1} + \ldots + (-1)^n \binom{n}{n} = 0. \tag{3}$$

Из соотношений (2) и (3) следуют тождества

$$\binom{n}{0} + \binom{n}{2} + \ldots + \binom{n}{n} = \binom{n}{1} + \binom{n}{3} + \ldots + \binom{n}{n-1} = 2^{n-1}$$

при чётном n и

$$\binom{n}{0} + \binom{n}{2} + \ldots + \binom{n}{n-1} = \binom{n}{1} + \binom{n}{3} + \ldots + \binom{n}{n} = 2^{n-1}$$

при нечётном n.

Сочетания с повторениями элементов. *Сочетанием с повторениями* элементов из $A = \{a_1, \ldots, a_n\}$ по k (или *сочетанием с повторениями* из n элементов по k) называется неупорядоченный набор из k элементов множества A, в котором элементы могут повторяться.

Для $A = \{a_1, a_2, a_3\}$ всевозможными сочетаниями с повторениями по 2 элемента будут (a_1, a_1), (a_1, a_2), (a_1, a_3), (a_2, a_2), (a_2, a_3), (a_3, a_3).

Обозначим через H_n^k число сочетаний с повторениями из n элементов по k; найдём это число.

Пусть $\widetilde{a}$ — произвольное сочетание с повторениями элементов из $A = \{a_1, \ldots, a_n\}$ по k. Этому сочетанию поставим в соответствие набор $\widetilde{\alpha}(\widetilde{a})$ длины $n + k - 1$ из $n - 1$ нулей и k единиц. В наборе $\widetilde{\alpha}(\widetilde{a})$ число единиц, находящихся между $(i-1)$-м и i-м нулями $(i = 2, \ldots, n-1)$, равно числу элементов a_i, входящих в сочетание $\widetilde{a}$, а число единиц, стоящих перед первым нулём (после последнего нуля), равно числу элементов a_1 (соответственно элементов a_n), входящих в сочетание $\widetilde{a}$. Указанное соответствие между сочетаниями $\widetilde{a}$ и наборами $\widetilde{\alpha}(\widetilde{a})$ взаимно однозначно. Различных наборов длины $n + k - 1$, содержащих $n - 1$ нулей и k единиц, имеется ровно $\binom{n+k-1}{k}$ штук, поскольку каждому такому набору можно взаимно однозначно сопоставить сочетание из $n + k - 1$ элементов по k. В итоге получаем

$$H_n^k = \binom{n+k-1}{k}.$$

§ 2. Формула включения-исключения. Производящие функции и возвратные последовательности

Рассмотрим — главным образом с иллюстративной целью — несколько способов подсчёта комбинаторных чисел.

Формула включения-исключения. Пусть имеется m предметов и n различных свойств, которыми могут обладать эти предметы. Пусть $m(i_1, \dots, i_k)$ — число предметов, обладающих i_1-м, $\dots$, i_k-м свойствами. В таком случае число m_0 предметов, не обладающих ни одним из n свойств, определяется по следующей формуле включения-исключения:

$$m_0 = m - \sum_{i=1}^{n} m(i) + \sum_{1 \leqslant i_1 < i_2 \leqslant n} m(i_1, i_2) - \dots +$$
$$+ (-1)^l \sum_{1 \leqslant i_1 < \dots < i_l \leqslant n} m(i_1, i_2, \dots, i_l) + \dots + (-1)^n m(1, 2, \dots, n). \quad (1)$$

Формулу (1) можно получить, например, индукцией по n. При $n = 1$ имеем $m_0 = m - m(1)$, т. е. формула (1), очевидно, справедлива. Предположим, что формула (1) справедлива для $(n-1)$ свойств. Пусть $m(\overline{i}_1, \dots, \overline{i}_k)$ — число предметов, не обладающих ни одним из свойств $i_1, \dots, i_k$. По предположению индукции имеем

$$m_0' = m(\overline{1}, \dots, \overline{n-1}) = m - \sum_{i=1}^{n-1} m(i) +$$
$$+ \sum_{1 \leqslant i_1 < i_2 \leqslant n-1} m(i_1, i_2) - \dots + (-1)^{n-1} m(1, 2, \dots, n-1). \quad (2)$$

Эта формула справедлива и для отдельно взятой совокупности предметов, обладающих n-м свойством, т. е.

$$m(\overline{1}, \dots, \overline{n-1}, n) = m(n) - \sum_{i=1}^{n-1} m(i, n) +$$
$$+ \sum_{1 \leqslant i_1 < i_2 \leqslant n-1} m(i_1, i_2, n) - \dots + (-1)^{n-1} m(1, 2, \dots, n-1, n), \quad (3)$$

где $m(\overline{1}, \dots, \overline{n-1}, n)$ — число предметов, обладающих свойством n, но не обладающих ни одним из свойств $1, \dots, n-1$. Как нетрудно заметить,

$$m(\overline{1}, \dots, \overline{n-1}, \overline{n}) = m(\overline{1}, \dots, \overline{n-1}) - m(\overline{1}, \dots, \overline{n-1}, n).$$

Отсюда следует, что формула (1) получается вычитанием формулы (3) из (2).

Замечание. Формулу включения-исключения можно интерпретировать ещё и таким образом. Множество всех m предметов обозначим через A, а через A_i обозначим подмножество всех предметов, обладающих i-м свойством ($i = 1, \dots, n$; понятно, что подмножества $A_1, \dots, A_n$ могут быть совершенно произвольными). В этом случае формула включения-исключения запишется так:

$$|A \setminus (A_1 \cup \dots \cup A_n)| = |A| - \sum_{i=1}^{n} |A_i| + \sum_{1 \leqslant i_1 < i_2 \leqslant n} |A_{i_1} \cap A_{i_2}| - \dots +$$
$$+ (-1)^l \sum_{1 \leqslant i_1 < \dots < i_l \leqslant n} |A_{i_1} \cap A_{i_2} \cap \dots \cap A_{i_l}| + \dots +$$
$$+ \dots + (-1)^n |A_1 \cap \dots \cap A_n|.$$

Производящие функции часто используются при нахождении комбинаторных чисел и при установлении комбинаторных тождеств.

Пусть имеется последовательность $a_0, a_1, \dots, a_k, \dots$ Этой последовательности сопоставим ряд

$$\sum_{k=0}^{\infty} a_k t^k$$

(для конечной последовательности $a_0, a_1, \dots, a_k$ указанный ряд превращается в многочлен). В ряде случаев, например, при определённых ограничениях, данный ряд может оказаться сходящимся и задавать некоторую функцию $A(t)$:

$$A(t) = \sum_{k=0}^{\infty} a_k t^k.$$

Эта функция $A(t)$ называется *производящей* для последовательности $\{a_k\}$. Если производящая функция $A(t)$ известна и допускает разложение в ряд Маклорена, причём единственным образом, то этим обстоятельством можно воспользоваться для нахождения комбинаторных чисел a_i.

Проиллюстрируем сказанное на следующем простом примере. Предположим, что нужно найти $\binom{n}{k}$. Нетрудно заметить, что

из правил возведения в степень и определения числа сочетаний из n элементов по k следует равенство

$$(1+t)^n = 1 + \binom{n}{1}t + \ldots + \binom{n}{k}t^k + \ldots + \binom{n}{n}t^n;$$

правую часть данного равенства формально можно рассматривать как некоторый ряд, а выражение в левой части равенства — как производящую функцию для данного ряда. Легко заметить, что функция $(1+t)^n$ допускает разложение в ряд Маклорена, причём единственным образом:

$$(1+t)^n = 1 + \frac{n}{1!}t + \frac{n(n-1)}{2!}t^2 + \ldots + \\ + \frac{n(n-1)\cdot\ldots\cdot(n-k+1)}{k!}t^k + \ldots + \frac{n(n-1)\cdot\ldots\cdot 2\cdot 1}{n!}t^n.$$

В силу единственности разложения коэффициенты при t^k в обоих полученных выражениях для $(1+t)^n$ должны быть равны, т. е.

$$\binom{n}{k} = \frac{n\cdot(n-1)\cdot\ldots\cdot(n-k+1)}{k!}.$$

Второй пример. С помощью производящих функций установим тождество

$$\binom{2n}{n} = \sum_{k=0}^{n}\binom{n}{k}^2.$$

Для этого возьмём тождество

$$(1+t)^{2n} = (1+t)^n(1+t)^n$$

и разложим функции $(1+t)^{2n}$ и $(1+t)^n$ в левой и в правой частях этого тождества в ряды:

$$\sum_{i=1}^{2n}\binom{2n}{i}t^i = \left(\sum_{k=0}^{n}\binom{n}{k}t^k\right)\left(\sum_{m=0}^{n}\binom{n}{m}t^m\right).$$

Сравнивая коэффициенты при t^n в левой и в правой частях последнего тождества, получаем

$$\binom{2n}{n} = \sum_{k=0}^{n}\binom{n}{k}\binom{n}{n-k} = \sum_{k=0}^{n}\binom{n}{k}^2.$$

Возвратные последовательности. При решении многих задач (и не только комбинаторного характера) возникают рекур-

рентные соотношения. Рассмотрим те из них, которые приводят к так называемым возвратным последовательностям.

Последовательность $a_0, a_1, \ldots, a_n, \ldots$ называется *возвратной последовательностью степени* k, если для некоторого k и всех n выполняется соотношение

$$a_{n+k} + p_1 a_{n+k-1} + \ldots + p_k a_n = 0, \tag{4}$$

где коэффициенты p_i $(i = 1, \ldots, k)$ не зависят от n. Многочлен

$$P(x) = x^k + p_1 x^{k-1} + \ldots + p_k \tag{5}$$

называется *характеристическим* для возвратной последовательности (4). Сформулируем и докажем несколько утверждений, в совокупности позволяющих находить решения рекуррентных соотношений вида (4).

Утверждение 1. *Возвратная последовательность степени k полностью определяется заданием её первых k членов и соотношением* (4).

Доказательство легко получается индукцией по n. Первые k членов известны по условию. Далее согласно (4) каждый последующий член последовательности однозначно находится по k предыдущим:

$$a_{n+1} = -p_1 a_n - p_2 a_{n-1} - \ldots - p_k a_{n-k+1}.$$

Утверждение 2. *Если λ является корнем характеристического многочлена, то последовательность $\{\lambda^n\}$ удовлетворяет соотношению* (4).

Доказательство. Требуется доказать, что $\lambda^{n+k} + p_1\lambda^{n+k-1} + \ldots + p_k\lambda^n = 0$, или, что то же самое, $\lambda^n(\lambda^k + p_1\lambda^{k-1} + \ldots + p_k) = 0$. По условию λ является корнем многочлена (5), и выражение в скобках обращается в 0.

Утверждение 3. *Если $\lambda_1, \ldots, \lambda_k$ — простые корни характеристического многочлена* (5), *то общее решение рекуррентного соотношения* (4) *представимо в виде*

$$a_n = c_1\lambda_1^n + \ldots + c_k\lambda_k^n,$$

где $c_1, \ldots, c_k$ — произвольные константы.

Доказательство. Заметим, что если последовательности $\{a_n\}$ и $\{b_n\}$ удовлетворяют соотношению (4), то и последовательность $\{d_n\}$, где $d_n = a_n + b_n$, удовлетворяет соотношению (4). Отсюда и из предыдущего утверждения следует, что $a_n = c_1\lambda_1^n + \ldots + c_k\lambda_k^n$ удовлетворяет соотношению (4).

Убедимся теперь, что любая последовательность $\{a_n\}$, удовлетворяющая (4), может быть представлена в виде $a_n = c_1\lambda_1^n +$

$+ \ldots + c_k\lambda_k^n$, где $c_1, \ldots, c_k$ — подходящие константы. Согласно утверждению 1 любую последовательность $\{a_n\}$, удовлетворяющую соотношению (4), можно указать первыми k членами $a_0, a_1, \ldots, a_{k-1}$. Но в таком случае остаётся лишь показать, что для любых $a_0, a_1, \ldots, a_{k-1}$ существуют $c_1, \ldots, c_k$, такие, что

$$\begin{cases} c_1 + c_2 + \ldots + c_k = a_0, \\ c_1\lambda_1 + c_2\lambda_2 + \ldots + c_k\lambda_k = a_1, \\ \ldots \\ c_1\lambda_1^{k-1} + c_2\lambda_2^{k-1} + \ldots + c_k\lambda_k^{k-1} = a_{k-1}. \end{cases} \tag{6}$$

Определитель системы уравнений (6) является определителем Вандермонда. Известно, что он равен $\prod\limits_{1 \leqslant j < i \leqslant k} (\lambda_i - \lambda_j)$ и не обращается в нуль, если $\lambda_i \neq \lambda_j$ при $i \neq j$. Следовательно, система (6) имеет решение, и притом единственное. Утверждение доказано полностью.

Утверждение 4. *Если $\lambda_1, \ldots, \lambda_s$ — корни характеристического многочлена* (5) *кратностей $r_1, \ldots, r_s$ соответственно, то общее решение рекуррентного соотношения* (4) *имеет вид*

$$a_n = \sum_{i=1}^{s} (c_{i,1} + c_{i,2}n + \ldots + c_{i,r_i}n^{r_i-1})\lambda_i^n,$$

где $c_{i,j}$ $(i = 1, \ldots, s;\ j = 1, \ldots, r_i)$ — произвольные константы.

Доказательство. Убедимся, что всякая последовательность указанного в утверждении вида удовлетворяет (4). Для этого покажем, что если $a_n = n^m\lambda^n$, где λ — корень кратности r $(r > m)$ многочлена (5), то последовательность $\{a_n\}$ удовлетворяет (4).

Обозначим через $A(\lambda)$ выражение в левой части (4). Подставляя $a_l = l^m\lambda^l$ $(l = n, \ldots, n + k)$ в $A(\lambda)$ и полагая $p_0 = 1$, получим

$$A(\lambda) = (n+k)^m\lambda^{n+k} + p_1(n+k-1)^m\lambda^{n+k-1} + \ldots$$

$$\ldots + p_k n^m\lambda^n = \lambda^n \left(\sum_{j=0}^{k} p_j (n+k-j)^m \lambda^{k-j} \right) =$$

$$= \lambda^n \left(\sum_{j=0}^{k} \lambda^{k-j} p_j \sum_{i=0}^{m} \binom{m}{i} (k-j)^i n^{m-i} \right) =$$

$$= \lambda^n n^m \left(\sum_{i=0}^{m} \binom{m}{i} n^{-i} \sum_{j=0}^{k} p_j \lambda^{k-j} (k-j)^i \right) =$$

$$= \lambda^n n^m \left(\sum_{i=0}^{m} \binom{m}{i} n^{-i} P_i(\lambda) \right),$$

где $P_i(x) = \sum_{j=0}^{k} p_j (k-j)^i x^{k-j}$; отметим, что $P_0(x)$ есть характеристический многочлен $P(x)$. По условию λ — корень кратности r многочлена P_0, и потому $P_0(x)$ можно представить в виде

$$P_0(x) = (x-\lambda)^r Q_0(x),$$

где Q_0 — некоторый многочлен. Очевидно,

$$P_i(x) = x \frac{d}{dx} P_{i-1}(x)$$

при $0 < i < r$. Из последних двух соотношений получаем

$$P_i(x) = (x-\lambda)^{r-i} Q_i(x),$$

где Q_i — какой-то многочлен, и $P_i(\lambda)$ обращается в нуль при всех $i < r$. Значит, $A(\lambda) = 0$, т. е. соотношение (4) для рассматриваемой последовательности $\{a_n\}$ выполняется.

С другой стороны, докажем, что если λ_i — корень кратности r_i $(i = 1, \ldots, s)$ многочлена (5), то любая последовательность $\{a_n\}$, удовлетворяющая (4), может быть задана в виде, указанном в утверждении. Без ограничения общности будем считать, что $p_k \neq 0$, т. е. что корни характеристического многочлена отличны от нуля (если $p_k = 0$, то соотношение (4) можно упростить). Для доказательства достаточно показать (как и в соответствующем месте доказательства предыдущего утверждения), что для любых $a_0, a_1, \ldots, a_{k-1}$ система

$$\begin{cases} c_{1,1} + \ldots + c_{s,1} = a_0, \\ (c_{1,1} + c_{1,2} + \ldots + c_{1,r_1})\lambda_1 + \ldots \\ \quad \ldots + (c_{s,1} + c_{s,2} + \ldots + c_{s,r_s})\lambda_s = a_1, \\ \ldots \\ (c_{1,1} + c_{1,2}(k-1) + \ldots + c_{1,r_1}(k-1)^{r_1-1})\lambda_1^{k-1} + \ldots \\ \quad \ldots + (c_{s,1} + c_{s,2}(k-1) + \ldots + c_{s,r_s}(k-1)^{r_s-1})\lambda_s^{k-1} = a_{k-1} \end{cases} \tag{7}$$

имеет единственное решение (относительно неизвестных $c_{i,j}$). Для этого, в свою очередь, достаточно показать, что определи-

тель системы (7) не равен нулю, или что векторы $\widetilde{\lambda}_0, \widetilde{\lambda}_1, \ldots, \widetilde{\lambda}_{k-1}$, где $\widetilde{\lambda}_i = (\lambda_1^i, i\lambda_1^i, \ldots, i^{r_1-1}\lambda_1^i, \ldots, \lambda_s^i, i\lambda_s^i, \ldots, i^{r_s-1}\lambda_s^i)$ $(i = 0, 1, \ldots$ $\ldots, k-1)$, линейно независимы.

Предположим противное, т. е. предположим, что существуют константы $b_0, b_1, \ldots, b_{k-1}$, не все равные нулю, такие, что $\widetilde{\lambda} = b_0\widetilde{\lambda}_0 + \ldots + b_{k-1}\widetilde{\lambda}_{k-1} = \widetilde{0}$. Пусть $Q(x) = \sum_{i=0}^{k-1} b_i x^i$ и Δ — оператор, такой, что $\Delta f(x) = x\frac{df}{dx}$; как обычно, считаем $\Delta^k f = \Delta(\Delta^{k-1} f)$. В таком случае

$$\widetilde{\lambda} = (Q(\lambda_1), \Delta Q(\lambda_1), \ldots, \Delta^{r_1-1}Q(\lambda_1), \ldots$$
$$\ldots, Q(\lambda_s), \Delta Q(\lambda_s), \ldots, \Delta^{r_s-1}Q(\lambda_s)).$$

Отметим, что $Q(\lambda) = \Delta Q(\lambda) = \ldots = \Delta^{r-1}Q(\lambda) = 0$ в том и только в том случае, когда λ является корнем кратности не меньше r многочлена $Q(x)$. Следовательно, $\widetilde{\lambda} = \widetilde{0}$ означает, что λ_1 является корнем кратности не меньше r_1, λ_2 является корнем кратности не меньше r_2 и т. д., наконец, λ_s является корнем кратности не меньше r_s многочлена $Q(x)$. Но $r_1 + r_2 + \ldots + r_s = k$, а $Q(x)$ является многочленом степени меньше k и потому не может иметь k корней. Полученное противоречие исключает наше предположение. Значит, система (7) имеет решение и притом единственное. Утверждение доказано.

Глава II

ГРАФЫ И СЕТИ

§ 1. Элементы графа. Подграфы. Способы задания графов

Множество $V = \{v_1, v_2, \ldots\}$ вместе с набором $E = (e_1, e_2, \ldots)$ пар элементов из V называется *графом* G (через e_i обозначаем i-ю пару $(v_{j(i)}, v_{k(i)})$ в E); элементы множества V называются *вершинами* графа G, а пары набора E — *рёбрами* графа G. В наборе E допускаются пары вида (v_i, v_i) и одинаковые пары. Ребро (v_i, v_j) *соединяет* вершины v_i и v_j, которые *инцидентны* данному ребру. Граф G — *конечный*, если множество V и набор E конечны, и *бесконечный* в противном случае.

Если (v_i, v_j) — упорядоченная пара вершин (т.е. $v_i \neq v_j$ и $(v_i, v_j) \neq (v_j, v_i)$), то ребро (v_i, v_j) считается *ориентированным* ребром, или *дугой*, *исходящей* из вершины v_i и *входящей* в вершину v_j; v_i называется *началом*, а v_j — *концом* дуги (v_i, v_j). Если (v_i, v_j) — неупорядоченная пара, то ребро (v_i, v_j) считается неориентированным; вершины v_i, v_j являются *концами* неориентированного ребра (v_i, v_j).

Граф считается *ориентированным* (*частично ориентированным* или *неориентированным*), если все его рёбра ориентированы (соответственно некоторые рёбра ориентированы или все рёбра неориентированы).

Вершина графа, не инцидентная ни одному ребру, называется *изолированной*.

Вершина, инцидентная ровно одному ребру, и само это ребро называются *концевыми* (или *висячими*).

Ребро с совпадающими концами называется *петлёй*.

Две вершины, инцидентные одному и тому же ребру, называются *соседними* (или *смежными*).

Два ребра, инцидентные одной и той же вершине, называются *смежными*.

Одинаковые пары в наборе E графа G задают *кратные*, или *параллельные* рёбра.

Весьма часто требуется уточнить, какие графы считаются различными, а какие не различаются. Это уточнение обычно связывается с понятием изоморфизма графов. Два графа называются *изоморфными*, если существуют взаимно однозначное соответствие между их вершинами и взаимно однозначное соответствие между их рёбрами, такие, что соответствующие рёбра соединяют соответствующие вершины.

Граф $G_1 = (V_1, E_1)$ называется *подграфом* графа $G = (V, E)$, если $V_1 \subseteq V$ и $E_1 \subseteq E$.

Некоторые разновидности графов (и подграфов) встречаются очень часто и носят специальные названия. Укажем некоторые из них.

Последовательность вершин и рёбер графа $G = (V, E)$ вида

$$v_{i_0}, (v_{i_0}, v_{i_1}), v_{i_1}, (v_{i_1}, v_{i_2}), v_{i_2}, \ldots, v_{i_{n-1}}, (v_{i_{n-1}}, v_{i_n}), v_{i_n}, \qquad (1)$$

где все $v_{i_j} \in V$ и все $(v_{i_j}, v_{i_{j+1}}) \in E$, называется *путём из вершины* v_{i_0} *в вершину* v_{i_n}. Вершина v_{i_0} называется *началом*, а v_{i_n} — *концом* пути; число n называется *длиной* пути.

В общем случае среди вершин и рёбер последовательности (1) могут быть повторения. Каждое ребро пути может быть ориентированным (обязательно из $v_{i_{j-1}}$ в v_{i_j}, или, как ещё говорят, от $v_{i_{j-1}}$ к v_{i_j}; $1 \leqslant j \leqslant n$) или неориентированным. Будем говорить, что путь (1) проходит через вершины $v_{i_0}, v_{i_1}, \ldots, v_{i_n}$ по рёбрам $(v_{i_0}, v_{i_1}), (v_{i_1}, v_{i_2}), \ldots, (v_{i_{n-1}}, v_{i_n})$.

Цепь — это путь без повторяющихся рёбер. Цепь без повторяющихся вершин называется *простой цепью*.

Если в последовательности (1) $v_{i_0} = v_{i_n} (n > 0)$, то такой путь называется *замкнутым*. Замкнутый путь без повторяющихся рёбер (дуг) называется *циклом*. *Простой цикл* — это цикл без повторяющихся вершин (не считая последней вершины в записи цикла (1)). Простой цикл с ориентированными рёбрами будем называть также *контуром*.

На рисунках графы изображаются так. Вершины изображаются точками (или кружочками), а каждое ребро (дуга) $(v_i, v_j) \in E$ — линией, соединяющей вершины (точки) v_i и v_j. Если (v_i, v_j) — дуга, то на соответствующей линии указывается стрелка от v_i к v_j.

Для графа на рис. 1 имеем: v_2, e_3, v_4, e_4, v_1, e_1, v_2, e_3, v_4, e_6, v_3 – путь из вершины v_2 в вершину v_3; v_4, e_4, v_1, e_1, v_2, e_3, v_4,

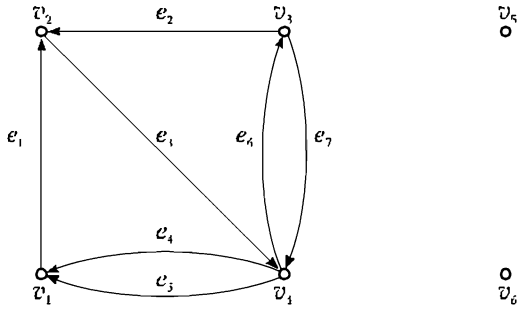


Рис. 1

e_5, v_1 – цепь; v_4, e_4, v_1, e_1, v_2 — простая цепь; v_4, e_6, v_3, e_7, v_4, e_4, v_1, e_1, v_2, e_3, v_4 – цикл; v_4, e_6, v_3, e_7, v_4 — простой цикл.

Существуют различные способы задания графов. Укажем некоторые из них.

1) Указанием множества вершин и списка всех рёбер, в котором каждое ребро есть пара (упорядоченная для ориентированных графов) вершин, являющихся концами этого ребра. Такое задание, очевидно, отвечает приведённому определению графа.

2) Перечислением (списком) рёбер графа с добавлением списка V' изолированных вершин.

3) *Матрицей инциденций* графа с n вершинами $v_1, \ldots, v_n$ и m рёбрами $e_1, \ldots, e_m$ — прямоугольной матрицей $A = \|a_{ij}\|$ с n строками и m столбцами:

$$a_{ij} = \begin{cases} 0, & \text{если вершина } v_i \text{ не инцидентна ребру } e_j; \\ 1, & \text{если } v_i \text{ является концом дуги} \\ & \text{или неориентированного ребра } e_j; \\ -1, & \text{если } v_i \text{ является началом дуги } e_j \\ & \text{(у неориентированных рёбер} \\ & \text{обоим концам соответствует } +1). \end{cases}$$

В каждом столбце матрицы — два ненулевых элемента (если ребро не петля).

4) *Матрицей соседства вершин* — квадратной матрицей $B = \|b_{ij}\|$ размера n, в которой b_{ij} равно числу рёбер, идущих из вершины v_i в вершину v_j (b_{ij} не равно, вообще говоря, b_{ji}, однако для неориентированных графов $b_{ij} = b_{ji}$).

Матрица инциденций задаёт граф однозначно; матрица соседства вершин определяет граф с точностью до замены любого неориентированного ребра парой противоположно направленных

дуг между теми же вершинами. Однако граф без кратных рёбер задаётся и матрицей соседства однозначно.

Приведённый на рис. 1 граф G в соответствии с перечисленными выше способами можно задать так.

1) $G : V = \{v_1, v_2, \dots, v_6\}$; $E = ((v_1, v_2),\ (v_3, v_2),\ (v_2, v_4),\ (v_4, v_1),\ (v_4, v_1),\ (v_4, v_3),\ (v_3, v_4))$.

2) $G : E = ((v_1, v_2),\ (v_3, v_2),\ (v_2, v_4),\ (v_4, v_1),\ (v_4, v_1),\ (v_4, v_3),\ (v_3, v_4))$; $V' = \{v_5, v_6\}$.

3) G: матрица инциденций

$$A = \begin{bmatrix} -1 & 0 & 0 & 1 & 1 & 0 & 0 \\ 1 & 1 & -1 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 & 1 & -1 \\ 0 & 0 & 1 & -1 & -1 & -1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}.$$

4) G: матрица соседства вершин

$$B = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 2 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}.$$

§ 2. Геометрическая реализация графов. Верхняя оценка числа неизоморфных графов с m рёбрами

Пусть задан граф $G = (V, E)$ с множеством вершин V и множеством рёбер E. Пусть каждой вершине v_i $(i = 1, \dots, n)$ из V взаимно однозначно соответствует точка a_i в некотором евклидовом пространстве, а каждому ребру $e = (v_i, v_j)$ из E взаимно однозначно соответствует непрерывная кривая L, соединяющая точки a_i и a_j и не проходящая через другие точки a_k. Если все кривые, соответствующие рёбрам, не имеют общих точек, кроме, быть может, концевых, то это множество точек и кривых называется *геометрической реализацией графа* G в евклидовом пространстве.

Теорема 1. *Каждый конечный граф можно реализовать в трёхмерном евклидовом пространстве.*

Доказательство. Пусть граф $G = (V, E)$ содержит n вершин и m рёбер. Возьмём прямую и проведём через неё m плоскостей. На прямой выделим n точек $a_1, \dots, a_n$ и сопоставим их со-

ответственно вершинам графа $v_1, \ldots, v_n$. Каждому ребру графа G поставим в соответствие одну из проведённых плоскостей. Пусть $e = (v_i, v_j)$ — ребро графа G. В плоскости, соответствующей ребру e, соединим точки a_i и a_j дугой окружности. Выполнив такое построение для всех рёбер графа G, получим (из точек $a_1, \ldots, a_n$ и дуг окружностей) геометрическую реализацию графа G. Теорема доказана.

Обозначим через $\gamma(m)$ число попарно неизоморфных графов без изолированных вершин с m рёбрами.

Теорема 2. *Для любого натурального m*

$$\gamma(m) < (cm)^m,$$

где c — некоторая константа.

Доказательство. Очевидно, что в каждом из рассматриваемых графов число вершин n не превосходит $2m$. Каждое ребро определяется двумя вершинами (концами), не обязательно различными, так что для каждого ребра имеется не более $(2m)^2$ возможностей выбора концов, а для m рёбер — не более, чем число сочетаний с повторениями из $4m^2$ элементов по m, т. е. не более, чем $C^m_{4m^2+m-1}$. Следовательно,

$$\gamma(m) \leqslant C^m_{4m^2+m-1} < \frac{(5m^2)^m}{m!}.$$

Используя известное неравенство

$$m! > \left(\frac{m}{3}\right)^m,$$

получаем

$$\gamma(m) < \frac{(5m^2)^m \cdot 3^m}{m^m} = (cm)^m,$$

где $c = 15$. Теорема доказана.

§ 3. Деревья. Характеристические свойства деревьев

Будем рассматривать неориентированные графы.

Граф $G = (V, E)$ называется *связным*, если для любых двух вершин $v, v' \in V$ в G существует путь из v в v'.

Две вершины v и v' графа G *связаны*, если в G существует путь из v в v'.

Легко видеть, что отношение между вершинами графа «быть связанными» рефлексивно (v связана с v для любой вершины $v \in V$), симметрично (если v связана с v', то и v' связана с v) и транзитивно (если v связана с v', а v' связана с v'',

то v связана с v''), т. е. является отношением эквивалентности. Вершины графа G разбиваются на классы эквивалентности: $V = V_1 \cup V_2 \cup \ldots \cup V_k$, где $V_i \cap V_j = \varnothing$ при $i \neq j$ и любые две связанные вершины v, v' принадлежат некоторому подмножеству V_i. Обозначим через E_i множество всех тех рёбер $e = (v, v')$, для которых концы $v, v' \in V_i$. Подграф $G_i = (V_i, E_i)$ называется *связной компонентой*, или *компонентой связности* графа G. Например, граф на рис. 2 имеет 3 связные компоненты.

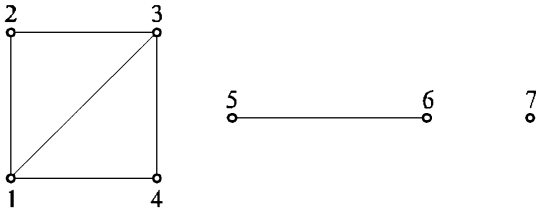


Рис. 2

Лемма 1. *Если $G = (V, E)$ — связный граф, то при добавлении к G любого ребра (v, v'), где $v, v' \in V$, в полученном графе будет простой цикл.*

Доказательство. Поскольку G — связный граф, то в нём найдётся путь P из v в v'. Если этот путь не является простой цепью, то в нём повторяется некоторая вершина v_i, т. е. он имеет вид $P = vP_1v_iP_2v_iP_3v'$. В этом случае сократим путь P, выбросив из него P_2, и получим путь $P' = vP_1v_iP_3v'$. Поскольку P — конечный путь, то, повторяя при необходимости указанную операцию сокращения, получим простую цепь Z из v в v' (соединяющую v с v'). После добавления к Z первоначально отсутствовавшего в G ребра (v, v') получим простой цикл. Лемма доказана.

Лемма 2. *Граф из n вершин и m рёбер содержит не менее $(n - m)$ связных компонент, а если в этом графе нет циклов, то он содержит ровно $n - m$ связных компонент.*

Доказательство. При добавлении к произвольному графу, содержащему вершины v и v', одного ребра $e = (v, v')$ число связных компонент может только уменьшиться, но не более чем на 1. При добавлении m рёбер число связных компонент в графе может уменьшиться не более чем на m. Граф $G = (V, E)$ с n вершинами и m рёбрами получается из графа $G' = (V, \varnothing)$, имеющего n связных компонент, добавлением m рёбер, и согласно вышесказанному в G будет не менее $n - m$ связных компонент.

Предположим теперь, что в G нет циклов. Допустим, что в процессе получения G из G' добавляется ребро (v, v'). Если бы

вершины v и v' принадлежали одной связной компоненте, то в G, согласно лемме 1, появился бы цикл. Следовательно, v и v' принадлежат двум разным связным компонентам и при добавлении ребра (v, v') число связных компонент уменьшается ровно на 1. Это означает, что в G окажется ровно $n - m$ связных компонент. Лемма доказана.

Следствие. *Любой граф G с n вершинами и m рёбрами при $m \leqslant n - 2$ несвязен.*

Доказательство. Из леммы 2 и условия $m \leqslant n - 2$ следует, что G содержит не менее двух связных компонент.

Граф $G = (V, E)$ называется *деревом*, если он связный и не содержит циклов.

Если $G = (V, E)$ — связный граф, $e \in E$ и e входит в некоторый цикл, т. е. e — *циклическое ребро*, то граф G_1, полученный из G удалением ребра e (но с сохранением вершин, инцидентных e) остаётся, как нетрудно заметить, связным. Удаляя из G в произвольной последовательности циклические рёбра, можно получить связный подграф графа G с множеством вершин V, но без циклов, т. е. дерево; это дерево называется *остовным деревом* (или *остовом*) графа G. Таким образом, всякий связный граф $G = (V, E)$ содержит хотя бы один подграф $D' = (V, E')$ с тем же множеством вершин, являющийся деревом.

Теорема 3. *Пусть $G = (V, E)$ — граф с n вершинами и m рёбрами. Тогда следующие условия эквивалентны*:

1) *G — дерево*;

2) *любые две вершины G связаны ровно одной простой цепью*;

3) *G не содержит циклов и $m = n - 1$*;

4) *G — связный граф и $m = n - 1$*;

5) *G — связный граф, но при удалении любого ребра становится несвязным*;

6) *G не содержит циклов, но при добавлении любого нового ребра с концами из V появляется цикл.*

Доказательство. Докажем переходы 1) $\Rightarrow$ 2) $\Rightarrow$ ... $\Rightarrow$ $\Rightarrow$ 6) $\Rightarrow$ 1), откуда будет следовать, что из любого условия вытекает любое другое, т. е. перечисленные условия представляют собой характеристические свойства дерева.

1) $\Rightarrow$ 2). Предположим, что данный переход не выполняется, тогда в дереве G существуют две простые цепи Z_1 и Z_2, связывающие v и v'. В этом случае хотя бы одно какое-то ребро e принадлежит одной из этих цепей, например, Z_1, и не принадлежит другой. Пойдём по Z_1 в обе стороны от ребра e до первой встречи со второй цепью в вершинах v_i и v_j, т. е. возьмём максимальную

(по включению) подцепь Z_1' первой цепи, содержащую ребро e и не содержащую внутри себя вершин второй цепи. Подцепь Z' вместе с подцепью Z_2' второй цепи, ведущей из v_i в v_j, образует цикл. Следовательно, G не есть дерево. Получаем противоречие. Значит переход 1) $\Rightarrow$ 2) выполняется.

2) $\Rightarrow$ 3). Условие 2) означает, что в G отсутствуют циклы, поскольку концы любого ребра из цикла связаны по крайней мере двумя простыми цепями, одну из которых составляет само ребро (вместе со своими концами), а вторую — дополнение этого ребра до цикла. Из условия 2) следует также, что граф G представляет собой связную компоненту. Поэтому из леммы 2 получаем $m = n - 1$.

3) $\Rightarrow$ 4). Из условия 3) и леммы 2 следует, что число связных компонент в G равно $n - m = 1$, т. е. G — связный граф и $m = n - 1$.

4) $\Rightarrow$ 5). Вытекает из следствия к лемме 2.

5) $\Rightarrow$ 6). Если бы в G был цикл, то при удалении ребра из этого цикла граф G остался бы связным, а это противоречит условию 5). Значит, G не содержит циклов. Вторая часть условия 6) вытекает, согласно лемме 1, из того, что G — связный граф.

6) $\Rightarrow$ 1). Если G — не связный граф и вершины v, v' принадлежат разным связным компонентам графа G, то добавление к G ребра (v, v'), очевидно, не порождает циклов, что противоречит 6). Значит, G – связный граф, не содержащий циклов (по условию), т. е. G — дерево. Теорема доказана.

Условия 5) и 6) показывают, что множество всех деревьев с n вершинами – это множество всех минимальных (по числу рёбер) связных графов и вместе с тем множество всех максимальных графов без циклов.

§ 4. Верхняя оценка числа неизоморфных корневых деревьев с m рёбрами

Выделим в дереве какую-нибудь одну вершину, которую назовём *корнем*; полученное дерево с выделенной вершиной называется *деревом с корнем*, или *корневым деревом*. Если в двух изоморфных деревьях выделяются в качестве корней соответствующие друг другу вершины, то получающиеся при этом корневые деревья также считаются *изоморфными*.

Теорема 4. *Число неизоморфных корневых деревьев с m рёбрами не превосходит 4^m.*

Доказательство. Будем кодировать корневые деревья словами (наборами) из нулей и единиц по следующему индуктивному правилу.

1. Кодом дерева из одного ребра является слово 01.

2. а) Если дерево имеет одно *корневое ребро* (т. е. ребро, инцидентное корню), то его кодом является слово $0A1$, где A — код дерева, получающегося при удалении корневого ребра и объявлении корнем второго конца этого ребра.

б) Если корневых рёбер несколько, скажем, k (рис. 3,*а*; здесь и далее корень помечен звёздочкой), то можно считать, что дерево получено склеиванием в корне k поддеревьев с меньшим числом рёбер (рис. 3,*б*). Если $A_1, \ldots, A_k$ — коды этих поддеревьев, то слово $A_1 \ldots A_k$ является кодом всего дерева.

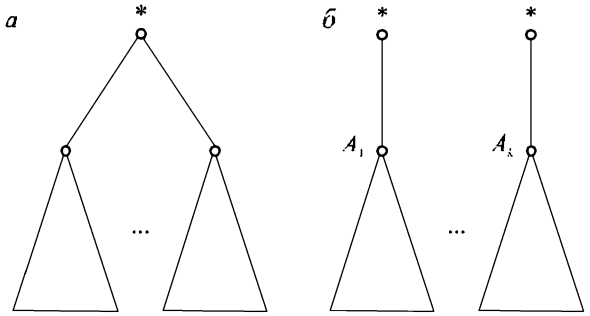


Рис. 3

Очевидно, что код дерева с m рёбрами есть слово длины $2m$, содержащее поровну нулей и единиц, причём в любом начальном отрезке этого слова нулей не меньше, чем единиц (формально это можно установить индукцией по m в соответствии с сформулированным выше индуктивным правилом построения слов). Например, кодом дерева, изображённого на рис. 4, является слово 0010110011.

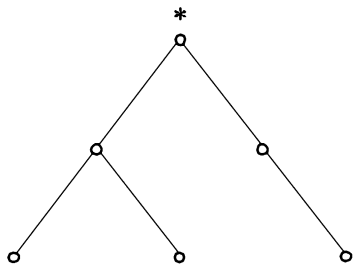


Рис. 4

Заметим, что в случае применения правила 2б) слово A_1 является наименьшим непустым начальным отрезком слова $A_1 \ldots A_k$, содержащим поровну нулей и единиц; то же самое верно для A_2 по отношению к слову $A_2 \ldots A_k$ и т. д. Поэтому по своему коду дерево восстанавливается однозначно. Таким

образом, число неизоморфных корневых деревьев с m рёбрами не превосходит числа слов длины $2m$ из нулей и единиц, т. е. 4^m. Теорема доказана.

§ 5. Теорема Кэли о числе деревьев с занумерованными вершинами

Будем рассматривать n-вершинные деревья, в каждом из которых вершины занумерованы числами $1, 2, \ldots, n$. Различными будем считать деревья, не переводимые друг в друга изоморфизмом, сохраняющим нумерацию вершин. Докажем следующее утверждение.

Теорема 5 (Кэли). *Число деревьев с n занумерованными вершинами равно n^{n-2}.*

Доказательство. Рассмотрим дерево с n занумерованными вершинами. Условимся обозначать каждую вершину её номером. В (конечном) дереве существует самая длинная простая цепь — иначе в нём были бы циклы; начальная и конечная вершины этой цепи будут концевыми вершинами в дереве. Следовательно, в любом дереве (содержащем хотя бы одно ребро) найдутся по крайней мере две концевые вершины. Пусть i_1 — концевая вершина дерева с наименьшим номером, j_1 — единственная вершина, соседняя с i_1. Удалим из D вершину i_1 и ребро (i_1, j_1), в результате получим (см. условие 4) теоремы 3) дерево D_1. Пусть i_2 — концевая вершина с наименьшим номером в D_1, а (i_2, j_2) — инцидентное ей ребро; удалим его, и т. д. до тех пор, пока после удаления ребра (i_{n-2}, j_{n-2}) не останется дерево D_{n-2} из одного ребра. Введём наборы $I = (i_1, i_2, \ldots, i_{n-2})$, $J = (j_1, j_2, \ldots, j_{n-2})$. Для дерева, изображённого на рис. 5, $I = (3, 5, 6, 1, 2, 7)$, $J = (2, 4, 1, 2, 4, 4)$.

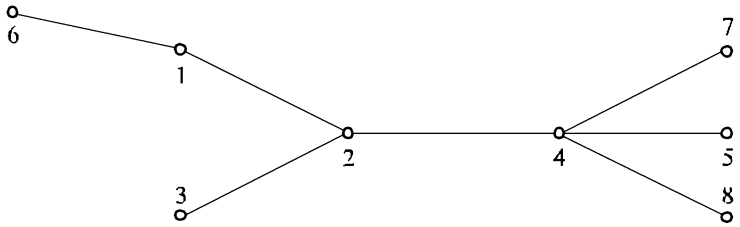


Рис. 5

Набор $I = (i_1, i_2, \ldots, i_{n-2})$, а вместе с ним и набор $J = (j_1, j_2, \ldots, j_{n-2})$ определяются по данному дереву D однозначно, причём в первом наборе все вершины различны, а во втором

могут повторяться. Убедимся, что любому набору $J = (j_1, \ldots, j_{n-2})$, где $1 \leqslant j_k \leqslant n$ для $k = 1, 2, \ldots, n-2$, соответствует некоторое дерево; оно может быть построено следующим образом.

Пусть i_1 — наименьшее число из $N = \{1, 2, \ldots, n\}$, отсутствующее в наборе J: такое число обязательно существует, так как J содержит $n-2$ чисел. Соединим ребром вершины i_1 и j_1, вычеркнем число j_1 из набора J, число i_1 из N и повторим процесс, выбирая из $N_1 = N \setminus \{i_1\}$ наименьшее число i_2, отсутствующее в наборе $J_1 = (j_2, j_3, \ldots, j_{n-2})$, соединяя i_2 с j_2 и вычёркивая j_2 из набора J_1 и i_2 из N_1, и т. д. Последними соединяются ребром две вершины, оставшиеся в N_{n-2}.

Нетрудно заметить, что для любого $k = 1, 2, \ldots, n-2$ среди рёбер, построенных после k-го шага, нет рёбер, инцидентных вершине i_k, и имеется хотя бы одно ребро, инцидентное вершине j_k. Учитывая это и рассматривая процесс построения в обратном порядке, нетрудно показать индукцией по k, что действительно получается дерево, так как теперь на каждом шаге присоединяется ребро с новой вершиной, т. е. концевое. Аналогичным образом, по индукции, но уже при рассмотрении процесса построения в прямом направлении доказывается, что этому дереву соответствует как раз набор J. Получается, что число деревьев не меньше числа наборов, т. е. не меньше, чем n^{n-2} (элементом набора J может быть любое число из N).

Наконец, различным деревьям, очевидно, не может соответствовать одна и та же пара наборов (I, J), т. к. по паре (I, J) дерево восстанавливается однозначно. Но если $I' \neq I''$, то $J' \neq J''$. Действительно, пусть k – наименьший номер, при котором $i'_k \neq i''_k$, скажем, $i'_k < i''_k$; отсюда, в частности, следует, что i'_k не может быть концевой вершиной у дерева $(j''_k, \ldots, j''_{n-2})$ и поэтому i'_k войдёт в $(j''_k, \ldots, j''_{n-2})$. В таком случае i'_k не входит в набор $(j'_k, \ldots, j'_{n-2})$, но входит в $(j''_k, \ldots, j''_{n-2})$. Поэтому разным деревьям соответствуют разные наборы J; значит, число деревьев не превосходит числа наборов, равного n^{n-2}.

В итоге получаем, что искомое число деревьев равно n^{n-2}. Теорема доказана.

§ 6. Двудольные графы. Паросочетания и трансверсали. Теорема Холла

Граф называется *двудольным*, если существует такое разбиение множества его вершин на два непересекающихся подмно-

жества (на две доли), что концы каждого ребра принадлежат разным подмножествам (разным долям; как и при рассмотрении деревьев, здесь мы ограничиваемся неориентированными графами).

Произвольное подмножество попарно несмежных рёбер графа (взятых вместе со своими вершинами) называется *паросочетанием* (или *независимым множеством рёбер*).

Отметим, что паросочетания могут рассматриваться в произвольных графах (не обязательно двудольных). Но весьма часто паросочетания естественным образом возникают и рассматриваются в двудольных графах. (С построением паросочетаний в двудольном графе связано много самых разных задач. Типичный пример — задача о назначении на должности.)

Установим критерий существования паросочетания, покрывающего долю вершин в двудольном графе.

Пусть v — некоторая вершина графа $G = (V, E)$. Через $N(v)$ обозначим множество вершин графа G, соседних с вершиной v. Для произвольного подмножества V' вершин графа G рассмотрим множество

$$N_G(V') = \bigcup_{v \in V'} N(v) \setminus V',$$

которое назовём *окружением* подмножества V' в графе G. Например, для графа G (рис. 6) $N_G(\{3, 5, 7\}) = \{2, 4, 6, 8\}$.

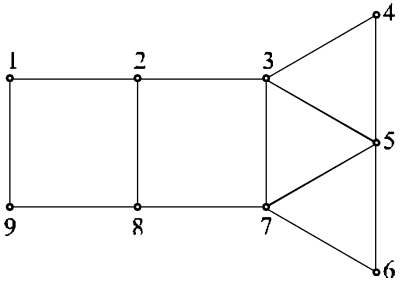


Рис. 6

Теорема 6. *Для существования в двудольном графе* $G = (V \cup W, E)$ *паросочетания, покрывающего долю вершин* V, *необходимо и достаточно, чтобы любое подмножество* V' *множества* V *удовлетворяло условию*

$$|V'| \leqslant |N_G(V')|.$$

Доказательство. *Необходимость.* Очевидно, что если в графе $G = (V \cup W, E)$ существует паросочетание, покрываю-

щее V, то для любого подмножества V' в V число вершин, смежных с вершинами из V', не меньше, чем число вершин в V', т. е. $N_G(V') \geqslant |V'|$.

Достаточность докажем индукцией по числу вершин в V. Пусть $G(V \cup W, E)$ – двудольный граф с $|V| = m > 0$, удовлетворяющий условию теоремы. При $m = 1$ единственная вершина из V инцидентна хотя бы одному ребру, которое (вместе со своими концами) и является нужным паросочетанием. Базис индукции выполняется.

Индуктивный переход. Пусть теорема верна для графов с числом вершин в доле V, равным $1, 2, \ldots, k-1$; докажем её для произвольного графа $G = (V \cup W, E)$, у которого $|V| = k$, $k > 1$. Отдельно рассмотрим два следующих случая.

1) Для каждого собственного подмножества V' в V выполняется строгое неравенство $|V'| < |N_G(V')|$ (условие теоремы выполняется «с запасом»). Возьмём в G произвольную вершину v и ребро $e' = (v, w)$, соединяющее вершину v из V с некоторой вершиной w из W. Затем удалим из G вершины v, w и все рёбра, инцидентные этим вершинам; в результате получим некоторый двудольный граф $G_1 = (V_1 \cup W_1, E_1)$, у которого $V_1 = V \setminus \{v\}$, $W_1 = W \setminus \{w\}$ и $E_1 = E \setminus \{e : e$ инцидентно хотя бы одной из вершин $v, w\}$, а $|V_1| = k - 1$. Пусть V_1' — произвольное подмножество множества V_1. Так как для исходного графа G выполняется условие $|V_1'| < |N_G(V_1')|$ («с запасом»!), а из W удалена только одна вершина, то для полученного графа G_1 выполняется условие $|V_1'| \leqslant |N_{G_1}(V_1')|$ и по индуктивному предположению в графе G_1 существует паросочетание P, покрывающее V_1. Добавив к P ребро e', получим нужное паросочетание для исходного графа G.

2) Во множестве V существует такое непустое собственное подмножество V_0, для которого верно равенство

$$|V_0| = |N_G(V_0)|. \tag{1}$$

Пусть G' и G'' — подграфы графа G, порождённые множествами вершин $V_0 \cup N_G(V_0)$ и $(V \cup W) \setminus (V_0 \cup N_G(V_0))$ соответственно (если V' — подмножество вершин графа G, то подграф данного графа, порождённый множеством вершин V', есть подграф графа G, содержащий в качестве вершин все вершины из V', а в качестве рёбер — все рёбра исходного графа G, концами которых являются вершины из V').

Рассмотрим подграф G'. Для любого подмножества $V' \subseteq V_0$ имеем $N_G(V') = N_{G'}(V')$ и, следовательно, $|V'| \leqslant |N_{G'}(V')|$.

По индуктивному предположению в G' существует паросочетание, покрывающее V_0; построим его.

Обратимся к подграфу G''. Для любого подмножества $V' \subseteq V \setminus V_0$ выполняются соотношения

$$|V_0| + |V'| = |V_0 \cup V'| \leqslant |N_G(V_0 \cup V')| = |N_G(V_0)| + |N_{G''}(V')|$$

и верно равенство (1). Следовательно, $|V'| \leqslant |N_{G''}(V')|$ и по индуктивному предположению в графе G'' существует паросочетание, покрывающее $V \setminus V_0$. Объединяя это паросочетание с построенным для подграфа G', получим паросочетание в графе G, покрывающее V. Теорема доказана.

Пусть $A = \{a_1, \ldots, a_n\}$ — конечное непустое множество, а $S = (S_1, \ldots, S_k)$ — k-членное семейство его подмножеств (некоторые из них могут пересекаться и даже совпадать). Всякое подмножество $T = \{a_{i_1}, \ldots, a_{i_k}\}$ элементов из A называется *трансверсалью* (или *системой различных представителей*) семейства S, если $a_{i_j} \in S_j$, $1 \leqslant j \leqslant k$, $i_j \neq i_{j'}$ при $j \neq j'$.

Теорема 7 (Холла). *Пусть A — непустое конечное множество, $S = (S_1, S_2, \ldots, S_k)$ — семейство его подмножеств. Для существования трансверсали семейства S необходимо и достаточно, чтобы для каждого $j = 1, \ldots, k$ объединение $S_{i_1} \cup S_{i_2} \cup \ldots \cup S_{i_j}$ любых j подмножеств из S содержало не менее j различных элементов.*

Если теперь сравнить две последние теоремы, то нетрудно заметить, что теорема 6 является переформулировкой теоремы Холла на языке графов (элементы множества A образуют одну долю W вершин графа, а подмножества семейства S образуют другую долю V — покрываемую паросочетанием).

§ 7. Сети. Потоки в сетях. Теорема Форда–Фалкерсона

Граф, некоторые вершины которого выделены, называется *сетью*. Выделенные вершины называются *полюсами* сети. Например, дерево с корнем можно рассматривать как однополюсную сеть.

Вершины, отличные от полюсов, называются *внутренними* вершинами сети. Ребро, инцидентное хотя бы одному полюсу, называется *полюсным* ребром; остальные рёбра называются *внутренними*.

Для удобства изложения в этом параграфе введём понятие псевдоцепи.

Всякому ориентированному (или частично ориентированному) графу $G = (V, E)$ поставим в соответствие *соотнесённый* неориентированный граф $\widehat{G} = (V, \widehat{E})$, в котором набор $\widehat{E}$ содержит все те же пары вершин, что и E, но все пары в $\widehat{E}$ неупорядоченные ($\widehat{G}$ получается из G «стиранием» ориентации всех ориентированных в G рёбер).

Псевдоцепью (*простой псевдоцепью*) графа G будем считать всякую последовательность вершин и рёбер графа G, образующую цепь (соответственно простую цепь) в соотнесённом графе $\widehat{G}$. Например, рёбра e_1 и e_2 графа, изображённого на рис. 7, вместе со своими концами образуют псевдоцепь, связывающую вершины v_1 и v_3.

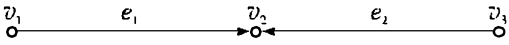


Рис. 7

Две вершины ориентированного (или частично ориентированного) графа G *связаны*, если они связаны в соотнесённом графе $\widehat{G}$. В соответствии с таким определением связанных вершин осуществляется и разбиение G на компоненты связности.

Далее будем рассматривать двухполюсные сети без петель и без изолированных внутренних вершин (кратные рёбра допускаются). Один из полюсов будем считать *входным* (или *входом, истоком*), второй — *выходным* (или *выходом, стоком*).

Пусть S — произвольная частично ориентированная сеть, каждому ребру e которой приписано неотрицательное число $c(e)$ — *пропускная способность*. *Потоком в сети* S называется пара (функций) (f, ω), где ω — некоторая ориентация всех неориентированных рёбер сети, а $f(e)$ – заданная на множестве всех рёбер функция с неотрицательными значениями, не превосходящими пропускных способностей, и такая, что в каждой внутренней вершине v выполняется закон Кирхгофа, согласно которому сумма значений функции на рёбрах, исходящих из вершины, равна сумме значений функции на рёбрах, входящих в вершину. Другими словами, для $f(e)$ выполнены условия:

1) $0 \leqslant f(e) \leqslant c(e)$ для всех рёбер сети;

2) для всех внутренних вершин $R(v) = 0$, где

$$R(v) = \sum_{e \in \Gamma(v)} f(e) - \sum_{e \in \Gamma'(v)} f(e),$$

а $\Gamma(v)$ (соответственно $\Gamma'(v)$) — множество всех рёбер, исходящих из v (соответственно входящих в v) при ориентации ω.

Пусть α_S и β_S — соответственно входной и выходной полюсы сети S. Поскольку сумма значений $R(v)$ по всем вершинам сети, включая полюсы, равна нулю (каждое ребро является исходящим для одной вершины и входящим для другой), то

$$R(\alpha_S) = -R(\beta_S).$$

Значение $R = R(\alpha_S)$ называется *величиной* потока.

Из таких же рассуждений следует, что если сеть допускает поток ненулевой величины, то полюсы сети находятся в одной компоненте связности. Если сеть изображает какую-либо проводящую систему (сеть дорог, трубопроводов и т. п.), то R определяет величину потока, входящего в одном полюсе и выходящего из другого, т. е. величину потока, проходящего через сеть.

Добавим к сети S дополнительное, *источниковое* ребро, связывающее полюсы и ориентированное от выхода β_S к входу α_S сети S; на этом добавленном ребре положим $f = R$ (для этого ребра допускается $f < 0$). В таком случае закон Кирхгофа будет выполняться для всех вершин сети, а R будет определять величину *циркуляции* через сеть.

Поставим задачу определения максимальной величины $R_{\max}$ потока через сеть S при заданных значениях пропускных способностей рёбер. Ответ может быть получен в терминах сечений сети.

Сечением сети называется множество рёбер, при удалении которого сеть становится несвязной, причём полюсы попадают в разные компоненты связности. Ясно, что каждая простая псевдоцепь, соединяющая α_S и β_S (а тем более каждый путь из α_S в β_S), проходит хотя бы через одно ребро сечения. В сети на рис. 8 примерами сечений являются $\{d, e, f\}$, $\{b, c, e, g, h\}$, $\{d, g, h, i\}$.

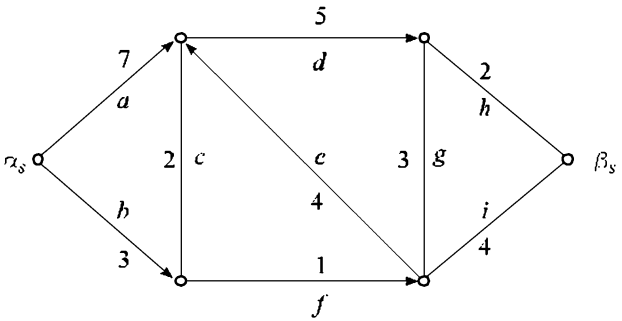


Рис. 8

Сечение будем называть *простым*, если при удалении любого ребра оно перестаёт быть сечением. Так $\{d, e, f\}$ и $\{b, c, e, g, h\}$ являются простыми сечениями, тогда как $\{d, g, h, i\}$ не является таковым. Очевидно, что для каждого ребра простого сечения можно указать простую псевдоцепь, соединяющую α_S и β_S, которая проходит через это ребро, но не проходит через другие рёбра данного сечения.

Лемма 3. *Если в связной сети удаляется простое сечение, то сеть распадается ровно на две компоненты (не связанные между собой): левую, содержащую α_S, и правую, содержащую β_S.*

Доказательство. Действительно, то, что сеть S распадётся на несвязные компоненты, а полюсы α_S и β_S окажутся в разных компонентах связности при удалении рёбер простого сечения W, ясно из определения простого сечения. Убедимся, что этих компонент будет две — левая и правая.

Пусть v — произвольная вершина. Покажем, что она останется связанной псевдоцепью либо с α_S, либо с β_S. Возьмём (в исходной сети S) псевдоцепь Z, связывающую v с концом v' какого-нибудь ребра e из простого сечения, так чтобы среди её внутренних вершин (отличных от концов v и v') не было концов рёбер из простого сечения (в силу связности сети такая псевдоцепь существует). Поскольку e принадлежит простому сечению, то существует простая псевдоцепь Z', соединяющая α_S и β_S и содержащая только данное ребро e из простого сечения. Отрезок Z'' этой цепи, соединяющий v' либо с α_S, либо с β_S, не содержит рёбер из простого сечения. Но в таком случае, очевидно, из Z и Z'' можно составить псевдоцепь, связывающую v с одним из полюсов и остающуюся в одной из компонент связности после удаления рёбер простого сечения. Лемма доказана.

Согласно лемме 3 каждое ребро простого сечения связывает вершины из двух разных частей: левой и правой. Будем называть ребро сечения *прямым*, если оно в сети не ориентировано или ориентировано слева направо, и *обратным* в противном случае. Будет ли ориентированное ребро прямым или обратным, вообще говоря зависит от выбора сечения. Так, в последнем примере (рис. 8) ребро e в сечениях $\{d, e, f\}$ и $\{b, c, e, g, h\}$ — обратное, а в сечении $\{a, c, e, g, i\}$ — прямое.

Каждому простому сечению W припишем *пропускную способность* $c(W)$, равную сумме пропускных способностей всех его прямых рёбер. В том же примере (рис. 8) сечение $\{d, e, f\}$ имеет пропускную способность $5 + 1 = 6$, а сечение $\{b, c, e, g, h\}$ имеет пропускную способность $3 + 2 + 3 + 2 = 10$. Если сеть

несвязна и её полюсы находятся в разных компонентах связности, то естественно считать (единственным) простым сечением сети пустое множество, а его пропускную способность нулевой.

Теорема 8 (Форда–Фалкерсона). *Максимальная величина потока $R_{\max}$ через сеть S равна минимальной из пропускных способностей $c_{\min}$ её простых сечений.*

Доказательство. Докажем сначала, что $R_{\max} \leqslant c_{\min}$. Для этого достаточно показать, что для любого потока и любого сечения $R \leqslant c(W)$ (из физических соображений это неравенство достаточно очевидно).

Просуммируем $R(\alpha)$ по всем вершинам α левой (относительно простого сечения W) компоненты сети. Эта сумма, с одной стороны, равна $R = R(\alpha_S)$ (с учётом выполнения закона Кирхгофа для внутренних вершин сети), а с другой стороны, она равна алгебраической сумме величин потоков, идущих через сечение слева направо (потоки по прямым рёбрам суммируются со знаком плюс, а по обратным — со знаком минус; последнее утверждение усматривается легко, если учесть, что по каждому ребру из одной его вершины «вытекает» некоторая часть потока, а в другую его вершину «втекает» точно такая же часть потока). Поскольку $f(e) \leqslant c(e)$, то отсюда и следует требуемое неравенство. Ясно также, что равенство $R = c(W)$ может достигаться только в случае, когда все прямые рёбра сечения ориентированы слева направо (при ориентации ω) и для них $f(e) = c(e)$, а для всех обратных рёбер $f(e) = 0$.

Докажем теперь, что в сети S можно создать поток величины $c_{\min}$. Доказательство проведём индукцией по числу рёбер в сети S.

Если число рёбер равно нулю, т. е. S состоит из двух изолированных полюсов, то очевидно, что $R = c_{\min} = 0$.

Предположим, что для всякой сети S, содержащей менее m рёбер, утверждение уже доказано.

В таком случае имеет место следующее утверждение (*): *в сети S можно создать поток любой величины, меньшей, чем $c_{\min}$.*

Действительно, пусть R — произвольное число, удовлетворяющее условию

$$0 \leqslant R < c_{\min}.$$

Отыскав в сети S простое сечение с пропускной способностью $c_{\min}$ и снизив её до R (уменьшая для этого пропускные способности некоторых прямых рёбер сечения), мы получим сеть, в которой согласно индуктивному предположению можно создать

поток величины R. Но этот же самый поток будет потоком и в сети S.

Используя данное утверждение (*), докажем утверждение для произвольной сети S с m рёбрами. Простые сечения сети, имеющие пропускную способность $c_{\min}$, будем называть *минимальными*. В сети S можно уменьшить пропускные способности некоторых рёбер так, что в изменённой сети — назовём её *приведённой* — будут выполнены следующие условия:

1) пропускная способность каждого простого сечения не меньше $c_{\min}$;

2) каждое ребро является прямым в некотором минимальном сечении или имеет нулевую пропускную способность.

Для получения приведённой сети следует в каком-либо порядке просматривать все рёбра сети S и уменьшать (быть может, до нуля) пропускные способности тех из них, которые не удовлетворяют условию 2), но с таким расчётом, чтобы условие 1) не нарушалось. (Вообще говоря, в зависимости от выбранного порядка могут получаться различные приведённые сети.)

Очевидно, что любой поток в приведённой сети является потоком и в исходной сети S. Поэтому достаточно установить, что поток величины $c_{\min}$ можно создать в приведённой сети.

Если приведённая сеть содержит рёбра с нулевой пропускной способностью, то, удалив их, мы придём к сети, содержащей менее m рёбер и допускающей те же самые потоки. Очевидно (по построению), что в этой сети каждое простое сечение имеет пропускную способность не меньше $c_{\min}$. Используя предположение индукции, а при необходимости и утверждение (*), создадим в полученной сети поток величины $c_{\min}$; он же будет потоком и в самой приведённой сети.

Будем считать теперь, что в приведённой сети нет рёбер с нулевой пропускной способностью, т. е. каждое её ребро является прямым в некотором минимальном сечении (свойство 2)) и, следовательно, принадлежит некоторой простой псевдоцепи, соединяющей α_S и β_S (иначе сечение не простое, а тем самым и не минимальное).

Рассмотрим два случая:

а) каждая простая псевдоцепь, соединяющая α_S и β_S в приведённой сети, содержит не более двух рёбер;

б) в приведённой сети имеется простая псевдоцепь, связывающая α_S и β_S и содержащая не менее трёх рёбер.

Если имеет место случай а), то, очевидно, все рёбра сети полюсные, т. е. каждое ребро инцидентно хотя бы одному полюсу, и либо не ориентированы, либо ориентированы слева направо

(в противном случае получили бы ребро, которое было бы обратным в любом сечении — из-за того, что оно полюсное, — а потому его пропускная способность была бы уменьшена до нуля в соответствии с условием 2) и это ребро было бы выброшено при построении приведённой сети). При этом приведённая сеть представляет собой параллельное соединение некоторого числа сквозных рёбер (т. е. таких, каждое из которых соединяет полюсы), и некоторого числа подсетей (по одной на каждую внутреннюю вершину) весьма простой конструкции: каждая из них является последовательным соединением двух своих частей, в которых все рёбра соединены параллельно (рис. 9).

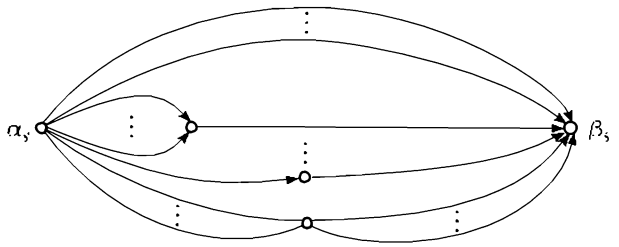


Рис. 9

Легко видеть, что простыми сечениями этой сети являются такие и только такие множества, которые содержат все сквозные рёбра, а из каждой подсети либо все рёбра из её левой части и ни одного ребра из правой, либо наоборот. Отсюда уже нетрудно заключить, что для каждой подсети сумма пропускных способностей рёбер из её левой части равна сумме пропускных способностей рёбер из её правой части. (Действительно, если бы какая-нибудь из этих сумм — скажем, первая — была бы меньше, то, взяв минимальное сечение, содержащее какое-либо ребро из правой части подсети, и заменив в нём все рёбра из правой части рёбрами из левой части, мы получили бы простое сечение с пропускной способностью, меньшей $c_{\min}$, что в силу свойства 1) невозможно.) Но тогда пропускные способности всех простых сечений одинаковы и равны $c_{\min}$. Поэтому в приведённой сети можно создать поток величины $c_{\min}$ — следует только загрузить каждое ребро вплоть до его пропускной способности.

Если имеет место случай б), то, очевидно, приведённая сеть содержит внутреннее ребро (не инцидентное ни одному из полюсов). Рассмотрим следующее преобразование приведённой сети. Выделим минимальное сечение, содержащее внутреннее ребро. Заметим, что ни входная звезда Z_{α_S} (т. е. подсеть, образован-

ная полюсом α_S и инцидентными ему рёбрами), ни выходная звезда Z_{β_S} не содержатся целиком в этом сечении (поскольку указанные звёзды сами — без внутреннего ребра — составляют сечения). Каждое ребро $e_i = (\alpha_i, \beta_i)$ этого сечения заменим двумя последовательно соединёнными рёбрами (α_i, γ_i) и (γ_i, β_i) (все γ_i — новые вершины), имеющими ту же пропускную способность, что и ребро e_i. Отождествив теперь все вершины γ_i, т. е. склеив их в одну вершину, мы получим сеть, представляющую собой последовательное соединение двух сетей S_1 и S_2. Это следует из леммы 3, так как согласно этой лемме при удалении всех рёбер (α_i, β_i) выделенного минимального сечения исходная сеть распадается на две связные компоненты: левую подсеть S_1', содержащую α_S, и правую подсеть S_2', содержащую β_S. Левая подсеть S_1 получается из S_1' добавлением рёбер (α_i, γ_i), а правая подсеть S_2 получается из S_2' добавлением рёбер (γ_i, β_i).

Каждая из сетей S_1 и S_2 содержит менее m рёбер. Например, S_1 не может содержать все рёбра из Z_{β_S}. Действительно, как сказано выше, в Z_{β_S} входит некоторое ребро e, отсутствующее в выделенном минимальном сечении. Не может это ребро e входить и в S_1', поскольку его концом является полюс β_S, принадлежащий S_2'. Из аналогичных рассуждений следует, что и в S_2 отсутствует хотя бы одно ребро из Z_{α_S}. Всякое простое сечение каждой из подсетей S_1 и S_2 имеет пропускную способность не меньше $c_{\min}$ (так же, как прообраз этого сечения в приведённой сети). Поэтому согласно индуктивному предположению в каждой из этих подсетей можно создать поток величины $c_{\min}$.

Поскольку выбранное сечение минимальное, то его пропускная способность равна $c_{\min}$, и чтобы получить величину потока $c_{\min}$, надо загрузить все прямые рёбра до их пропускных способностей — только тогда величина потока сравняется с пропускной способностью рассматриваемого минимального сечения. Следовательно, полученные по индуктивному предположению потоки загружают все прямые рёбра (α_i, γ_i) и (γ_i, β_i) до их пропускных способностей. Более того, каждое ребро нашей приведённой сети является прямым в некотором минимальном сечении и потому — согласно приведённым выше рассуждениям применительно к произвольному минимальному сечению — окажется загруженным до пропускной способности этого сечения. Отсюда ясно, что прообразы полученных потоков (в S_1 и в S_2) в исходной приведённой сети также составляют поток и его величина равна $c_{\min}$. Теорема доказана.

Глава III

БУЛЕВЫ ФУНКЦИИ И ФОРМУЛЫ

§ 1. Булевы функции. Элементарные булевы функции

Функция $f(x_1, \ldots, x_n)$ называется *булевой* (или *булевской*, или *функцией алгебры логики*, или *логической*), если её переменные и сама функция принимают значения из множества $E_2 = \{0, 1\}$.

Булева функция от n переменных может быть задана таблицей, в которой перечислены всевозможные булевы наборы из нулей и единиц длины n и для каждого набора указано значение функции на этом наборе (табл. 1).

Таблица 1

x_1	x_2	...	x_n	$f(x_1, \ldots, x_n)$
0	0	...	0	$f(0, \ldots, 0)$
0	0	...	1	$f(0, \ldots, 1)$
...	...	...	...	...
1	1	...	0	$f(1, \ldots, 0)$
1	1	...	1	$f(1, \ldots, 1)$

В этой таблице обычно используется естественное расположение наборов: если набор рассматривать как двоичную запись числа, то расположение наборов сверху вниз соответствует расположению чисел $0, 1, \ldots, 2^n - 1$. Таблицы, задающие булевы функции от n переменных, отличаются лишь последним столбцом. Поскольку имеется ровно 2^{2^n} различных булевых столбцов высоты 2^n, то и число булевых функций от n переменных равно 2^{2^n}.

Булева функция $f(x_1, \ldots, x_n)$ *существенно зависит* от переменной x_i, если существуют такие значения $\alpha_1, \ldots, \alpha_{i-1}, \alpha_{i+1}, \ldots$ $\ldots, \alpha_n$ переменных $x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n$, что $f(\alpha_1, \ldots, \alpha_{i-1}, 0,$

$\alpha_{i+1}, \ldots, \alpha_n) \neq f(\alpha_1, \ldots, \alpha_{i-1}, 1, \alpha_{i+1}, \ldots, \alpha_n)$; переменная x_i в этом случае называется *существенной*. Если переменная x_i не является существенной, то она называется *несущественной*, или *фиктивной*.

Пусть для функции $f(x_1, \ldots, x_n)$ переменная x_i является фиктивной. Возьмём таблицу, задающую $f(x_1, \ldots, x_n)$. Вычеркнем из неё все строки вида $(\alpha_1, \ldots, \alpha_{i-1}, 1, \alpha_{i+1}, \ldots, \alpha_n, f(\alpha_1, \ldots$ $\ldots, \alpha_{i-1}, 1, \alpha_{i+1}, \ldots, \alpha_n))$, а также столбец, отвечающий переменной x_i. Полученная таблица будет определять некоторую булеву функцию $g(x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n)$. Говорят, что функция $g(x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n)$ получается из функции $f(x_1, \ldots, x_n)$ *удалением фиктивной переменной* x_i, а также, что функция $f(x_1, \ldots, x_n)$ получается из $g(x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n)$ *введением фиктивной переменной* x_i.

Булевы функции f_1 и f_2 называются *равными*, если функцию f_1 можно получить из функции f_2 введением и (или) удалением фиктивных переменных.

Далее предполагаем, что с заданием некоторой булевой функции f заданы все равные ей функции, и для обозначения равных функций будем использовать один и тот же функциональный символ.

Приведём булевы функции от одной переменной и некоторые функции от двух переменных, которые будем часто использовать далее; эти и другие функции (от одной и от двух переменных) обычно называют *элементарными*. Имеется четыре булевы функции от одной переменной (табл. 2).

Этими функциями являются константы 0 и 1, значения которых не зависит от значений переменной, *тождественная функция* x, повторяющая значение переменной, и функция $\overline{x}$,

Таблица 2

x	0	1	x	$\overline{x}$
0	0	1	0	1
1	0	1	1	0

Таблица 3

x_1	x_2	$x_1 \& x_2$	$x_1 \vee x_2$	$x_1 \to x_2$	$x_1 \oplus x_2$	$x_1 \sim x_2$
0	0	0	0	1	0	1
0	1	0	1	1	1	0
1	0	0	1	0	1	0
1	1	1	1	1	0	1

принимающая значение, противоположное значению переменной. Функция $\overline{x}$ называется *отрицанием* (или *инверсией*).

Приведённые в табл. 3 булевы функции от двух переменных называются соответственно *конъюнкцией*, *дизъюнкцией*, *импликацией*, *суммой по модулю два* и *эквивалентностью*.

§ 2. Формулы и функции, реализуемые формулами. Простейшие эквивалентности

Так же, как и в элементарной алгебре, исходя из элементарных булевых функций, можно строить формулы. Обозначим через P_2 множество всех булевых функций.

Пусть B — некоторое (не обязательно конечное) подмножество функций из P_2. Дадим индуктивное определение формулы над B.

а) *Базис индукции*. Каждое выражение $f(x_1, \ldots, x_n)$, где $f \in B$, называется *формулой над* B.

б) *Индуктивный переход*. Пусть $f_0(x_1, \ldots, x_m)$ — функция из B и $\Phi_1, \ldots, \Phi_m$ — выражения, являющиеся либо формулами над B, либо символами переменных. Тогда выражение $f_0(\Phi_1, \ldots, \Phi_m)$ называется *формулой над* B.

Если подмножество B содержит элементарные функции $f_1 = 0$, $f_2 = 1$, $f_3(x) = x$, $f_4(x) = \overline{x}$, $f_5(x, y) = x\&y$, $f_6(x, y) = x \vee y$ и т. д., причём символы этих функций являются внешними в выражениях, обозначающих формулы над B, то такие формулы часто записывают как 0, 1, (x), $(\overline{x})$, $(x\&y)$, $(x \vee y)$ и т. д. Когда в формулы (x), $(\overline{x})$, $(x\&y)$, $(x \vee y)$ и т. д. вместо x и y подставляют (при индуктивном переходе) выражения Φ_1 и Φ_2, то новые формулы обозначают соответственно через (Φ_1), $(\overline{\Phi_1})$, $(\Phi_1\&\Phi_2)$, $(\Phi_1 \vee \Phi_2)$ и т. д. Формулы, которые используются при построении формулы Φ в соответствии с вышеприведенным индуктивным определением, называются *подформулами* формулы Φ.

Примеры формул над множеством элементарных функций:

1) $(((x_1\&x_2) \oplus x_1) \oplus x_2)$;

2) $(\overline{x}_1\&(x_2 \oplus x_3))$;

3) $(x_1 \vee ((x_2 \to x_3)\&(x_3 \to x_2)))$.

Сопоставим каждой формуле $\Phi(x_1, \ldots, x_n)$ (в скобках указаны переменные, входящие в формулу) над B функцию $f(x_1, \ldots, x_n)$, опираясь на индуктивное определение формулы.

а) *Базис индукции*. Если $\Phi(x_1, \ldots, x_n)$ совпадает с $f(x_1, \ldots, x_n)$, где $f \in B$, то формуле $\Phi(x_1, \ldots, x_n)$ сопоставим функцию $f(x_1, \ldots, x_n)$.

б) *Индуктивный переход.* Если $\Phi(x_1, \ldots, x_n)$ совпадает с $f_0(A_1, \ldots, A_m)$, где f_0 — символ функции из B, $A_i (i = 1, \ldots, m)$ — либо формула над B, либо символ переменной $x_{j(i)}$, то тогда в первом случае, по предположению индукции, A_i сопоставляется функция f_i из P_2, а во втором случае A_i сопоставляется тождественная функция $f_i = x_{j(i)}$. Сопоставим формуле $\Phi(x_1, \ldots, x_n)$ функцию $f(x_1, \ldots, x_n) = f_0(f_1, \ldots, f_m)$; значение функции f на каждом наборе $(\alpha_1, \ldots, \alpha_n)$ находится как значение функции f_0 на наборе $(f_1(\alpha_1, \ldots, \alpha_n), \ldots, f_m(\alpha_1, \ldots, \alpha_n))$.

Если функция f сопоставлена формуле Φ, то говорят, что формула Φ *реализует функцию* f.

Как нетрудно убедиться, формулы, приведённые выше в качестве примера, реализуют соответственно функции $f_1(x_1, x_2)$, $f_2(x_1, x_2, x_3)$, $f_3(x_1, x_2, x_3)$, приведённые в табл. 4 и 5.

Таблица 4

x_1	x_2	f_1
0	0	0
0	1	1
1	0	1
1	1	1

Таблица 5

x_1	x_2	x_2	f_2	f_3
0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	0	0
1	0	0	0	0
1	0	1	0	0
1	1	0	0	0
1	1	1	0	0

Формулы Φ_1 и Φ_2 над B называются *эквивалентными*, если они реализуют равные функции f_1 и f_2.

В рассмотренном примере вторая и третья формулы эквивалентны.

Приведём некоторые эквивалентности, характеризующие свойства элементарных функций. Обозначим через $(x_1 \circ x_2)$ любую из функций $(x_1 \& x_2)$, $(x_1 \vee x_2)$, $(x_1 \oplus x_2)$, ниже называемых также *операциями*.

1. Операция $(x_1 \circ x_2)$ обладает свойством ассоциативности:

$$((x_1 \circ x_2) \circ x_3) = (x_1 \circ (x_2 \circ x_3)).$$

2. Операция $(x_1 \circ x_2)$ обладает свойством коммутативности:

$$(x_1 \circ x_2) = (x_2 \circ x_1).$$

3. Для конъюнкции и дизъюнкции выполняются дистрибутивные законы:

$$((x_1 \vee x_2)\&x_3) = ((x_1\&x_3) \vee (x_2\&x_3))$$

— дистрибутивность конъюнкции относительно дизъюнкции;

$$((x_1\&x_2) \vee x_3) = ((x_1 \vee x_3)\&(x_2 \vee x_3))$$

— дистрибутивность дизъюнкции относительно конъюнкции.

4. $x = \overline{\overline{x}}$ — закон двойного отрицания.

5. $\left.\begin{array}{l}\overline{(x_1\,\&\,x_2)} = (\overline{x}_1 \vee \overline{x}_2),\\ \overline{(x_1 \vee x_2)} = (\overline{x}_1\,\&\,\overline{x}_2)\end{array}\right\}$ — законы де Моргана.

6. $\left.\begin{array}{l}(x\&(x \vee y)) = x,\\ (x \vee (x\&y)) = x\end{array}\right\}$ — законы поглощения.

7. $(x\&\overline{x}) = 0$ — закон противоречия.

8. $(x \vee \overline{x}) = 1$ — закон исключённого третьего.

9. $\left.\begin{array}{l}(x\&x) = x,\\ (x \vee x) = x\end{array}\right\}$ — идемпотентность.

10. $\left.\begin{array}{l}(x\&0) = 0,\\ (x\&1) = x,\\ (x \vee 0) = x,\\ (x \vee 1) = 1\end{array}\right\}$ — законы умножения на константу и сложения с константой.

Все перечисленные равенства 1–10 легко проверяются вычислением значений функций в левой и в правой частях равенства на каждом наборе значений переменных.

С целью упрощения записи формул условимся, что операция «&» сильнее операции «∨», т. е. если нет скобок, то сначала выполняется операция «&», а затем операция «∨». Кроме того, в силу закона ассоциативности для $(x_1 \circ x_2)$ можно вместо формул $((x_1 \circ x_2) \circ x_3)$, $(x_1 \circ (x_2 \circ x_3))$ использовать выражение $(x_1 \circ x_2 \circ x_3)$.

В формулах, в которых внешняя операция (функция) является либо конъюнкцией, либо дизъюнкцией, либо сложением по модулю два, либо импликацией, либо эквивалентностью (либо другой элементарной функцией от двух переменных), внешние скобки будем опускать. Опускаются внешние скобки также в выражении, над которым стоит знак «$\overline{\ }$», например, пишем $x_1 \to x_2$

вместо $\overline{(x_1 \to x_2)}$. В дальнейшем будем употреблять также следующие обозначения:

$$\&_{i=1}^{m} x_i = x_1 \& x_2 \& \ldots \& x_m,$$

$$\bigvee_{i=1}^{m} x_i = x_1 \vee x_2 \vee \ldots \vee x_m,$$

$$x_1 x_2 = x_1 \cdot x_2 = x_1 \& x_2.$$

Очевидно, справедливо следующее

Утверждение. *Если Φ' — подформула формулы Φ, то при замене любого её вхождения на эквивалентную формулу Ψ' формула Φ перейдёт в формулу Ψ, которая эквивалентна Φ.*

(Формально данное утверждение можно доказать индукцией по числу функциональных символов в формуле.)

Данное утверждение вместе с тождествами для элементарных функций, к которым присоединяются все тождества, получаемые подстановкой вместо переменных x_1, x_2, x_3, x, y любых переменных и формул, позволяет осуществлять эквивалентные преобразования и, в частности, упрощать исходные формулы.

Примеры:

1) $x_1 \vee x_1 x_2 = x_1 \cdot 1 \vee x_1 x_2 = x_1(1 \vee x_2) = x_1 \cdot 1 = x_1$;

2) $x_1 x_2 \vee \overline{x}_1 x_2 = (x_1 \vee \overline{x}_1) x_2 = 1 \cdot x_2 = x_2$.

§ 3. Разложение булевых функций. Дизъюнктивные нормальные формы

Введём обозначение:

$$x^{\sigma} = \begin{cases} \overline{x} \text{ при } \sigma = 0, \\ x \text{ при } \sigma = 1. \end{cases}$$

Очевидно, $x^{\sigma} = 1$ тогда и только тогда, когда $x = \sigma$.

Теорема 9 (о разложении функций по переменным). *Каждую булеву функцию $f(x_1, \ldots, x_n)$ при любом m $(1 \leqslant m \leqslant n)$ можно представить в форме*

$$f(x_1, \ldots, x_m, x_{m+1}, \ldots, x_n) = \bigvee_{(\sigma, \ldots, \sigma_m)} x_1^{\sigma_1} \& \ldots \& x_m^{\sigma_m} \& f(\sigma_1, \ldots, \sigma_m, x_{m+1}, \ldots, x_n), \quad (1)$$

где дизъюнкция берётся по всевозможным наборам значений переменных $x_1, \ldots, x_m$.

Доказательство. Возьмём произвольный набор значений переменных $(\alpha_1, \dots, \alpha_n)$ и покажем, что левая и правая части соотношения (1) принимают на нём одно и то же значение. Левая часть даёт $f(\alpha_1, \dots, \alpha_n)$. Правая —

$$\bigvee_{(\sigma_1,\dots,\sigma_m)} \alpha_1^{\sigma_1} \& \dots \& \alpha_m^{\sigma_m} \& f(\sigma_1, \dots, \sigma_m, \alpha_{m+1}, \dots, \alpha_n) =$$

$$= \alpha_1^{\alpha_1} \& \dots \& \alpha_m^{\alpha_m} \& f(\alpha_1, \dots, \alpha_m, \alpha_{m+1}, \dots, \alpha_n) = f(\alpha_1, \dots, \alpha_n).$$

Теорема доказана.

Представление (1) называется *разложением функции по переменным* $x_1, \dots, x_m$. Выражение $f(\sigma_1, \dots, \sigma_m, x_{m+1}, \dots, x_m)$ естественным образом можно рассматривать как формулу над множеством булевых функций $\{0, 1, f(x_1, \dots, x_n)\}$; реализуемую этой формулой функцию обычно считают подфункцией функции $f(x_1, \dots, x_n)$, получающейся из $f(x_1, \dots, x_n)$ подстановкой констант $\sigma_1, \dots, \sigma_m$ вместо переменных $x_1, \dots, x_m$.

В качестве следствий из последней теоремы получаем два специальных случая разложения.

1) Разложение по переменной:

$$f(x_1, \dots, x_n) = x_n \& f(x_1, \dots, x_{n-1}, 1) \vee \overline{x}_n \& f(x_1, \dots, x_{n-1}, 0).$$

2) Разложение по всем n переменным:

$$f(x_1, \dots, x_n) = \bigvee_{(\sigma_1,\dots,\sigma_n)} x_1^{\sigma_1} \& \dots \& x_n^{\sigma_n} \& f(\sigma_1, \dots, \sigma_n).$$

При $f(x_1, \dots, x_n) \not\equiv 0$ оно может быть преобразовано к виду

$$f(x_1, \dots, x_n) = \bigvee_{\substack{(\sigma_1,\dots,\sigma_n): \\ f(\sigma_1,\dots,\sigma_n)=1}} x_1^{\sigma_1} \& \dots \& x_n^{\sigma_n}. \qquad (2)$$

Последнее разложение называется *совершенной дизъюнктивной нормальной формой* (совершенной д.н.ф., или с.д.н.ф.). Дизъюнкция в (2) берётся по всем наборам значений переменных, на которых $f(x_1, \dots, x_n)$ обращается в 1.

Теорема 10. *Каждая булева функция может быть выражена в виде формулы через отрицание, дизъюнкцию и конъюнкцию.*

Доказательство. 1) Пусть $f(x_1, \dots, x_n) \equiv 0$. Тогда, очевидно, $f(x_1, \dots, x_n) = x_1 \& \overline{x}_1$.

2) Пусть $f(x_1, \dots, x_n) \not\equiv 0$. Представим f в виде (2).

В обоих случаях функция f выражается в виде формулы через отрицание, дизъюнкцию и конъюнкцию. Теорема доказана.

§ 4. Полнота систем булевых функций. Представление булевых функций полиномами Жегалкина

Система булевых функций $\{f_1, f_2, \ldots, f_s, \ldots\}$ из P_2 называется *(функционально) полной*, если любая булева функция может быть реализована формулой над этой системой.

Теорема 11 (о полноте двух систем). *Пусть даны две системы функций из* P_2:

$$B_1 = \{f_1, f_2, \ldots\},$$
$$B_2 = \{g_1, g_2, \ldots\},$$

причем первая система полна и каждая её функция реализуется формулой над второй системой. Тогда вторая система также является полной.

Доказательство. Для доказательства теоремы покажем, что для любой формулы над B_1 существует эквивалентная ей формула над B_2. Доказательство последнего утверждения проведём индукцией по сложности формул, понимая под сложностью формулы число функциональных символов в ней.

Базис индукции. Пусть Φ — формула сложности 1 над B_1, т. е. $\Phi = f_i(x_{j_1}, \ldots, x_{j_m})$. По условию теоремы f_i можно реализовать формулой над B_2, а значит, можно построить формулу над B_2, эквивалентную Φ (говоря об эквивалентности формул, в данном случае формально можно иметь в виду формулы над $B_1 \cup B_2$).

Индуктивный переход. Пусть утверждение верно для всех формул над B_1 сложности не больше k; докажем его для произвольной формулы Φ над B_1 сложности $k+1$.

Пусть $\Phi = f_i(A_1, \ldots, A_m)$, где A_j — либо формула над B_1, либо символ переменной ($j = 1, \ldots, m$). Возьмём формулу над B_1 вида $f_i(y_1, \ldots, y_m)$, по условию теоремы существует эквивалентная ей формула $\Psi(y_1, \ldots, y_m)$ над B_2. Подставим в f_i и в Ψ вместо y_j всюду A_j (в f_i переменная y_j входит только один раз, а Ψ может содержать несколько вхождений y_j). В результате f_i и Ψ перейдут в некоторые формулы, которые опять же будут эквивалентными — с учётом определения функций, реализуемых формулами. Такую подстановку проделаем сразу для всех $j = 1, \ldots, m$. В итоге вместо $f_i(y_1, \ldots, y_m)$ получим формулу Φ над B_1, реализующую $f(x_1, \ldots, x_n)$, а вместо $\Psi(y_1, \ldots, y_m)$ — некоторую формулу Ψ' над $B_1 \cup B_2$, также реализующую $f(x_1, \ldots, x_n)$. В формуле Ψ' каждую подформулу A_j (отличную от переменной) над B_1 заменим эквивалентной

ей подформулой Ψ_j над B_2 — такую замену можно проделать по предположению индукции, поскольку сложность A_j, очевидно, по крайней мере на единицу меньше сложности формулы Φ. В итоге вместо Ψ' получим формулу Ψ'' над B_2, реализующую $f(x_1, \ldots, x_n)$ — согласно ранее приведённому утверждению о замене в формуле некоторой подформулы эквивалентной ей подформулой. Теорема доказана.

Приведём примеры полных систем.

1) Система P_2 — множество всех булевых функций — является, очевидно, полной.

2) Система $B_1 = \{\overline{x}, x_1 \& x_2, x_1 \vee x_2\}$ полна по теореме 10.

3) Система $B_2 = \{\overline{x}, x_1 \& x_2\}$ является полной. Для доказательства достаточно воспользоваться тождеством $x_1 \vee x_2 = \overline{\overline{x}_1 \& \overline{x}_2}$ и теоремой 11.

4) Система $B_3 = \{\overline{x}, x_1 \vee x_2\}$ является полной. Для доказательства этого утверждения достаточно воспользоваться тождеством $x_1 \& x_2 = \overline{\overline{x} \vee \overline{x}_2}$ и теоремой 11.

5) Система $B_4 = \{0, 1, x_1 \& x_2, x_1 \oplus x_2\}$ является полной. Для доказательства этого факта достаточно заметить, что $x \oplus 1 = \overline{x}$, и воспользоваться теоремой 11 (взяв в качестве первой системы $\{\overline{x}, x_1 \& x_2\}$, а в качестве второй систему B_4).

Формула, построенная из констант 0, 1 и функций $x_1 \& x_2$, $x_1 \oplus x_2$, после раскрытия скобок и несложных алгебраических преобразований переходит в выражение вида

$$\sum_{(i_1, \ldots, i_s)} a_{i_1 \ldots i_s} x_{i_1} \ldots x_{i_s},$$

которое является полиномом по модулю 2 и называется *полиномом Жегалкина* (знаком $\sum$ обозначается суммирование по модулю 2).

В формуле для полинома в качестве свободного члена может присутствовать 1 (пустая конъюнкция). Если все коэффициенты $a_{i_1, \ldots, i_s}$ равны 0, то получаем полином Жегалкина, реализующий константу 0; такой полином обозначают через 0.

Теорема 12 (Жегалкина). *Каждая булева функция может быть представлена, и притом единственным образом, своим полиномом Жегалкина.*

Доказательство. Возможность реализации любой булевой функции полиномом Жегалкина уже установлена выше — такая возможность следует из полноты системы $\{0, 1, x_1 \& x_2, x_1 \oplus x_2\}$. Докажем единственность.

Подсчитаем число полиномов Жегалкина от переменных $x_1, \ldots, x_n$, т. е. число выражений вида

$$\sum_{(i_1,\ldots,i_s)} a_{i_1\ldots i_s} x_{i_1} \ldots x_{i_s}.$$

Число слагаемых $x_{i_1} \ldots x_{i_s}$ равно количеству подмножеств $\{i_1, \ldots, i_s\}$ из n чисел $1, \ldots, n$, т. е. 2^n. Поскольку $a_{i_1\ldots i_s}$ равно 0 или 1, то искомое число полиномов равно 2^{2^n}, т. е. числу булевых функций от тех же n переменных. Отсюда получаем единственность представления функций полиномами Жегалкина. Теорема доказана.

§ 5. Функции k-значной логики

Функция $f(x_1, \ldots, x_n)$ называется *функцией k-значной логики*, если переменные этой функции принимают значения из множества $E_k = \{0, 1, \ldots, k-1\}$ и $f(\alpha_1, \ldots, \alpha_n) \in E_k$ при $\alpha_i \in E_k (i = 1, \ldots, n)$.

Функции k-значной логики можно задавать с помощью таблиц (аналогично табличному заданию булевых функций). Из табличного задания функций k-значной логики легко усмотреть, что число всех функций k-значной логики от n переменных равно k^{k^n}.

Аналогично тому, как это делалось для булевых функций, для функций k-значной логики вводятся понятия существенной и фиктивной переменных, равенства функций, формул над множеством функций, реализуемых формулами функций, эквивалентности формул.

В качестве элементарных функций в k-значной логике часто рассматриваются, например, такие функции:

1) $\overline{x} = x + 1 \pmod k$ — отрицание Поста;

2) $\mathcal{N}x = k - 1 - x$ — отрицание Лукасевича;

3) $I_j(x) = \begin{cases} k-1 & \text{при } x = j, \\ 0 & \text{при } x \neq j \end{cases} (j = 0, 1, \ldots, k-1)$ — характеристическая функция (второго рода) числа j;

4) $j_i(x) = \begin{cases} 1 & \text{при } x = i, \\ 0 & \text{при } x \neq i \end{cases} (i = 0, 1, \ldots, k-1)$ — характеристическая функция числа i;

5) $\min(x_1, x_2)$ — минимум x_1 и x_2;

6) $x_1 \cdot x_2 \pmod k$ — произведение по модулю k;

7) $\max(x_1, x_2)$ — максимум x_1 и x_2;

8) $x_1 + x_2 \pmod k$ — сумма по модулю k;

9) $V_k(x_1, x_2) = \max(x_1, x_2) + 1 \pmod k$ — функция Вебба.

Функция $\min(x_1, x_2)$ является обобщением конъюнкции и часто обозначается как $x_1 \& x_2$. Функция $\max(x_1, x_2)$ является обобщением дизъюнкции и часто обозначается как $x_1 \vee x_2$.

Наряду с очевидными элементами сходства, между булевыми функциями и функциями k-значной логики имеются и существенные различия между этими функциями. Например, отрицания Поста и Лукасевича можно рассматривать как некоторые обобщения инверсии булевой функции; вместе с тем при $k \geqslant 3$ для отрицания Поста в k-значной логике $\overline{\overline{x}} \neq x$. Или, скажем, для булевых функций простейшие эквивалентности можно было доказывать тривиальным образом — перебирая всевозможные наборы значений переменных и убеждаясь, что во всех случаях значения функций, отвечающих эквивалентным формулам, в левой и в правой частях соответствующих равенств при одних и тех же значениях переменных совпадают. В k-значной логике перебор всевозможных наборов значений переменных исключается, поскольку k может быть, вообще говоря, каким угодно натуральным числом (бóльшим двух).

Для функций k-значной логики возможны представления, которые можно рассматривать как некоторые аналоги совершенной д.н.ф., например:

$$f(x_1, \ldots, x_n) = \bigvee_{(\sigma_1, \ldots, \sigma_n)} I_{\sigma_1}(x_1) \& \ldots \& I_{\sigma_n}(x_n) \& f(\sigma_1, \ldots, \sigma_n),$$

где $x \& y = \min(x, y)$, $x \vee y = \max(x, y)$. Доказывается это тождество (как и разложение булевой функции по всем переменным) непосредственной проверкой того факта, что при любом наборе $(\alpha_1, \ldots, \alpha_n)$ значений переменных в левой и в правой частях тождества получаем $f(\alpha_1, \ldots, \alpha_n)$. Из этого представления следует, в частности, что система функций k-значной логики $\{0, 1, \ldots, k-1, I_0(x), I_1(x), \ldots, I_{k-1}(x), \min(x_1, x_2), \max(x_1, x_2)\}$ полна в P_k (через P_k обозначаем множество всех функций k-значной логики при заданном k).

Глава IV

ПРЕДИКАТЫ

§ 1. Высказывания, предикаты, кванторы. Геометрический смысл кванторов

Пусть M — произвольное непустое множество (не обязательно конечное), а $x_1, \ldots, x_n$ (или $x, y, z, \ldots$) — произвольные элементы из этого множества. Будем рассматривать предложения, представляющие собой некоторые высказывания относительно $x_1, \ldots, x_n$, которые становятся определёнными — истинными или ложными, — когда $x_1, \ldots, x_n$ заменяются определёнными элементами из предметной области M.

Заметим, что «высказывание» в данном случае — исходное понятие, оно не определяется формально и может только быть пояснено: высказывание — это предложение, относительно которого можно лишь утверждать, истинно оно или ложно.

Например, в качестве M можно взять множество натуральных чисел и сформулировать такие высказывания:

1) «x есть простое число»;

2) «$x < y$»;

3) «$x + y = z$».

Очевидно, первое высказывание истинно, например, при $x = 3$ и ложно при $x = 6$; второе высказывание истинно при $x = 3$, $y = 6$ и ложно при $x = 6$, $y = 3$ и т. п.

Если приведённые высказывания обозначим соответственно через $P(x)$, $Q(x, y)$, $R(x, y, z)$, а истину и ложь условимся обозначать соответственно единицей и нулём, то получаем: $P(3) = 1$; $P(6) = 0$; $Q(3, 6) = 1$; $Q(6, 3) = 0$; $R(2, 2, 4) = 1$; $R(3, 2, 1) = 0$ и т. п.

Функцию $P(x_1, \ldots, x_n)$ будем называть *предикатом*, если переменные $x_1, \ldots, x_n$ принимают значения из некоторого непустого множества M, а сама функция — значения 1 («истина») или 0 («ложь»).

Над предикатами можно выполнять обычные логические операции, скажем, $\&, \vee, \bar{}, \to, \sim$; в результате также будут получаться предикаты.

Примеры. 1) Пусть предикат $D_2(x)$ принимает значение 1 тогда и только тогда, когда x делится на 2, т. е. $D_2(x)$ — предикат делимости на 2. Пусть $D_3(x)$ — аналогичный предикат делимости на 3. Тогда $D_2(x)\&D_3(x)$ будет предикатом делимости на 6.

2) Пусть предикат $Q(x, y)$ означает, как говорилось выше, высказывание «$x < y$», т. е. $Q(x, y)$ обращается в единицу в том и только том случае, когда $x < y$. Тогда предикат $Q(x, y)\&Q(y, z) \to Q(x, z)$ принимает значение 1 при любых значениях x, y, z, а предикат $(Q(x, y) \vee \overline{Q}(x, y)) \to (Q(x, y)\&Q(y, x))$ принимает значение 0 при любых значениях x, y.

Формальное определение операций над предикатами будет дано ниже; здесь же затрагивается, скорее, содержательная сторона рассматриваемых математических моделей.

Помимо логических операций специально для предикатов вводятся две новые операции (навешивание кванторов).

Квантор общности. Пусть $P(x)$ — некоторый предикат. Высказывание «$P(x)$ истинно для всех x» будем изображать выражением $(\forall x)\, P(x)$, которое читается «для всех x $P(x)$». Это высказывание истинно тогда и только тогда, когда $P(x)$ истинно для всех значений x. Переход от предиката $P(x)$ к $(\forall x)\, P(x)$ называется *навешиванием квантора общности*.

Квантор существования. Высказывание «$P(x)$ истинно для некоторых x» будем изображать выражением $(\exists x)\, P(x)$, которое читается «существует x, для которого $P(x)$». Это высказывание истинно тогда и только тогда, когда $P(x)$ истинно хотя бы при одном значении переменной x. Переход от предиката $P(x)$ к предикату $(\exists x)\, P(x)$ называется *навешиванием квантора существования*.

Операции навешивания кванторов можно применять и к предикатам от большего числа переменных. В результате применения одной такой операции у получающихся новых предикатов число переменных уменьшается на единицу (подробнее и формально об этом речь пойдёт ниже).

Примеры. Для предикатов из примеров 1) и 2) имеем: $(\forall x)\, D_2(x)$ — ложное высказывание, а $(\exists x)\, (D_2(x)\&D_3(x))$ — истинное высказывание; $(\forall x)(\forall y)(\forall z)(Q(x, y)\&Q(y, z) \to \to Q(x, z))$ — истинное высказывание.

Каждой числовой функции $f(x_1, \ldots, x_n)$ можно сопоставить предикат $P(x_1, \ldots, x_n, z)$, истинный тогда и только тогда, когда $f(x_1, \ldots, x_n) = z$. Этот предикат называется *представля-*

ющим для функции $f(x_1, \ldots, x_n)$. Пусть предикат $A(x, y, z)$ представляет функцию $x + y = z$. Тогда на языке предикатов утверждение о коммутативности сложения можно выразить так: $(\forall x)(\forall y)(\forall z)(A(x, y, z) \to A(y, x, z))$.

Существует простая связь между логическими и теоретико-множественными операциями. С каждым предикатом $P(x_1, \ldots, x_n)$ можно связать множество M_P всех наборов значений переменных $(a_1, \ldots, x_n)$, на которых этот предикат обращается в единицу. Пусть предикатам $P(x_1, \ldots, x_n)$ и $Q(x_1, \ldots, x_n)$ отвечают соответственно множества M_P и M_Q. Тогда предикату $P(x_1, \ldots, x_n)\&Q(x_1, \ldots, x_n)$, очевидно, будет соответствовать пересечение множеств M_P и M_Q, а предикату $P(x_1, \ldots, x_n) \vee Q(x_1, \ldots, x_n)$ — объединение этих множеств M_P и M_Q. Предикату $\overline{P}$ соответствует дополнение множества M_P. Т. о., $M_{P\&Q} = M_P \cap M_Q$, $M_{P \vee Q} = M_P \cup M_Q$, $M_{\overline{P}} = CM_P$, где CM_P — дополнение M_P.

Операции $(\exists x_1) P(x_1, \ldots, x_n)$ соответствует проецирование множества M_P вдоль оси x_1. Действительно, если в M_P входит хотя бы один набор $(a_1, a_2, \ldots . a_n)$, то в $M_{(\exists x_1) P}$ входит набор $(a_2, \ldots . a_n)$.

Поскольку $\overline{(\forall x_1) P(x_1, \ldots, x_n)} = (\exists x_1) \overline{P(x_1, \ldots, x_n)}$ (это свойство будет доказано формально), то операции $(\forall x_1) P(x_1, \ldots, x_n)$ соответствует взятие дополнения к проекции вдоль оси x_1 дополнения множества M_P.

§ 2. Модель, сигнатура модели, формулы в модели. Свободные и связанные переменные

Предикат $P(x_1, \ldots, x_n)$, переменные которого принимают значения из множества M, будем называть *предикатом, определённым на множестве* M.

Множество M с определёнными на нем предикатами $P_1(x_1, \ldots, x_{n_1}), \ldots, P_m(x_1, \ldots, x_{n_m})$ будем называть *моделью*. Число предикатов в модели предполагается конечным, а на мощность множества M не накладывается никаких ограничений.

При необходимости число переменных у предикатов будем указывать с помощью соответствующих верхних индексов, например, $P_1^{(n_1)}(x_1, \ldots, x_{n_1}), \ldots, P_m^{(n_m)}(x_1, \ldots, x_{n_m})$.

Совокупность знаков предикатов, входящих в модель, будем называть *сигнатурой* модели (предполагается, что при указании сигнатуры модели указывается число переменных каждого предиката).

Пусть $\{x_1, x_2, \ldots, y, z, \ldots\}$ — счётное множество переменных, D — модель, $\Sigma = \{P_1^{(n_1)}, \ldots, P_m^{(n_m)}\}$ — её сигнатура. Определим по индукции (одновременно) понятия *формулы* (в модели D), множества *свободных* переменных формулы, множества *связанных* переменных формулы, *значения* формулы в модели D на произвольном наборе значений свободных переменных формулы.

1. Пусть $P_i^{(n_i)}(x_{r_1}, x_{r_2}, \ldots, x_{r_{n_i}})$ — формула (переменные x_{r_1}, $x_{r_2}, \ldots, x_{r_{n_i}}$ могут и не быть попарно различными). Такие формулы будем называть *атомными* (ещё их называют *элементарными*).

Пусть $x_{p_1}, \ldots, x_{p_s}$ — все попарно различные переменные, входящие в данную формулу и выписанные в некотором порядке (этот порядок зафиксируем). Тогда множество $\{x_{p_1}, \ldots, x_{p_s}\}$ является множеством свободных переменных этой формулы, а множество связанных переменных этой формулы пусто.

Пусть $\widetilde{a} = (a_{p_1}, \ldots, a_{p_s})$ — произвольный набор длины s элементов из M. Значением формулы $P_i^{(n_i)}(x_{r_1}, \ldots, x_{r_{n_i}})$ на наборе $\widetilde{a}$ будем считать то значение, которое принимает предикат $P_i^{(n_i)}(x_{r_1}, x_{r_2}, \ldots, x_{r_{n_i}})$ при соответствующем замещении его переменных элементами из набора $\widetilde{a}$ (при этом переменная x_{r_1} замещается элементом a_{r_1}, переменная x_{r_2} — элементом a_{r_2} и т. д., наконец, переменная $x_{r_{n_i}}$ замещается элементом $a_{r_{n_i}}$; все элементы a_{r_1}, a_{r_2},..., $a_{r_{n_i}}$ берутся из набора $\widetilde{a} = (a_{p_1}, \ldots, a_{p_s})$).

2. Пусть A — формула, X' — множество её свободных переменных, X'' — множество её связанных переменных. Тогда $\overline{A}$ — формула, множество её свободных переменных есть X', множество её связанных переменных есть X''. Значение формулы $\overline{A}$ на любом наборе $\widetilde{a}$ значений её свободных переменных противоположно значению формулы A на том же наборе $\widetilde{a}$.

3. Пусть A_1 и A_2 — формулы, X_1' и X_2' — множества их свободных переменных, X_1'' и X_2'' — множества их связанных переменных, причем никакая свободная переменная из A_1 не является связанной в A_2 и никакая свободная переменная из A_2 не является связанной в A_1 (т. е. $(X_1' \cap X_2'') \cup (X_2' \cap X_1'')$ пусто). Тогда $(A_1)\&(A_2)$, $(A_1) \vee (A_2)$, $(A_1) \to (A_2)$, $(A_1) \sim (A_2)$ — формулы. Множество X' свободных переменных в каждой из них есть $X_1' \cup X_2'$; множество X'' связанных переменных в каждой из них есть $X_1'' \cup X_2''$.

Пусть $\widetilde{a}$ — произвольный набор значений переменных из X'. Этот набор определяет набор $\widetilde{a}_1$ значений переменных из X_1' и набор $\widetilde{a}_2$ значений переменных из X_2'. Пусть α_1 есть значение

формулы A_1 на наборе $\widetilde{a}_1$, а α_2 — значение формулы A_2 на наборе $\widetilde{a}_2$. Тогда значения указанных (новых) формул на наборе $\widetilde{a}$ равны соответственно

$$\alpha_1 \& \alpha_2,\ \alpha_1 \vee \alpha_2,\ \alpha_1 \to \alpha_2,\ \alpha_1 \sim \alpha_2.$$

Если формула A имеет свободные переменные $x_1, \ldots, x_s$, то её значение на наборе $\widetilde{a} = (a_1, \ldots, a_s)$ будем обозначать символом $A|_{\widetilde{a}}$ или $A|_{(a_1,\ldots,a_s)}$.

4. Пусть A — формула, $\{x, x_1, \ldots, x_s\}$ — множество её свободных переменных, $\{y_1, \ldots, y_t\}$ — множество её связанных переменных. Тогда $(\forall x)(A)$ есть формула; множество её свободных переменных есть $\{x_1, \ldots, x_s\}$, множество её связанных переменных есть $\{x, y_1, \ldots, y_t\}$. Значение $(\forall x)(A)|_{(a_1,\ldots,a_s)}$ формулы $(\forall x)(A)$ на наборе $(a_1, \ldots, a_s)$ значений свободных переменных определим следующим образом.

Рассмотрим всевозможные наборы $(a, a_1, \ldots, a_s)$ при фиксированных $a_1, \ldots, a_s$ (a — любой элемент из M). Если для каждого набора $(a, a_1, \ldots, a_s)$ имеем $A|_{(a,a_1,\ldots,a_s)} = 1$, то полагаем $(\forall x)(A)|_{(a_1,\ldots,a_s)} = 1$. Если же хотя бы для одного набора $(a, a_1, \ldots, a_s)$ имеет место равенство $A|_{(a,a_1,\ldots,a_s)} = 0$, то полагаем $(\forall x)(A)|_{(a_1,\ldots,a_s)} = 0$.

В формуле $(\forall x)(A)$ формулу A объявляем *областью действия* квантора $\forall x$.

5. Пусть A — формула, $\{x, x_1, \ldots, x_s\}$ — множество её свободных переменных, $\{y_1, \ldots, y_t\}$ — множество её связанных переменных. Тогда $(\exists x)(A)$ есть формула; множество её свободных переменных есть $\{x_1, \ldots, x_s\}$, множество её связанных переменных есть $\{x, y_1, \ldots, y_t\}$. Значение формулы $(\exists x)(A)$ на наборе $(a_1, \ldots, a_s)$ определим следующим образом.

Рассмотрим всевозможные наборы $(a, a_1, \ldots, a_s)$ при фиксированных $a_1, \ldots, a_s$ (a — любой элемент из M). Если для каждого набора $(a, a_1, \ldots, a_s)$ имеем $A|_{(a,a_1,\ldots,a_s)} = 0$, то полагаем $(\exists x)(A)|_{(a_1,\ldots,a_s)} = 0$. Если же хотя бы для одного набора $(a, a_1, \ldots, a_s)$ имеет место равенство $A|_{(a,a_1,\ldots,a_s)} = 1$, то полагаем $(\exists x)(A)|_{(a_1,\ldots,a_s)} = 1$.

В формуле $(\exists x)(A)$ формулу A объявляем областью действия квантора $\exists x$.

Из определения формулы следует, что никакая переменная не может быть одновременно свободной и связанной.

При фиксированной модели D каждая формула, имеющая свободные переменные, выражает некоторый вполне определён-

ный предикат от этих переменных; формула, не имеющая свободных переменных, выражает определённую константу (0 или 1).

Примеры. Пусть в модели D имеются предикаты $P(x_1, x_2)$ и $Q(x_1, x_2)$.

1) $(\forall x)(P(x, x))$ — формула; множество её свободных переменных пусто, множество связанных переменных содержит одну переменную x.

2) $(P(x, x) \sim Q(y, y))$ — формула, в которой обе переменные x, y — свободные, а множество связанных переменных пусто.

3) $(P(x, y)) \to ((\exists x)(Q(x, x)))$ — выражение, не являющееся формулой; переменная x является свободной в левой части импликации и связанной в её правой части.

4) $(P(x, y)) \to ((\forall z)(Q(z, x)))$ — формула.

Наряду с соглашениями, принятыми в алгебре логики, введем ещё некоторые соглашения, позволяющие упрощать записи формул:

а) не будем заключать в скобки атомные формулы;

б) не будем заключать в скобки формулы, в которых отрицание является внешней операцией;

в) не будем заключать в скобки формулу, в которой внешняя операция есть навешивание квантора, если следующая операция — также навешивание квантора;

г) будем считать, что квантор связывает сильнее всех других операций, и опускать соответствующие скобки.

При таких соглашениях формулы из приведённых выше примеров 1), 2), 4) приобретут вид: $(\forall x)P(x, x)$; $P(x, y) \sim Q(y, y)$; $P(x, y) \to (\forall z)(Q(z, x))$. Вместо $(\forall x)\left((\exists y)\overline{((P(x, y)) \to (Q(y, y)))}\right)$ будем писать $(\forall x)(\exists y)\overline{P(x, y) \to Q(y, y)}$.

§ 3. Истинность формулы в модели, на множестве. Тождественно истинные формулы

Формулу F будем называть *истинной в модели* D, если она принимает значение 1 на всех наборах значений своих свободных переменных.

Сигнатурой формулы будем называть множество символов, обозначающих входящие в неё предикаты (с указанием числа переменных). Например, формула

$$P_1(x_1, x_2) \vee (\exists y)(P_1(y, x_1) \& (\forall z)P_2(z, x_1))$$

имеет сигнатуру $\{P_1(x_1, x_2), P_2(x_1, x_2)\}$ (или $\{P_1^{(2)}, P_2^{(2)}\}$).

Пусть формула F имеет сигнатуру Σ. Будем называть эту формулу *истинной на множестве* M, если она истинна во всех моделях, определённых на множестве M и имеющих сигнатуру Σ.

Легко заметить, что если формула истинна на некотором множестве, то она истинна на любом множестве той же мощности и потому можно говорить об истинности формул на множествах определённой мощности.

Формула называется *тождественно истинной* (или *общезначимой*), если она истинна на всех множествах (или, что то же самое, если она истинна во всех моделях).

Примеры. 1) Формула $P(x, x)$ истинна в модели D_1, но не является истинной в модели D_2 (соответственно табл. 6 и 7).

Таблица 6

x \ y	a	b
a	1	0
b	0	1

Таблица 7

x \ y	a	b
a	1	1
b	0	0

2) Формула $(\exists x)P(x) \to (\forall x)P(x)$ истинна на одноэлементном множестве, но не является истинной на множестве из большего числа элементов.

3) Формула $(\exists x)(\forall y)P(x, y) \to (\forall y)(\exists x)P(x, y)$ является тождественно истинной.

В некоторых случаях для решения задачи установления истинности формул логики предикатов можно указать эффективные алгоритмы (например, при установлении истинности формул на конечных множествах). Однако в общем случае такая задача (например, в случае установления тождественной истинности) оказывается существенно более сложной, чем для формул алгебры логики.

§ 4. Эквивалентность формул. Правила преобразования формул с кванторами

Пусть формулы F и G имеют одно и то же множество свободных переменных (в частности, пустое).

Будем считать эти формулы F и G *эквивалентными* (или *равносильными*) *в модели* D, если на любом наборе значений свободных переменных они принимают равные значения (т. е. если они выражают один и тот же предикат).

Будем считать эти формулы *эквивалентными* (*равносильными*) *на множестве* M, если они эквивалентны во всех моделях, заданных на множестве M и имеющих сигнатуру, включающую символы — обозначения предикатов этих формул. Очевидно, что данное понятие зависит только от мощности множества M (от числа элементов, если множество конечно).

Будем считать две формулы *эквивалентными* (*равносильными*), если они эквивалентны на всех множествах.

Примеры. 1) Возьмём множество $M = \{a, b\}$ и определённый на нем предикат $P(x, y)$, заданный табл. 6. Полученную модель обозначим через D_1.

Рассмотрим две формулы: $F_1 = P(x_1, x_2)\&P(x_1, x_3)$ и $F_2 = P(x_1, x_2)\&P(x_2, x_3)$. Легко заметить, что каждая из этих формул обращается в единицу только на двух наборах: (a, a, a) и (b, b, b). Значит, они эквивалентны в модели D_1.

Возьмем теперь модель D_2, отличающуюся от модели D_1 тем, что предикат $P(x_1, x_2)$ в D_2 задается теперь табл. 7.

Легко видеть, что на наборе (a, b, b) формула F_1 принимает значение 1, а формула F_2 — значение 0. Значит, эти формулы не эквивалентны в D_2.

2) Рассмотрим формулы $(\exists x)P(x)$ и $(\forall x)P(x)$. Нетрудно заметить, что они эквивалентны на множестве из одного элемента. Действительно, на множестве из одного элемента каждый предикат принимает только одно значение и потому имеется ровно два предиката: равный 0 и равный 1. Эти предикаты определяют две модели сигнатуры $\{P(x)\}$. В первой модели обе формулы принимают значение 0 (и потому эквивалентны), а во второй модели обе формулы принимают значение 1 (и тоже эквивалентны).

С другой стороны, на множестве $\{a, b\}$ из двух элементов существует модель, в которой рассматриваемые формулы не эквивалентны; можно, например, предикат P определить так:

$$P(a) = 0,\ P(b) = 1.$$

3) Очевидно, что формулы $P(x) \to P(y)$ и $\overline{P(x)} \vee P(y)$ эквивалентны (в любой модели).

Рассмотрим некоторые правила перехода от одних формул к другим, эквивалентным им (во всех моделях). Эквивалентность формул будем изображать знаком равенства.

Очевидно, что для предикатных формул сохраняются все правила эквивалентных преобразований формул из алгебры логики. Справедлив также соответствующий аналог правила замены (утверждение в гл. III, § 2) на эквивалентную подформулу.

Имеются и специфические правила, относящиеся к формулам с кванторами.

Правило переноса квантора через отрицание. Пусть $F(x)$ — формула, имеющая свободную переменную x (и, быть может, ещё свободные переменные $x_1, \ldots, x_s$). Тогда справедливы соотношения:

$$\overline{(\forall x)F(x)} = (\exists x)\overline{F(x)}, \tag{1}$$

$$\overline{(\exists x)F(x)} = (\forall x)\overline{F(x)}. \tag{2}$$

Сначала докажем первое из этих соотношений. Пусть D — произвольная модель, сигнатура которой содержит все предикаты из $F(x)$, а M — множество элементов в этой модели.

Пусть $(a_1, \ldots, a_s)$ — произвольный набор значений переменных $x_1, \ldots, x_s$ (если x — единственная свободная переменная формулы $F(x)$, то такой набор не берётся). Рассмотрим всевозможные наборы $(a, a_1, \ldots, a_s)$, где $a_1, \ldots, a_s$ фиксированы, a — произвольный элемент из M. Если для любого набора $(a, a_1, \ldots, a_s)$ имеем $F(x)|_{(a,a_1,\ldots,a_s)} = 1$, то для этого набора получим $\overline{F(x)}|_{(a,a_1,\ldots,a_s)} = 0$. Поэтому, с одной стороны, $\overline{(\forall x)F(x)}|_{(a_1,\ldots,a_s)} = 0$, а, с другой стороны, $(\exists x)\overline{F(x)}|_{(a_1,\ldots,a_s)} = 0$.

Если существует набор $(a, a_1, \ldots, a_s)$, такой, что $F(x)|_{(a,a_1,\ldots,a_s)} = 0$, то для этого набора $\overline{F(x)}|_{(a,a_1,\ldots,a_s)} = 1$. Поэтому, с одной стороны, $(\forall x)F(x)|_{(a_1,\ldots,a_s)} = 0$, $\overline{(\forall x)F(x)}|_{(a_1,\ldots,a_s)} = 1$, а, с другой стороны, $(\exists x)\overline{F(x)}|_{(a_1,\ldots,a_s)} = 1$. Таким образом, в обоих случаях значения формул равны.

Соотношение (2) получим из (1) с использованием правил алгебры логики. Применим (1) к формуле $\overline{F(x)}$. Тогда, используя закон двойного отрицания, получим

$$\overline{(\forall x)\overline{F(x)}} = (\exists x)\overline{\overline{F(x)}} = (\exists x)F(x).$$

Отсюда (навешиванием отрицания) получаем

$$\overline{\overline{(\forall x)\overline{F(x)}}} = \overline{(\exists x)F(x)};$$

используя ещё один раз закон двойного отрицания, получаем требуемое соотношение: $(\forall x)\overline{F(x)} = \overline{(\exists x)F(x)}$.

Правило выноса квантора за скобки. Пусть формула $F(x)$ содержит свободную переменную x, а формула G не содержит переменную x (F и G могут содержать ещё какие-то переменные). Пусть, далее, при соединении формул $F(x)$ и G знаком

логической операции получается формула (в соответствии с п. 3 определения формул). Тогда справедливы соотношения:

$$(\forall x)(F(x)\&G) = (\forall x)F(x)\&G,$$
$$(\forall x)(F(x) \vee G) = (\forall x)F(x) \vee G,$$
$$(\exists x)(F(x)\&G) = (\exists x)F(x)\&G,$$
$$(\exists x)(F(x) \vee G) = (\exists x)F(x) \vee G.$$

Докажем первое соотношение (остальные доказываются аналогично). Рассмотрим произвольную модель D, сигнатура которой содержит все предикаты из $F(x)$ и G, а M — множество элементов в этой модели. Пусть $x_1, \ldots, x_s$ — все свободные переменные формулы $(\forall x)(F(x)\&G)$ (они же — все свободные переменные также формулы $(\forall x)F(x)\&G$). Пусть $(a_1, \ldots, a_s)$ — произвольный набор значений этих переменных. Поскольку G не содержит переменную x, то можно определить значение формулы G на наборе $(a_1, \ldots, a_s)$ (или, точнее, на части этого набора, определяющей значения свободных переменных формулы G).

Если $G|_{(a_1,\ldots,a_s)} = 0$, то $((\forall x)F(x)\&G)|_{(a_1,\ldots,a_s)} = 0$. С другой стороны, тогда для любого набора $(a, a_1, \ldots, a_s)$ (здесь $a_1, \ldots, a_s$ фиксированы, а a — произвольный элемент множества M) $F(x)\&G|_{(a,a_1,\ldots,a_s)} = 0$, поэтому $(\forall x)(F(x)\&G)|_{(a_1,\ldots,a_s)} = 0$.

Если же $G|_{(a_1,\ldots,a_s)} = 1$, то

$$(\forall x)(F(x)\&G)|_{(a_1,\ldots,a_s)} = \\ = (\forall x)F(x)|_{(a_1,\ldots,a_s)} = ((\forall x)F(x)\&G)|_{(a_1,\ldots,a_s)},$$

т. е. требуемое соотношение выполняется.

Правило переименования связанных переменных. При замене любой связанной переменной на другую переменную (в кванторе и всюду в области действия квантора) так, чтобы не нарушалось определение формулы, получается формула, эквивалентная исходной.

Это правило почти очевидно (формально может быть доказано индукцией по длине формулы с использованием правила замены на эквивалентную подформулу).

§ 5. Приведённые формулы

Формулу будем называть *приведённой*, если в ней из логических операций встречаются лишь конъюнкция, дизъюнкция и отрицание, причём знак отрицания встречается только над атомными формулами.

Примеры. $P(x,y) \vee Q(x,y) \vee R(y,z)$, $(\forall x)P(x,y) \vee (\exists x)Q(x,y) \vee R(y,z)$, $(\forall x)\overline{P(x,y)} \vee (\exists x)Q(x,y) \vee \overline{R(y,z)}$ — приведённые формулы, а $\overline{P(x,y) \vee Q(x,y)} \vee R(y,z)$, $\overline{(\forall x)P(x,y)} \vee (\exists x)Q(x,y) \vee R(y,z)$, $(\forall x)(P(x,y) \to Q(x,y)) \vee \overline{R(y,z)}$ — формулы, не являющиеся приведёнными.

Теорема 13 (о приведённых формулах). *Для любой формулы F существует эквивалентная ей приведённая формула F', причем множества связанных переменных формул F и F' совпадают.*

(Множества свободных переменных формул F и F' также совпадают — это следует из их эквивалентности.)

Доказательство. *Длиной* формулы здесь и далее будем считать общее число входящих в неё знаков предикатов, кванторов и логических операций. Доказательство проведём индукцией по длине формул.

Формулы длины 1 — это атомные формулы; очевидно, что они являются приведёнными.

Допустим, что утверждение теоремы справедливо для всех формул, длина которых меньше l. Докажем его для формул длины l.

Пусть F — произвольная формула длины l. Она может представлять собой выражение одного из следующих видов:

1) $F_1 \& F_2$;
2) $F_1 \vee F_2$;
3) $F_1 \to F_2$;
4) $F_1 \sim F_2$;
5) $(\forall x)\, G(x)$;
6) $(\exists x)\, G(x)$;
7) $\overline{G}$

(формула $G(x)$ имеет свободную переменную x и, быть может, ещё и другие свободные переменные).

Рассмотрим отдельно каждый из представленных случаев.

1) Каждая из формул F_1 и F_2 имеет длину меньше l. По индуктивному предположению для них существуют эквивалентные приведённые формулы F_1' и F_2' соответственно. Множества свободных переменных формул F_1 и F_1' (F_2 и F_2') совпадают; множества связанных переменных формул F_1 и F_1' (F_2 и F_2') совпадают. Поскольку $F_1 \& F_2$ — формула, то и $F_1' \& F_2'$ — формула, которая эквивалентна формуле F и является приведённой. Требование согласованности множеств переменных также выполнено.

2) Этот случай рассматривается аналогично предыдущему.

3) В этом случае исходная формула F эквивалентна формуле $\overline{F}_1 \vee F_2$. Если длины формул F_1 и F_2 равны l_1 и l_2 соответственно, то длина l формулы F удовлетворяет условию $l = l_1 + l_2 + 1$. Поскольку $l_2 \geqslant 1$, то $l \geqslant l_1 + 2$. Длина формулы $\overline{F}_1$ равна $l_1 + 1$, т. е. меньше, чем l. По индуктивному предположению для формул $\overline{F}_1$ и F_2 существуют эквивалентные приведённые формулы F_1^* и F_2' соответственно. В качестве приведённой формулы, эквивалентной формуле F, возьмём $F_1^* \vee F_2'$.

4) В этом случае формула F эквивалентна формуле $(F_1 \& F_2) \vee (\overline{F}_1 \& \overline{F}_2)$. Каждая из формул F_1, F_2, $\overline{F}_1$, $\overline{F}_2$ имеет длину меньше l (это доказывается, как и в предыдущем пункте). Для этих формул существуют — по индуктивному предположению — эквивалентные приведённые формулы F_1', F_2', F_1^*, F_2^*. В качестве искомой приведённой формулы, эквивалентной формуле F, возьмём $(F_1' \& F_2') \vee (F_1^* \& F_2^*)$.

5) Формула $G(x)$ имеет длину меньше l. По индуктивному предположению для неё существует эквивалентная приведённая формула $F'(x)$. В качестве приведённой формулы, эквивалентной формуле F, возьмём $(\forall x)\, F'(x)$.

6) Этот случай рассматривается аналогично предыдущему.

7) Этот случай распадается на несколько подслучаев, в зависимости от вида формулы G; эта формула с внешней операцией «$\overline{\ }$» может быть одного из следующих видов:

7.1) $\overline{G_1 \& G_2}$;

7.2) $\overline{G_1 \vee G_2}$;

7.3) $\overline{G_1 \to G_2}$;

7.4) $\overline{G_1 \sim G_2}$;

7.5) $\overline{(\forall x) G(x)}$;

7.6) $\overline{(\exists x) G(x)}$;

7.7) $\overline{\overline{G}}$;

7.8) $\overline{G}$, где G — атомная формула; в этом случае $\overline{G}$ — приведённая формула.

Для подслучаев 7.1)–7.7) формула G эквивалентна соответственно формулам $\overline{G}_1 \vee \overline{G}_2$; $\overline{G}_1 \& \overline{G}_2$; $G_1 \& \overline{G}_2$; $(\overline{G}_1 \vee \overline{G}_2) \& (G_1 \vee G_2)$; $(\exists x)\overline{G(x)}$; $(\forall x)\overline{G(x)}$; G. Формулы G_1, $\overline{G}_1$, G_2, $\overline{G}_2$, $\overline{G(x)}$, G имеют длину меньше l. Для подслучаев 7.1)–7.6) дальнейшие рассуждения аналогичны рассуждениям при рассмотрении основных случаев 1)–6). Для подслучая 7.7) ситуация ещё

проще: можно взять приведённую формулу, эквивалентную формуле G (это можно сделать по индуктивному предположению). Теорема доказана.

§ 6. Нормальные формулы

Приведённая формула называется *нормальной*, если она или не содержит кванторов, или в ней операции взятия кванторов следуют за всеми другими операциями. Нормальная формула имеет вид

$$(Q_1x_1)(Q_2x_2)\dots(Q_kx_k)F(x_1, x_2, \dots, x_k, x_{k+1}, \dots),$$

где (Q_ix_i) — квантор, а формула $F(x_1, x_2, \dots, x_k, x_{k+1}, \dots)$ — приведённая формула, не содержащая кванторов.

Примеры. Нормальные формулы:

$$P(x, y)\&Q(x, y) \vee R(z, y),$$
$$(\forall x)(R(x, y) \vee Q(x, y) \vee R(z, y)),$$
$$(\forall y)(\forall x)(\overline{R(x, y)} \vee Q(x, y) \vee R(z, y));$$

приведённые формулы, не являющиеся нормальными:

$$(\forall x)R(x, y) \vee (\exists x)(Q(x, y) \vee P(z, z)),$$
$$(\forall x)(P(x, y) \vee Q(x, y) \vee (\forall z)R(z, y)).$$

Теорема 14 (о нормальных формулах). *Для любой формулы F существует эквивалентная ей нормальная формула.*

Доказательство. Индукцией по длине формул докажем следующее утверждение $(*)$: для любой приведённой формулы длины l существует эквивалентная ей нормальная формула длины l. Из этого утверждения $(*)$ и теоремы 13 следует утверждение доказываемой теоремы.

Для формул длины 1 утверждение $(*)$ очевидно: формулы длины 1 — это атомные формулы, являющиеся нормальными.

Предположим, что утверждение $(*)$ справедливо для формул длины меньше l. Докажем его для формул длины l. Приведённая формула F длины l может быть одного из следующих видов:

1) $\overline{F}_1$;
2) $F_1\&F_2$;
3) $F_1 \vee F_2$;
4) $(\forall x)\, F_1(x)$;
5) $(\exists x)\, F_1(x)$;

Рассмотрим отдельно каждый из этих случаев.

1) Этот случай возможен лишь тогда, когда F_1 — атомная формула. Но в таком случае $\overline{F}_1$ — нормальная формула.

2) Пусть формулы F_1 и F_2 имеют длины l_1 и l_2 соответственно. Тогда $l_1 + l_2 + 1 = l$. Длина каждой из формул F_1 и F_2 меньше l. По индуктивному предположению существуют эквивалентные им нормальные формулы F_1' и F_2' длины l_1 и l_2 соответственно. Если каждая из этих формул не содержит кванторов, то формула $F_1' \& F_2'$ является нормальной формулой, эквивалентной формуле F, и имеет длину l.

Рассмотрим теперь случай, когда какая-то из нормальных формул F_1' и F_2' (хотя бы одна) содержит квантор. Пусть, например, формула F_1' имеет вид $(\forall x)F_3'(x)$ (случай квантора существования рассматривается аналогично). Тогда формула F имеет вид $(\forall x)F_3'(x) \& F_2'$. Применяя правило переименования связанных переменных, перейдём к эквивалентной формуле $(\forall x)F_4'(x) \& F_5'$ (той же длины l), в которой стоящие на разных местах кванторы связывают разные переменные. Применяя правило выноса квантора за скобки, получим эквивалентную формулу $(\forall x)(F_4'(x) \& F_5')$. Стоящая в скобках формула $F_4'(x) \& F_5'$ имеет длину $l - 1$. Для неё в силу предположения существует эквивалентная нормальная формула $F_6'(x)$ длины $l - 1$. Очевидно, что формула $(\forall x)\, F_6'(x)$ является нормальной и имеет длину l.

3) Этот случай рассматривается аналогично предыдущему.

4) Формула $F_1(x)$ имеет длину $l - 1$. Для неё по индуктивному предположению существует эквивалентная нормальная формула $F_1'(x)$ длины $l - 1$. Очевидно, что формула $(\forall x)F_1'(x)$ является нормальной и имеет длину l.

5) Этот случай рассматривается аналогично предыдущему. Теорема доказана полностью.

Глава V

СХЕМЫ ИЗ ФУНКЦИОНАЛЬНЫХ ЭЛЕМЕНТОВ. СИНТЕЗ И ОЦЕНКИ СЛОЖНОСТИ СХЕМ

§ 1. Схемы из функциональных элементов в базисе $\{\&, \vee, ^- \}$

Схемой из функциональных элементов в базисе $\{\&, \vee, ^- \}$ называется помеченный ориентированный граф $G = (V, E)$ без контуров, удовлетворяющий следующим условиям.

1) Все вершины этого графа разбиваются на три непересекающихся подмножества V_1, V_2, V_3 так, что:

а) $V = V_1 \cup V_2 \cup V_3$, $V_i \cap V_j = \varnothing$ при $i \neq j$;

б) в каждую вершину из V_1 не входит ни одной дуги;

в) в каждую вершину из V_2 входит по одной дуге;

г) в каждую вершину из V_3 входят по две дуги.

2) Каждой вершине v из V_1 приписана некоторая переменная x_i, причём разным вершинам приписаны разные переменные.

3) Каждой вершине v' из V_2 приписан символ $^-$ (функция $\overline{x}$).

4) Каждой вершине v'' из V_3 приписан один из символов $\&$, $\vee$ (одна из функций $x_1 \& x_2$, $x_1 \vee x_2$).

5) Выделено некоторое подмножество V^* ($V^* \subseteq V$) вершин.

Вершины из V_1 (в которые не входят дуги) называются *входами* схемы; если вершине v из V_1 приписана переменная x_i, то говорят, что на вход v подаётся переменная x_i.

Вершины из V_2 и V_3 (в которые входят дуги) называются (*функциональными*) *элементами* схемы. Каждая вершина из V_2 (с приписанным ей символом $^-$) называется *инвертором*. Каждая вершина из V_3 с приписанным ей символом $\&$ называется *конъюнктором*, а каждая вершина из V_3 с приписанным ей символом $\vee$ называется *дизъюнктором*.

Вершины из V^* называются *выходами* схемы.

На рис. 10,*а* приведена некоторая схема из функциональных элементов, имеющая два входа, один выход и четыре функциональных элемента.

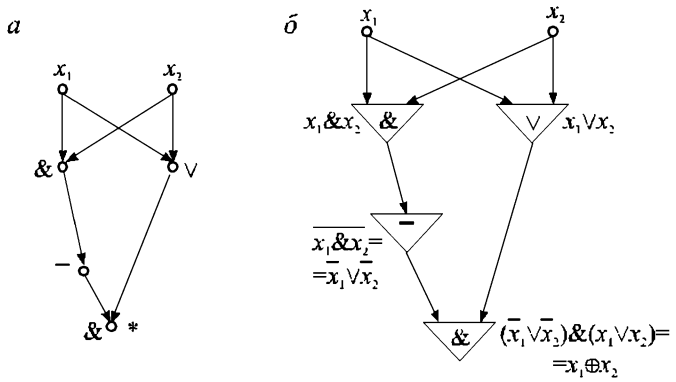


Рис. 10

Пусть $G = (V, E)$ — ориентированный граф, вершины которого занумерованы натуральными числами. Нумерацию вершин графа G будем считать *монотонной*, если для любой дуги $(v, v') \in E$ номер вершины v меньше номера вершины v'.

Лемма 4. *Любой (конечный) ориентированный граф без контуров допускает монотонную нумерацию вершин.*

Доказательство. Пусть $G = (V, E)$ — ориентированный граф без контуров. Возьмём в графе G произвольный путь Z максимальной длины (т. е. с максимальным числом рёбер) без повторяющихся рёбер; поскольку G — конечный граф и контуры в нём отсутствуют, то указанный путь Z найдётся. В вершину v, являющуюся началом пути Z, дуги не входят — иначе путь Z не являлся бы максимальным. Вершине v припишем номер 1 и удалим из G эту вершину вместе с выходящими из неё дугами.

Точно так же в оставшемся ориентированном графе без контуров G_1 найдём вершину, в которую не входят дуги, и припишем этой вершине номер 2, и т. д. Полученная нумерация вершин по построению будет монотонной для исходного графа G. Лемма доказана.

Рассмотрим теперь произвольную схему S из функциональных элементов с n входами, на которые подаются переменные $x_1, \ldots, x_n$. Занумеруем все вершины схемы S согласно лемме 4 так, чтобы полученная нумерация вершин была монотонной. Далее всем вершинам схемы S в порядке возрастания номеров этих вершин сопоставим булевы функции по следующим правилам.

Вершина v_1 имеет наименьший номер и является, очевидно, входом схемы, на который подаётся некоторая переменная x_i. Тождественную функцию x_i сопоставим вершине v_1.

Рассмотрим k-ю вершину v_k. Либо она является входом схемы, либо в v_k входит ровно одна дуга, либо, наконец, в v_k входят две дуги. В первом случае вершине v_k как входу схемы приписана некоторая переменная x_j; тогда вершине v_k сопоставим тождественную функцию x_j.

Во втором случае в вершину v_k входит ровно одна дуга, выходящая из некоторой вершины v. В силу монотонной нумерации вершин схемы вершине v уже сопоставлена некоторая булева функция f_v; отрицание этой функции, т. е. $\overline{f}_v$, сопоставим вершине v_k.

В третьем случае в вершину v_k входят две дуги, выходящие из некоторых вершин v', v'', которым уже сопоставлены (в силу монотонной нумерации вершин схемы) булевы функции $f_{v'}$ и $f_{v''}$ соответственно. В этом случае вершине v_k сопоставим булеву функцию $f_{v'} \& f_{v''}$, если v_k — конъюнктор, или $f_{v'} \vee f_{v''}$, если v_k — дизъюнктор.

Если вершине v схемы из функциональных элементов сопоставлена булева функция f, то будем говорить, что в вершине v *реализуется* булева функция f. Схема по определению *реализует* упорядоченную систему булевых функций, сопоставленных выходам данной схемы.

Лемма 5. *Булевы функции, реализуемые в вершинах схемы, полностью определяются самой схемой независимо от монотонной нумерации вершин схемы.*

Доказательство проведём индукцией по числу элементов в схемах.

Базис индукции. Для схем без элементов (такие схемы содержат только входы, и некоторые из этих входов являются одновременно и выходами схемы) утверждение леммы, очевидно, выполняется.

Индуктивный переход. Пусть утверждение леммы выполняется для всех схем, содержащих не более k элементов; докажем это утверждение для схемы S, содержащей $(k+1)$ элементов. Рассмотрим две монотонные нумерации вершин схемы S. Входам схемы, очевидно, сопоставляются одни и те же (тождественные) булевы функции как при первой нумерации, так и при второй.

Пусть v — элемент схемы S, получивший наименьший (среди номеров элементов) номер при первой нумерации, и этому элементу при первой нумерации сопоставлена булева функция f_v. Поскольку в вершину v могут входить только дуги, выходящие из входов схемы, то и при второй нумерации элементу v будет сопоставлена та же самая функция f_v.

Схеме S сопоставим схему S', получающуюся из S удалением всех дуг, входящих в v; вершину v в схеме S' объявим входом, на который подаётся $y = f_v$. Очевидно, что всякая монотонная для S нумерация вершин останется монотонной и для S'. Ясно, что при первой нумерации одноимённым вершинам в схемах S и S' будут сопоставлены одни и те же функции; точно так же при второй нумерации одноимённым вершинам в S и в S' будут сопоставлены одни и те же функции. Но по предположению индукции в схеме S' каждой вершине окажется сопоставленной одна и та же булева функция как при первой нумерации, так и при второй; то же самое справедливо и для схемы S. Лемма доказана.

Согласно лемме 5 функции, реализуемые схемой из функциональных элементов, определяются единственным образом. Часто функциональные элементы схем изображают в виде треугольников. Например, схему, приведённую на рис. 10,*а*, можно изобразить ещё и так, как показано на рис. 10,*б*; на последнем рисунке указаны также функции, реализуемые в вершинах схемы, или, как говорят, на выходах элементов схемы.

Пусть v и v' — две вершины схемы, а e — дуга, исходящая из v и входящая в v'; вершина v' будет некоторым элементом, а v — либо элементом, либо входом схемы. Вершинам v и v' сопоставлены некоторые функции φ и φ'; если v — вход схемы, то в этом случае φ есть некоторая переменная x_i, которая подаётся на вход v. Дуга e считается входом элемента v' и выходом элемента v (если v — функциональный элемент). Говорят, что φ реализуется на выходе элемента v и подаётся на вход элемента v', а вход элемента v' соединён с выходом элемента v (или со входом схемы v).

Две схемы называются *изоморфными*, если они получены из изоморфных графов так, что:

1) соответствующим входам приписаны одинаковые переменные;

2) соответствующим элементам приписаны одинаковые символы функций;

3) выходам одной схемы соответствуют выходы другой, и наоборот.

Ясно, что любые две изоморфные схемы реализуют одну и ту же (неупорядоченную) систему булевых функций.

§ 2. Синтез схем с использованием совершенных д.н.ф.

Сложность произвольной схемы S определим как число элементов в этой схеме и обозначим через $L(S)$. Положим $L(f) = \min L(S)$, где минимум берётся по всем схемам, реализующим булеву функцию f. Аналогично для системы функций F положим $L(F) = \min L(S)$, где минимум берётся по всем схемам, реализующим систему функций F. Число $L(f)$ называется *сложностью реализации функции* f или просто *сложностью функции* f. Аналогично, $L(F)$ — *сложность системы функций* (оператора) F.

Пусть $Q_n(x_1, \ldots, x_n)$ — система всех 2^n конъюнкций вида $x_1^{\sigma_1} \ldots x_n^{\sigma_n}$.

Теорема 15. $L(Q_n) \leqslant 2^{n+1} + n - 4$.

Доказательство проведём индукцией по n. При $n = 1$ утверждение очевидно – достаточно лишь с помощью одного инвертора реализовать $\overline{x}_1$. Предположим, что утверждение справедливо при $n = 1, \ldots, k$; докажем его для $n = k + 1$.

Опираясь на предположение индукции, построим схему S_1 из $(2^{k+1} + k - 4)$ элементов, реализующую все 2^k конъюнкций $x_1^{\sigma_1} \ldots x_k^{\sigma_k}$. Добавим к S_1 инвертор E^-, с помощью которого реализуем $\overline{x}_{k+1}$. Затем добавим ещё 2^{k+1} конъюнкторов, на выходах которых получим требуемые конъюнкции $x_1^{\sigma_1} \ldots x_k^{\sigma_k} x_{k+1}^{\sigma_{k+1}}$. При реализации конъюнкции $x_1^{\sigma_1} \ldots x_k^{\sigma_k} x_{k+1}^{\sigma_{k+1}}$ на выходе конъюнктора $E^{\&}$ один вход этого конъюнктора соединим с выходом S_1, на котором реализуется $x_1^{\sigma_1} \ldots x_k^{\sigma_k}$, а второй вход элемента $E^{\&}$ соединим со входом схемы, соответствующим x_{k+1}, если $\sigma_{k+1} = 1$, или с выходом инвертора E^-, если $\sigma_{k+1} = 0$; здесь и далее под входом элемента подразумеваем дугу, входящую в данный элемент, а под выходом элемента (при изображении схемы графом без использования треугольников) подразумевается сам этот элемент.

В итоге получим схему, содержащую $2^{k+2} + k + 1 - 4$ элементов. Теорема доказана.

Укажем теперь простой метод синтеза схем для реализации произвольной булевой функции.

Теорема 16. *Любую булеву функцию от* n *переменных можно реализовать схемой, содержащей не более* $(3 \cdot 2^n + n - 5)$ *элементов.*

Доказательство. Пусть дана произвольная булева функция $f(x_1, \ldots, x_n)$. Если $f(x_1, \ldots, x_n) \equiv 0$, то реализуем f схемой из двух элементов, приведённой на рис. 11.

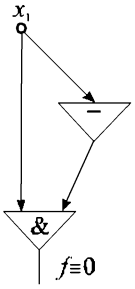


Рис. 11

Пусть $f(x_1, \ldots, x_n) \not\equiv 0$. Представим f в виде совершенной д.н.ф.:

$$f(x_1, \ldots, x_n) = \bigvee_{\substack{(\sigma_1,\ldots,\sigma_n):\\ f(\sigma_1,\ldots,\sigma_n)=1}} x_1^{\sigma_1} \ldots x_n^{\sigma_n}. \qquad (1)$$

Опираясь на утверждение теоремы 15, реализуем все элементарные конъюнкции вида $x_1^{\sigma_1} \ldots x_n^{\sigma_n}$ схемой S, содержащей не более чем $2^{n+1} + n - 4$ элементов. К S добавим цепочку Z из дизъюнкторов, с помощью которой реализуем логическую сумму из (1); входы цепочки Z соединим с теми выходами схемы S, на которых реализуются слагаемые из (1). В Z войдут, очевидно, не более чем $2^n - 1$ дизъюнкторов (рис. 12).

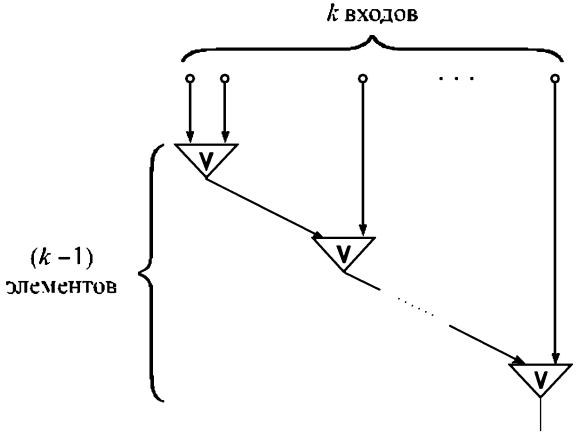


Рис. 12

В итоге получим схему, удовлетворяющую требованию теоремы. Теорема доказана.

При конструктивном доказательстве теоремы 16 строилась схема, в понятном смысле «моделирующая» совершенную д.н.ф. реализуемой булевой функции.

§ 3. Метод Шеннона

Пусть $L(n) = \max L(f)$, где максимум берётся по всем булевым функциям от n переменных; функцию $L(n)$ обычно называют *функцией Шеннона*. Содержательный смысл функции Шеннона прост — это наименьшее возможное число эле-

ментов, достаточное для реализации любой булевой функции от n переменных. Из теоремы 16, очевидно, следует оценка $L(n) \leqslant 3 \cdot 2^n + n - 5$. Эту оценку можно существенно улучшить, если воспользоваться методом синтеза схем, предложенным Шенноном. С использованием этого метода получим оценку

$$L(n) \lesssim \frac{8 \cdot 2^n}{n}$$

(запись $a(n) \lesssim b(n)$ означает, что $\overline{\lim\limits_{n\to\infty}} \frac{a(n)}{b(n)} \leqslant 1$).

Пусть $f(x_1, \ldots, x_n)$ — произвольная булева функция от n переменных. Разложим функцию f по $(n-k)$ переменным:

$$f(x_1, \ldots, x_n) = \bigvee_{(\sigma_1, \ldots, \sigma_{n-k})} x_1^{\sigma_1} \ldots x_{n-k}^{\sigma_{n-k}} f(\sigma_1, \ldots, \sigma_{n-k}, x_{n-k+1}, \ldots, x_n). \quad (1)$$

Схема S для функции f строится из трёх блоков (подсхем), показанных на рис. 13:

1) блока, реализующего $Q_{n-k}(x_1, \ldots, x_{n-k})$;

2) блока, реализующего систему $V_k(x_{n-k+1}, \ldots, x_n)$ всех 2^{2^k} булевых функций от k переменных $x_{n-k+1}, \ldots, x_n$;

3) блока, осуществляющего соединение первых двух блоков в соответствии с представлением (1).

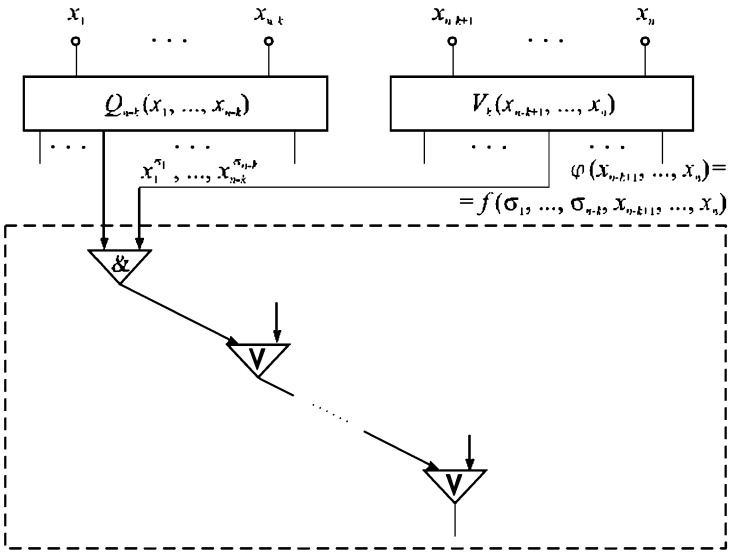


Рис. 13

В третьем блоке на каждое слагаемое из (1) приходится не более одного конъюнктора и не более одного дизъюнктора. Следовательно,

$$L(S) \leqslant L(Q_{n-k}) + L(V_k) + 2 \cdot 2^{n-k}. \tag{2}$$

Реализуя каждую функцию из V_k отдельной подсхемой и используя теорему 16, получаем оценку

$$L(V_k) \leqslant 2^{k+2} \cdot 2^{2^k}. \tag{3}$$

Из (2), (3) и теоремы 15 имеем

$$L(S) \leqslant 4 \cdot 2^{n-k} + 2^{k+2} \cdot 2^{2^k} + n - k - 4. \tag{4}$$

Положим (при достаточно больших n)

$$k = \lfloor \log(n - 3\log n) \rfloor.$$

(Здесь и далее используются обозначения: $\lfloor a \rfloor$ — целая часть числа a; $\log a$ — двоичный логарифм a.)

Тогда

$$\log(n - 3\log n) - 1 < k \leqslant \log(n - 3\log n),$$
$$\frac{n - 3\log n}{2} < 2^k \leqslant n - 3\log n,$$
$$2^{2^k} \leqslant \frac{2^n}{n^3}.$$

Из (4) и последних оценок следует, как нетрудно убедиться, требуемая оценка

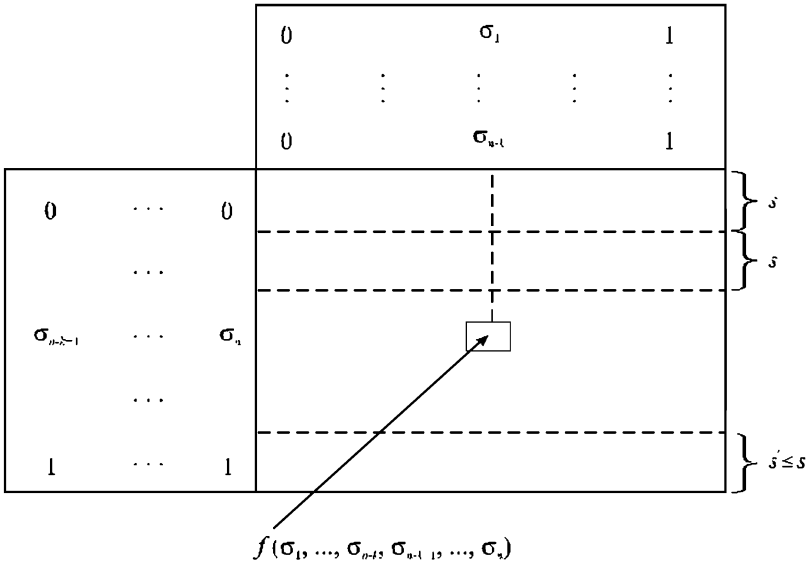
$$L(n) \lesssim \frac{8 \cdot 2^n}{n}.$$

§ 4. Асимптотически оптимальный метод синтеза схем (метод Лупанова)

Вначале введём специальное представление булевых функций. Пусть $f(x_1, \ldots, x_n)$ — произвольная булева функция; зададим её таблицей (табл. 8).

В этой таблице значение функции f на наборе $(\sigma_1, \ldots, \sigma_{n-k}, \sigma_{n-k+1}, \ldots, \sigma_n)$ помещается на пересечении столбца, соответствующего $(\sigma_1, \ldots, \sigma_{n-k})$, и строки, соответствующей $(\sigma_{n-k+1}, \ldots, \sigma_n)$. Разобьём по строкам таблицу на полосы $A_1, \ldots$

Таблица 8



$\ldots, A_p$ по s строк в полосе (последняя полоса может содержать меньшее число строк). Ясно, что

$$p \leqslant \frac{2^k}{s} + 1.$$

Пусть f_i — функция, совпадающая с f на полосе A_i и равная 0 вне этой полосы. Очевидно, что

$$f(x_1, \ldots, x_n) = \bigvee_{i=1}^{p} f_i(x_1, \ldots, x_n) =$$

$$= \bigvee_{i=1}^{p} \bigvee_{(\sigma_1, \ldots, \sigma_{n-k})} x_1^{\sigma_1} \ldots x_{n-k}^{\sigma_{n-k}} f_i(\sigma_1, \ldots, \sigma_{n-k}, x_{n-k+1}, \ldots, x_n) =$$

$$= \bigvee_{(\sigma_1, \ldots, \sigma_{n-k})} x_1^{\sigma_1} \ldots x_{n-k}^{\sigma^{n-k}} \bigvee_{i=1}^{p} f_i(\sigma_1, \ldots, \sigma_{n-k}, x_{n-k+1}, \ldots, x_n). \quad (1)$$

Заметим, что каждая из функций $f_i(\sigma_1, \ldots, \sigma_{n-k}, x_{n-k+1}, \ldots, x_n)$ задаётся одним столбцом таблицы для функции f_i и поэтому принимает значение 1 не более чем на s наборах; число различных функций $f_i(\sigma_1, \ldots, \sigma_{n-k}, x_{n-k+1}, \ldots, x_n)$ (при фиксированном i) не превосходит 2^s.

Опишем теперь метод синтеза, основанный на специальном представлении (1). Схема S для функции f составляется из

6 блоков (соединение блоков условно показано на рис. 14; двойной линией очерчен блок, содержащий «почти все» элементы схемы, что будет видно из дальнейшего).

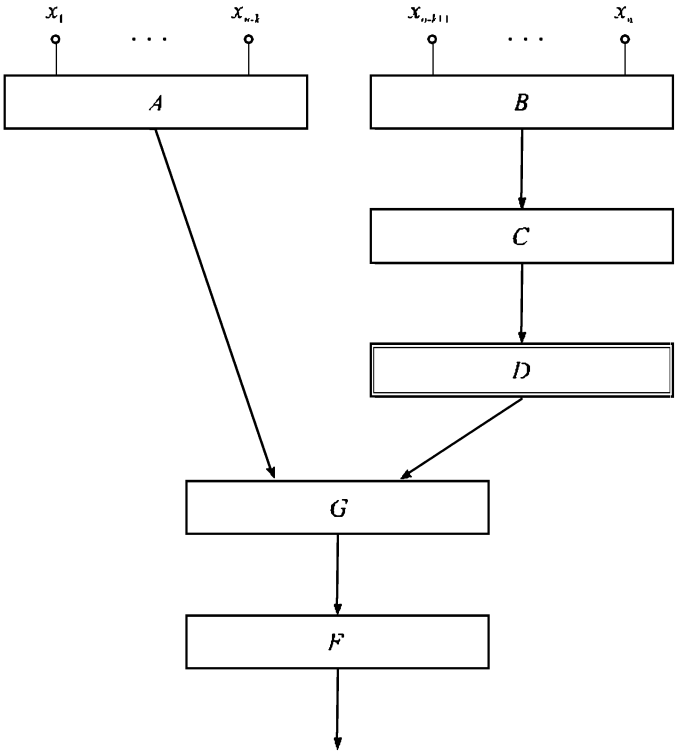


Рис. 14

Блок A реализует систему $Q_{n-k}(x_1, \ldots, x_{n-k})$ всех конъюнкций переменных $x_1, \ldots, x_{n-k}$; по теореме 15

$$L(A) \leqslant 2^{n-k+1} + n - k - 4.$$

Блок B реализует систему $Q(x_{n-k+1}, \ldots, x_n)$ всех конъюнкций переменных $x_{n-k+1}, \ldots, x_n$; по теореме 15

$$L(B) \leqslant 2^{k+1} + k - 4.$$

Блок C реализует все различные функции $f_i(\sigma_1, \ldots$ $\ldots, \sigma_{n-k}, x_{n-k+1}, \ldots, x_n)$. Из сказанного выше следует, что для реализации одной такой функции (с использованием конъюнкций, реализованных блоком B) требуется не более, чем

s дизъюнкторов (или один конъюнктор, если функция равна 0). А в целом

$$L(C) \leqslant ps \cdot 2^s.$$

Блок D реализует все функции

$$\bigvee_{i=1}^{p} f_i(\sigma_1, \ldots, \sigma_{n-k}, x_{n-k+1}, \ldots, x_n) =$$

$$= f(\sigma_1, \ldots, \sigma_{n-k}, x_{n-k+1}, \ldots, x_n);$$

$$L(D) \leqslant p \cdot 2^{n-k}.$$

Блок G осуществляет умножение

$$x_i^{\sigma_1} \ldots x_{n-k}^{\sigma_{n-k}} f(\sigma_1, \ldots, \sigma_{n-k}, x_{n-k+1}, \ldots, x_n);$$

$$L(G) \leqslant 2^{n-k}.$$

Наконец, блок F реализует функцию $f(x_1, \ldots, x_n)$ как дизъюнкцию функций, реализованных блоком G;

$$L(F) < 2^{n-k}.$$

Таким образом,

$$L(S) \leqslant L(A) + L(B) + L(C) + L(D) + L(G) + L(F) <$$
$$< 2^{n-k+1} + n - k - 4 + 2^{k+1} + k - 4 + ps2^s + p2^{n-k} + 2^{n-k+1} \leqslant$$
$$\leqslant 5 \cdot 2^{n-k} + 3 \cdot 2^k + \left(\frac{2^k}{s} + 1\right)\left(2^{n-k} + s \cdot 2^s\right).$$

Положим $k = \lfloor 3 \log n \rfloor$, $s = \lfloor n - 5 \log n \rfloor$. Непосредственной проверкой нетрудно убедиться, что при таком выборе параметров k и s

$$L(S) \leqslant \frac{2^n}{n}\left(1 + O\left(\frac{\log n}{n}\right)\right),$$

и, следовательно, имеет место

Теорема 17. $L(n) \leqslant \dfrac{2^n}{n}\left(1 + O\left(\dfrac{\log n}{n}\right)\right)$.

§ 5. Мощностной метод получения нижней оценки для сложности схем

Общий приём, с использованием которого можно получить нижнюю оценку для $L(n)$, был предложен К. Шенноном. Этот приём основан на том соображении, что схем малой сложности мало и их не хватает для реализации всех функций от n пере-

менных, в связи с чем существуют функции, требующие схем большей сложности.

Обозначим через $N(n,k)$ число различных (неизоморфных) схем из функциональных элементов в рассматриваемом базисе $\{\&, \vee, {}^-\}$, имеющих не более n полюсов (входов), соответствующих некоторым переменным из множества $\{x_1, \ldots, x_n\}$, один выход и не более k элементов. Кроме того будем рассматривать схемы, у которых каждый вход и каждый элемент, не являющийся выходным элементом (выходом) схемы, соединены выходящими из них дугами с некоторыми элементами (иначе полюс или элемент можно из схемы удалить и реализовать заданную функцию более «простой» схемой с меньшим числом входов или элементов). Получение нижней оценки основано на следующем утверждении.

Лемма 6 (о нижней оценке). *Если для некоторой функции $k(n)$ при $n \to \infty$ выполнено условие*

$$\frac{N(n,k(n))}{2^{2^n}} \to 0, \tag{1}$$

то, начиная с некоторого n, имеет место оценка

$$L(n) > k(n),$$

причём доля тех функций $f(x_1, \ldots, x_n)$, для которых $L(f) \leqslant k(n)$, стремится к 0 с ростом n.

Доказательство. Каждая функция $f(x_1, \ldots, x_n)$, такая, что $L(f) \leqslant k(n)$, реализуется некоторой схемой, содержащей не более n полюсов и не более $k(n)$ элементов, причём разные функции требуют разных схем. Поэтому число таких функций f не превосходит $N(n,k(n))$, и в силу условия (1) их доля среди всех 2^{2^n} функций от n переменных стремится к 0 с ростом n. Тем самым вторая часть утверждения леммы доказана.

Из (1) следует, что, начиная с некоторого n, выполняется неравенство

$$\frac{N(n,k(n))}{2^{2^n}} < \frac{1}{2},$$

т. е. при этих n по крайней мере половина всех функций от n переменных имеет сложность выше $k(n)$, а потому $L(n) > k(n)$. Лемма доказана.

Таким образом, задача получения нижней оценки для функции Шеннона $L(n)$ свелась к оценке величины $N(n,k(n))$ и к подбору функции $k(n)$, имеющей по возможности бóльшие значения и удовлетворяющей условию леммы.

Оценим число схем $N(n,k(n))$.

Лемма 7. $N(n,k) \leqslant (32(n+k))^{n+k+3}$.

Доказательство. Рассмотрим схему, содержащую $n' \leqslant n$ входов, k_1 инверторов и k_2 конъюнкторов и дизъюнкторов. Эта схема обладает следующими свойствами:

1) она имеет $n'+k_1+k_2$ вершин, причём некоторые из этих вершин отмечены символами переменных $x_{i_1},\dots,x_{i_{n'}}$ из множества $\{x_1,\dots,x_n\}$, а остальные — символами $\&, \vee, {}^-$; кроме того, вершине, из которой не выходит ни одна дуга (такая вершина единственна), приписана метка $*$;

2) она имеет k_1+2k_2 дуг, причём в каждую вершину, отмеченную символом $^-$, входит одна дуга, а в каждую вершину, отмеченную символом $\&$ или $\vee$, входят две дуги;

3) для каждой вершины существует путь, ведущий из этой вершины в вершину, помеченную символом $*$ (поскольку из каждого входа и из каждого элемента, отличного от выходного, выходят дуги, а контуры отсутствуют).

Обозначим через $S(n,n',k_1,k_2)$ множество всех схем, обладающих перечисленными тремя свойствами, а через $N(n,n',k_1,k_2)$ — их число. Ясно, что

$$N(n,k) \leqslant \sum_{\substack{n',k_1,k_2:\\ n'\leqslant n,k_1+k_2\leqslant k}} N(n,n',k_1,k_2). \tag{2}$$

Оценим величину $N(n,n',k_1,k_2)$.

Для произвольной схемы из $S(n,n',k_1,k_2)$ возьмём соотнесённый граф. Очевидно, что в этом графе можно выделить дерево с корнем в вершине $*$, содержащее все вершины графа (такое дерево можно получить, удаляя из исходного связного соотнесённого графа в произвольной последовательности циклические рёбра). Исходя из сказанного, каждую схему из $S(n,n',k_1,k_2)$ можно построить следующим образом.

1) Выбирается дерево с корнем, содержащее $n'+k_1+k_2$ вершин и $n'+k_1+k_2-1$ рёбер; согласно верхней оценке для числа корневых деревьев (теорема 4) это можно сделать не более, чем $4^{n'+k_1+k_2} \leqslant 4^{n+k_1+k_2}$ способами.

2) Корню приписывается символ $*$, а дуги ориентируются; это делается не более, чем $2^{n+k_1+k_2}$ способами.

3) Из множества $\{x_1,\dots,x_n\}$ выбираются n' переменных $x_{i_1},\dots,x_{i_{n'}}$; для этого имеется $C_n^{n'} \leqslant 2^n$ возможностей.

4) Одна из вершин отмечается символом x_{i_1}, другая — символом x_{i_2} и т. д., некоторая n'-я вершина отмечается символом $x_{i_{n'}}$; для выбора каждой вершины имеется не более $n'+k_1+k_2$ воз-

можностей, поэтому вариантов выбора всех вершин не больше $(n'+k_1+k_2)^{n'} \leqslant (n+k_1+k_2)^n$.

5) Среди k_1+k_2 вершин, не отмеченных символами переменных, выбираются k_1 вершин и отмечаются символом $^-$; это можно сделать $C_{k_1+k_2}^{k_1} \leqslant 2^{k_1+k_2}$ способами. Каждая из остальных k_2 вершин отмечается каким-нибудь символом из $\{\&, \vee\}$; для этого имеется 2^{k_2} возможностей.

6) Полученный граф (ориентированное дерево) содержит $n'+k_1+k_2-1$ дуг (на одну меньше числа вершин), а всего в графе должно быть (k_1+2k_2) дуг. Для недостающих k_1+2k_2- $-(n'+k_1+k_2-1)=k_2-n'+1$ дуг известны вершины, в которые они входят (поскольку в каждую вершину, помеченную $\&$ или $\vee$, должны входить две дуги, а в каждую вершину, помеченную $^-$, — одна дуга). Второй конец каждой из этих дуг можно выбрать не более чем $n'+k_1+k_2$ способами; при этом всего вариантов выбора не больше $(n'+k_1+k_2)^{k_2} \leqslant (n+k_1+k_2)^{k_2}$. После этого шага схема полностью построена.

Величина $N(n, n', k_1, k_2)$ не превосходит произведения оценок из п. 1)–6):

$$N(n,n',k_1,k_2) \leqslant 4^{n+k_1+k_2} \cdot 2^{n+k_1+k_2} \times$$
$$\times 2^n (n+k_1+k_2)^n \cdot 2^{k_1+k_2} \cdot 2^{k_2} (n+k_1+k_2)^{k_2} \leqslant$$
$$\leqslant (32(n+k_1+k_2))^{n+k_1+k_2} = (32(n+k))^{n+k}.$$

При суммировании в (2) по n', k_1 и k_2 для вычисления $N(n,k)$ индекс n' может принимать не более n значений (от 1 до n), а каждый из индексов k_1 и k_2 — не более $k+1$ значений (от 0 до k). Поэтому

$$N(n,k) \leqslant n(k+1)^2 (32(n+k))^{n+k}.$$

Отсюда, поскольку $n \geqslant 1$, окончательно получаем

$$N(n,k) \leqslant (32(n+k))^{n+k+3}.$$

Лемма доказана.

С использованием последних двух лемм докажем теперь следующее утверждение.

Теорема 18 (нижняя оценка сложности схем). *При всех достаточно больших n*

$$L(n) > \frac{2^n}{n},$$

и доля тех функций $f(x_1, \dots, x_n)$, *для которых* $L(f) \leqslant \frac{2^n}{n}$, *стремится к* 0 *с ростом* n.

Доказательство. Для доказательства теоремы достаточно проверить выполнимость условия (1) леммы 6 и применить затем эту лемму при $k(n) = \frac{2^n}{n}$. Условие (1), очевидно, эквивалентно при этом условию

$$\log N\big(n, \frac{2^n}{n}\big) - 2^n \to -\infty. \tag{3}$$

Используя лемму 7, получаем цепочку соотношений

$$\begin{aligned}
\log N\big(n, \frac{2^n}{n}\big) - 2^n &\leqslant \big(n + \frac{2^n}{n} + 3\big)\big(5 + \log\big(n + \frac{2^n}{n}\big)\big) - 2^n \leqslant \\
&\leqslant \big(n + \frac{2^n}{n} + 3\big)\big(5 + \log \frac{2^{n+2}}{n}\big) - 2^n = \\
&= \big(n + \frac{2^n}{n} + 3\big)\big(n + 7 - \log n\big) - 2^n = \\
&= -\frac{2^n}{n}\log n + 7\frac{2^n}{n} + n^2 - n\log n + 10n - 3\log n + 21 = \\
&= -\frac{2^n}{n}\log n(1 - o(1)).
\end{aligned}$$

Последняя величина стремится к $-\infty$, и условие (3), а с ним и условие (1) леммы 6, оказывается выполненным. Теорема доказана.

Сопоставляя верхнюю и нижнюю оценки для $L(n)$ (теоремы 17 и 18), получаем следующее утверждение.

Теорема 19 (О.Б. Лупанова). *Имеет место асимптотическое равенство*

$$L(n) \sim \frac{2^n}{n},$$

причём доля тех функций $f(x_1, \dots, x_n)$, *для которых* $L(f) \leqslant \frac{2^n}{n}$, *стремится к* 0 *с ростом* n.

Таким образом, почти все булевы функции являются асимптотически самыми сложными, и метод Лупанова строит для почти всех функций почти наилучшие (асимптотически наилучшие) схемы.

Глава VI

ТЕСТЫ

§ 1. Полные диагностические тесты для таблиц. Оценки длины тестов

Пусть M — произвольная прямоугольная булева таблица с попарно различными столбцами. Подмножество T строк таблицы M называется *полным диагностическим тестом* (или просто *тестом*) этой таблицы, если для любых двух столбцов этой таблицы в T найдётся строка, на пересечении с которой данные столбцы содержат различные элементы. Число строк, содержащихся в тесте T, называется *длиной* этого теста. Тест наименьшей возможной длины для данной таблицы называется *минимальным*.

Теорема 20 (верхняя оценка длины теста таблицы). *Для любой булевой таблицы, содержащей m попарно различных столбцов, существует полный диагностический тест, длина которого не превосходит $m-1$.*

Доказательство проведём индукцией по числу столбцов в таблице. При $m = 1, 2$ утверждение очевидно: при $m = 1$ тест пуст, а при $m = 2$ в качестве теста можно взять любую строку, в которой столбцы содержат различные значения. Предположим, что утверждение справедливо для всех рассматриваемых таблиц, содержащих $1, 2, \ldots, k$ столбцов; докажем его для таблиц, содержащих $k+1$ столбцов.

Пусть M — произвольная булева таблица, содержащая $k+1$ попарно различных столбцов. Произвольным образом выберем в M строку A, содержащую хотя бы один нуль и хотя бы одну единицу; такая строка найдётся, поскольку все столбцы таблицы M по условию попарно различны. Исходную таблицу M разобьём на две непересекающиеся подтаблицы M_0 и M_1, причём к M_0 (к M_1) отнесём все столбцы из M, содержащие в строке A одни лишь нули (соответственно одни лишь единицы). Число столбцов в M_i обозначим через m_i $(i = 0, 1)$; очевидно,

что $m_0 + m_1 = k + 1$. Из попарного различия всех столбцов в исходной таблице M и из способа разбиения M на M_0 и M_1 непосредственно следует, что: а) все столбцы в M_i, $i = 0, 1$, попарно различны; б) объединение тестов для M_0 и M_1 со строкой A дает тест для M. Отсюда и из предположения индукции следует, что для M_i, $i = 0, 1$, можно построить полный диагностический тест из не более, чем $m_i - 1$ строк. Объединяя тесты для M_0 и M_1 со строкой A, получим полный диагностический тест для исходной таблицы M, который содержит не более, чем $(m_0 - 1) + (m_1 - 1) + 1 = k$ строк. Теорема доказана.

Отметим, что оценка теоремы 20 неулучшаема (в общем случае). Это следует, например, из табл. 9, в которой столбец S_1 содержит только нули, а каждый из остальных столбцов S_i, $i = 2, \ldots, m$, содержит ровно одну единицу, находящуюся в $(i-1)$-й строке. Легко заметить, что полный диагностический тест для этой таблицы должен содержать все $m - 1$ строк с единицами.

Таблица 9

S_1	S_2	S_3	S_4	...	S_m
0	1	0	0	...	0
0	0	1	0	...	0
0	0	0	1	...	0
...	...	...	...	...	...
0	0	0	0	...	1

Теорема 21 (нижняя оценка длины теста таблицы). *Полный диагностический тест для всякой таблицы, содержащей m попарно различных столбцов, содержит не менее $\lceil \log m \rceil$ строк* (через $\lceil a \rceil$ обозначается наименьшее целое число, не меньшее a).

Доказательство проведём индукцией по m. При $m = 1, 2$ оценка очевидна. Предположим, что оценка справедлива для таблиц, содержащих $1, 2, \ldots, k$ столбцов. Возьмём произвольную таблицу M, содержащую $k + 1$ попарно различных столбцов. Пусть T — тест минимальной длины для этой таблицы, содержащий строки $S_1, S_2, \ldots, S_r$. Возьмём строку S_1 и разобьём таблицу T на две подтаблицы T_0 и T_1: подтаблица T_i ($i \in \{0, 1\}$) состоит из столбцов, содержащих i в строке S_1. Ясно, что одна из этих подтаблиц, например, T_0, содержит не менее $\frac{k+1}{2}$ попарно различных столбцов и $r - 1$ строк $S_2, \ldots, S_{r-1}$

должны составлять тест этой подтаблицы. Следовательно, по предположению индукции должно выполняться неравенство

$$r - 1 \geqslant \left\lceil \log \frac{k+1}{2} \right\rceil,$$

из которого следует требуемое неравенство

$$r \geqslant \lceil \log(k+1) \rceil.$$

Теорема доказана.

Легко заметить, что нижняя оценка, доставляемая теоремой 21, достигается на таблице, столбцы которой имеют высоту $\lceil \log m \rceil$ и являются двоичными записями чисел $0, 1, \ldots, m-1$.

§ 2. Тесты для схем. Построение минимальных тестов методом Яблонского

Будем рассматривать схемы из функциональных элементов в базисе $\{\&, \vee, \bar{}\}$. Предположим, что на схемы воздействует некоторый источник неисправностей и в результате такого воздействия некоторые элементы схем могут приходить в неисправные состояния. Для определённости ограничим класс неисправностей константными неисправностями на выходах элементов. В этом случае каждый неисправный элемент схемы выдаёт либо константу 0, либо константу 1 (при этом не исключается одновременное наличие в схеме неисправных элементов, реализующих константу 0, и неисправных элементов, реализующих константу 1).

Пусть S — некоторая схема из функциональных элементов, реализующая булеву функцию $f(\tilde{x})$, $\tilde{x} = (x_1, \ldots, x_n)$. Допустим, что схема S под воздействием источника неисправностей переходит в некоторую схему S', содержащую неисправные элементы и реализующую некоторую функцию $g(\tilde{x})$, которую обычно называют *функцией неисправности*; функцию неисправности $g(\tilde{x})$ будем считать *тривиальной*, если $g(\tilde{x}) \equiv f(\tilde{x})$, и *нетривиальной* в противном случае.

Всякое множество T входных наборов (т. е. наборов значений переменных, подаваемых на входы) схемы S называется *полным проверяющим тестом* для S, если для любой нетривиальной функции неисправности $g(\tilde{x})$ в T найдётся хотя бы один такой набор $\tilde{\sigma}$, что $f(\tilde{\sigma}) \neq g(\tilde{\sigma})$.

Знание значений, выдаваемых схемой на наборах из полного проверяющего теста, очевидно, позволяет выяснить, является ли

схема исправной и реализует требуемую функцию или же она неисправна и реализует функцию, отличную от требуемой.

Длиной теста называется число наборов, составляющих тест.

Кроме проверяющих, часто рассматриваются ещё и диагностические тесты.

Всякое множество T входных наборов схемы S называется *полным диагностическим тестом* для этой схемы, если T является полным проверяющим тестом для S и, кроме того, для любых двух различных функций неисправности $g'(\tilde{x})$ и $g''(\tilde{x})$ схемы S в T найдется набор $\tilde{\sigma}$ такой, что $g'(\tilde{\sigma}) \neq g''(\tilde{\sigma})$.

Из данного определения следует, что полный диагностический тест позволяет не только установить факт неисправности схемы, но и указать, какую именно функцию неисправности реализует схема (если она неисправна). При этом, естественно, предполагается, что состояние схемы (т. е. множество неисправных элементов и характер неисправностей этих элементов) за время тестирования не изменяется.

Заметим, что в качестве тривиального теста всегда, очевидно, можно взять тест, содержащий все 2^n входных наборов схемы, реализующей булеву функцию от n переменных. Однако наибольший интерес представляют *минимальные тесты*, т. е. тесты, содержащие наименьшее возможное число наборов. Опишем построение минимальных тестов методом С. В. Яблонского, совмещая общие рассуждения с их иллюстрацией на конкретном примере схемы, изображенной на рис. 15 и реализующей в исправном состоянии функцию $f(x_1, x_2) = x_1 \oplus x_2$. Для большей ясности весь процесс построения разобьём на ряд отдельных этапов.

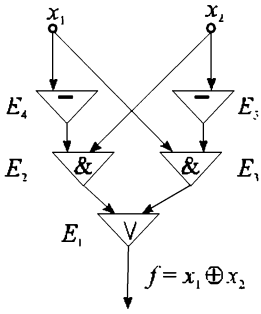


Рис. 15

1. *Нахождение функций неисправности.* Рассмотрим всевозможные комбинации неисправностей элементов и для каждой комбинации найдём соответствующую ей функцию неисправности. В результате получим таблицу функций неисправности заданной схемы (табл. 10). В этой таблице в 1-м, ..., 5-м столбцах указаны состояния соответствующих элементов схемы, причём используемые обозначения имеют следующий смысл: «и» — элемент исправен; «0» — элемент неисправен и реализует константу 0; «1» — элемент неисправен и реализует константу 1; «*» — элемент может находится в любом состоянии (как в исправном,

Таблица 10

Состояния элементов схемы S					Реализуемая схемой функция неисправности
E_1	E_2	E_3	E_4	E_5	
0	*	*	*	*	0
1	*	*	*	*	1
и	0	0	*	*	0
и	0	1	*	*	1
и	0	и	*	0	0
и	0	и	*	1	x_1
и	0	и	*	и	$x_1\&\overline{x}_2$
и	1	*	*	*	1
и	и	0	0	*	0
и	и	0	1	*	x_2
и	и	0	и	*	$\overline{x}_1\&x_2$
и	и	1	*	*	1
и	и	и	0	0	0
и	и	и	0	1	x_1
и	и	и	0	и	$x_1\&\overline{x}_2$
и	и	и	1	0	x_2
и	и	и	1	1	$x_1 \vee x_2$
и	и	и	1	и	$x_1 \vee x_2$
и	и	и	и	0	$\overline{x}_1\&x_2$
и	и	и	и	1	$x_1 \vee x_2$
и	и	и	и	и	$\overline{x}_1x_2 \vee x_1\overline{x}_2 = x_1 \oplus x_2$

так и в неисправном). В последнем столбце таблицы указаны все функции неисправности.

Первая (вторая) строка таблицы соответствует случаю, когда выходной элемент E_1 схемы неисправен и реализует константу 0 (1); в этом случае независимо от состояния остальных элементов схемы на её выходе будет реализована константа 0 (1), что и указано в верхних двух строках таблицы. Если элемент E_1 исправен, а элементы E_2, E_3 неисправны и реализуют константу 0, то независимо от состояний элементов E_4, E_5 на выходе схемы будет реализована константа 0, что и указано в третьей строке таблицы. Четвёртая строка таблицы соответствует случаю, когда элемент E_1 исправен, E_2 реализует константу 0, E_3 — констан-

ту 1, а E_4, E_5 могут быть в любых состояниях — при указанных состояниях элементов E_2, E_3 состояния элементов E_4, E_5 уже не влияют на реализуемую схемой функцию. Подобным образом заполняется и остальная часть таблицы.

Как видно из этой таблицы, учёт того обстоятельства, что иногда состояние лишь части элементов схемы полностью определяет реализуемую схемой функцию неисправности, позволяет значительно сократить размеры таблицы (в нашем примере полное перечисление всевозможных состояний элементов привело бы к таблице, содержащей $3^5 = 243$ строки).

2. *Табличное задание функций неисправности.* Исходную функцию f и все попарно различные функции неисправности $g_1, \ldots, g_7$ зададим табл. 11.

Таблица 11

x_1	x_2	$f = x_1 \oplus x_2$	$g_1 = 0$	$g_2 = 1$	$g_3 = x_1$	$g_4 = x_1\overline{x}_2$	$g_5 = x_2$	$g_6 = \overline{x}_1 x_2$	$g_7 = x_1 \vee x_2$
0	0	0	0	1	0	0	0	0	0
0	1	1	0	1	0	0	1	1	1
1	0	1	0	1	1	1	0	0	1
1	1	0	0	1	1	0	1	0	1

Задача поиска минимального диагностического теста для заданной схемы теперь, очевидно, сведена к задаче построения минимального (т. е. содержащего наименьшее возможное число строк) теста таблицы, составленной из столбцов значений функций f, $g_1, g_2, \ldots, g_7$.

3. *Построение вспомогательной таблицы.* По табл. 11 функций неисправности составим табл. 12, в которой строки соответствуют всевозможным парам функций из табл. 11, а столбцы — наборам значений переменных x_1, x_2; на пересечении i-й строки и j-го столбца ставим 1 в том и только в том случае, когда значения двух функций, соответствующих i-й строке, различны на входном наборе, соответствующем j-му столбцу, $i = 1, 2, \ldots, 28$; $j = 1, \ldots, 4$.

4. *Построение теста.* Дальнейшая задача — собственно построение теста — заключается теперь в решении задачи на покрытие для табл. 12: нужно из множества столбцов выбрать наименьшее по мощности подмножество (покрытие) такое, чтобы содержащиеся в этих столбцах единицы покрывали в совокупности все строки таблицы (всякая единица в строке по определению покрывает эту строку). Нетрудно заметить, что найденное

Таблица 12

Пары функций	Входные наборы			
	1-й набор (0, 0)	2-й набор (0, 1)	3-й набор (1, 0)	4-й набор (1, 1)
(f, g_1)		1	1	
(f, g_2)	1			1
(f, g_3)		1		1
(f, g_4)		1		
(f, g_5)			1	1
(f, g_6)			1	
(f, g_7)				1
(g_1, g_2)	1	1	1	1
(g_1, g_3)			1	1
(g_1, g_4)			1	
(g_1, g_5)		1		1
(g_1, g_6)		1		
(g_1, g_7)		1	1	1
(g_2, g_3)	1	1		
(g_2, g_4)	1	1		1
(g_2, g_5)	1		1	
(g_2, g_6)	1		1	1
(g_2, g_7)	1			
(g_3, g_4)				1
(g_3, g_5)		1	1	
(g_3, g_6)		1	1	1
(g_3, g_7)		1		
(g_4, g_5)		1	1	1
(g_4, g_6)		1	1	
(g_4, g_7)		1		1
(g_5, g_6)				1
(g_5, g_7)			1	
(g_6, g_7)			1	1

при этом подмножество столбцов, т. е. подмножество входных наборов нашей схемы, даст некоторый минимальный полный диагностический тест: ведь поскольку для каждой строки имеется столбец в найденном подмножестве, покрывающий эту строку

(т. е. имеющий единицу на пересечении с этой строкой), то это означает, что в найденном подмножестве наборов значений переменных есть набор, на котором принимают разные значения функции из пары, соответствующей данной строке.

Рассматриваемый ниже способ решения задачи на покрытие заключается в том, что вначале по табл. 12 составляется некоторое специальное выражение — «произведение сумм» $\prod\sum$, затем осуществляется переход к «сумме произведений» ($\sum\prod$), последнее выражение упрощается и, наконец, из него выбирается слагаемое, соответствующее минимальному тесту.

Поставим в соответствие каждому набору табл. 12 свою букву: 1-му набору — a_1, 2-му — a_2, ..., 4-му — a_4. Для i-й строки составим дизъюнкцию D_i некоторых букв из множества $\{a_1, \ldots, a_4\}$ согласно правилу: буква a_j входит в D_i в том и только в том случае, когда j-й столбец содержит единицу в пересечении с i-й строкой ($i = 1, \ldots, 28$; $j = 1, \ldots, 4$). Объединяя $D_1, \ldots, D_{28}$ в произведение, получим

$$\prod\sum = (a_2 \vee a_3)(a_1 \vee a_4)(a_2 \vee a_4)a_2(a_3 \vee a_4)a_3a_4\& \\ \&(a_1 \vee a_2 \vee a_3 \vee a_4)(a_3 \vee a_4)a_3(a_2 \vee a_4)a_2(a_2 \vee a_3 \vee a_4)(a_1 \vee a_2)\& \\ \&(a_1 \vee a_2 \vee a_4)(a_1 \vee a_3)(a_1 \vee a_3 \vee a_4)a_1a_4(a_2 \vee a_3)(a_2 \vee a_3 \vee a_4)\& \\ \&a_2(a_2 \vee a_3 \vee a_4)(a_2 \vee a_3)(a_2 \vee a_4)a_4a_3(a_3 \vee a_4).$$

Далее нужно раскрыть скобки и упростить (используя при необходимости свойства ассоциативности и коммутативности операций конъюнкции и дизъюнкции) полученное выражение в соответствии с соотношениями

$$a \cdot a = a, \quad A \vee AB = A,$$

где a — буква, A и B — некоторые произведения букв. Но если в исходном произведении сумм имеются сомножители вида $a_i(a_i \vee a_j \vee \ldots \vee a_t)$, то с учетом дальнейших упрощений сомножитель $(a_i \vee a_j \vee \ldots \vee a_t)$ можно отбросить, т.к. $a_i(a_i \vee a_j \vee \ldots \vee a_t) = a_i$. С учётом этого замечания вместо исходного $\prod\sum$ получим выражение

$$a_1a_2a_3a_4,$$

которое уже не требует дальнейших упрощений и задает единственный (тривиальный) полный диагностический тест, включающий все четыре набора значений переменных x_1x_2.

В общем случае после раскрытия скобок и упрощений получается выражение $\sum\prod$, в котором каждое слагаемое соответ-

ствует некоторому *тупиковому тесту*, т. е. тесту, из которого уже нельзя выбросить ни одного набора. Последнее утверждение следует из того, что при раскрытии скобок в $\prod\sum$ (т. е. при перемножении) получаются все без исключения тесты, а после проведенных преобразований в соответствии с упрощающими соотношениями останутся слагаемые, в каждом из которых нельзя выбросить уже ни одного сомножителя, т. е. эти слагаемые соответствуют тупиковым тестам. Среди полученных тупиковых тестов будут и все минимальные тесты.

Для прежней схемы S, пользуясь тем же способом, найдём теперь минимальный полный проверяющий тест. В этом случае в табл. 12 следует оставить только верхние семь строк, отвечающих парам функций вида (f, g_i), содержащим исходную функцию f. В качестве $\prod\sum$ имеем выражение

$$(a_2 \vee a_3)(a_1 \vee a_4)(a_2 \vee a_4)a_2(a_3 \vee a_4)a_3a_4,$$

от которого после соответствующих преобразований остаётся единственное слагаемое

$$a_2a_3a_4;$$

это слагаемое задаёт единственный (в данном примере) минимальный полный проверяющий тест $T = \{(0,1),(1,0),(1,1)\}$ из трех наборов.

§ 3. Верхние оценки длины единичных тестов для схем

Рассмотрим частный (но вместе с тем интересный и важный) случай, когда решается задача диагностики схем при наличии в них не более одного неисправного элемента. Понятие *единичного теста* (проверяющего или диагностического) вводится точно так же, как и выше, но только с учетом принятого ограничения — неисправным в схеме может быть только один элемент.

Теорема 22 (верхняя оценка длины единичного теста для схем). *Для всякой схемы из s функциональных элементов можно построить единичный диагностический тест, содержащий не более $2s$ наборов.*

Доказательство. Поскольку для каждого из s элементов схемы возможны лишь два неисправных состояния (в одном из них элемент выдает константу 0, а в другом — константу 1), то общее число попарно различных нетривиальных функций неисправности в рассматриваемом случае не превосходит $2s$. Возьмём таблицу T, задающую исходную функцию f (реализуемую

схемой в исправном состоянии) и все нетривиальные функции неисправности. Таблица T содержит не более, чем $(2s+1)$ столбцов, и всякий тест этой таблицы, очевидно, является единичным диагностическим тестом для исходной схемы. По теореме 20 для таблицы T существует тест, длина которого не превосходит $2s$; отсюда следует и утверждение доказываемой теоремы.

Ранее было установлено (теорема 17), что любую булеву функцию от n переменных можно реализовать схемой из функциональных элементов, содержащей асимптотически не более, чем $\frac{2^n}{n}$ элементов. Отсюда и из теоремы 22 вытекает

Теорема 23 (верхняя оценка длины единичного теста). *Любую булеву функцию от n переменных можно реализовать схемой, допускающей единичный диагностический тест, длина которого асимптотически не превосходит $\frac{2^{n+1}}{n}$.*

§ 4. Синтез легкотестируемых схем

Поскольку единичный диагностический тест является также и проверяющим, то утверждения и оценки последних двух теорем остаются в силе и применительно к единичным проверяющим тестам. Эти утверждения можно существенно усилить, а верхние оценки длины единичных проверяющих тестов понизить от экспоненциальных до линейных в случае схем из функциональных элементов в некоторых других базисах, например, в базисе $\{\&, \oplus, 0, 1\}$. Именно такой случай рассмотрим ниже и конструктивно докажем следующее утверждение.

Теорема 24 (о единичных тестах для легкотестируемых схем). *Любую булеву функцию от n переменных можно реализовать схемой из функциональных элементов в базисе $\{\&, \oplus, 0, 1\}$, допускающей при константных неисправностях на выходах элементов «$\&$», «$\oplus$» единичный проверяющий тест длины $n+3$.*

Доказательство. Пусть задана произвольная булева функция $f(x_1, \ldots, x_n)$, существенно зависящая от всех своих переменных (несущественные переменные можно отбросить). Представим эту функцию полиномом Жегалкина

$$f(x_1, \ldots, x_n) = \sum_{(m_1, \ldots, m_i)} x_{m_1} \ldots x_{m_i}$$

(в котором все слагаемые попарно различны и каждое слагаемое определяется соответствующим набором номеров $(m_1, \ldots, m_i)$ вошедших в него переменных). В соответствии с данным представ-

лением строится и схема S, изображённая на рис. 16. В этой схеме каждая вертикальная цепочка Z_j из конъюнкторов реализует некоторое слагаемое (конъюнкцию K_j) из полинома Жегалкина, причём крайняя левая цепочка Z_1 реализует конъюнкцию $K_1 = x_{a_1}x_{a_2}\dots x_{a_r}$, содержащую наименьшее число букв (если в полиноме Жегалкина наименьшее число букв содержат сразу несколько конъюнкций, то в качестве K_1 берётся любая из них). Если в полиноме Жегалкина в качестве слагаемого присутствует константа 1, то к схеме S добавляется ещё один элемент «$\oplus$» и один вход этого элемента соединяется с выходом схемы S, на второй вход подается константа 1 (со входа схемы), а выход становится выходом всей схемы.

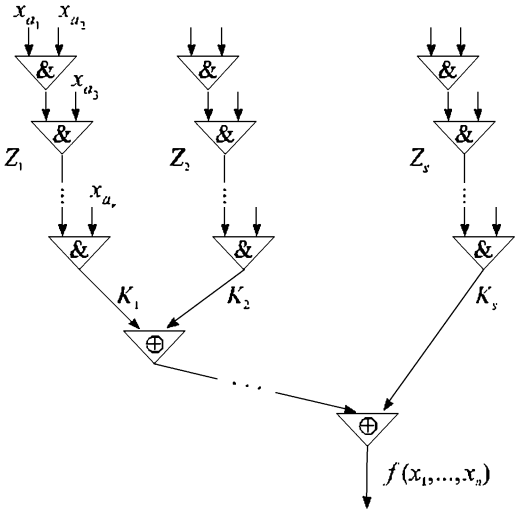


Рис. 16

Рассмотрим множество T из $(n+3)$ входных наборов, в которое входят наборы $\widetilde{\sigma}_1 = (0,1,1,\dots,1,1)$, $\widetilde{\sigma}_2 = (1,0,1,\dots,1,1)$, …, $\widetilde{\sigma}_n = (1,1,1,\dots,1,0)$, $\widetilde{\sigma}_{n+1} = (1,1,1,\dots,1,1)$, $\widetilde{\sigma}_{n+2} = (0,0,0,\dots$ $\dots,0,0)$, $\widetilde{\sigma}_{n+3} = (0,0,\dots,0,\underset{a_1}{1},0,\dots,0,\underset{a_2}{1},0,\dots,0,\underset{a_r}{1},0,\dots,0)$ (последний набор $\widetilde{\sigma}_{n+3}$ содержит единицы только в разрядах, соответствующих переменным $x_{a_1}, x_{a_2}, \dots, x_{a_r}$ из конъюнкции K_1). Покажем, что T — единичный проверяющий тест для S.

Допустим, что в цепочке Z_j, реализующей конъюнкцию K_j, ровно один элемент перейдёт в неисправное состояние и будет выдавать тождественный нуль. В этом случае цепочка Z_j на единичном наборе $\widetilde{\sigma}_{n+1}$ вместо единицы, выдаваемой при

исправности всех элементов, выдаст нуль. Но тогда в сумме по модулю 2, вычисляемой цепочкой из элементов «⊕», вместо одной из единиц окажется нуль, а значит, и на выходе крайнего правого элемента «⊕» вместо значения $f(1,1,\ldots,1)$, выдаваемого исправной схемой, будет $\overline{f}(1,1,\ldots,1)$.

Таким образом, рассматриваемая неисправность на выходе любого одного конъюнктора обнаруживается на наборе $\widetilde{\sigma}_{n+1}$.

Предположим теперь, что какой-то неисправный конъюнктор выдает константу 1; допустим, что этот конъюнктор принадлежит цепочке Z_j и на вход его подаётся переменная x_h. В этом случае на выходе j-й цепочки на наборе $\widetilde{\sigma}_h$ (с единственным нулём в h-м разряде) вместо нуля, выдаваемого цепочкой при исправном состоянии всех её элементов, будет единица. А из-за ошибочного значения на выходе одной цепочки Z_j ошибочное значение будет и на выходе всей схемы.

Таким образом, первые $n+1$ наборов из T позволяют обнаружить любую одиночную неисправность любого конъюнктора схемы.

Среди всех конъюнкций из полинома конъюнкция K_1 содержит наименьшее число букв, и потому любая другая конъюнкция в полиноме содержит по крайней мере одну букву, отсутствующую в K_1. Отсюда следует, что на последнем наборе $\widetilde{\sigma}_{n+3}$ из T конъюнкция K_1 равна единице, а все остальные конъюнкции $K_2,\ldots,K_s$ равны нулю; на предпоследнем нулевом наборе из T все конъюнкции $K_1,K_2,\ldots,K_s$ равны нулю. Следовательно, если на входы схемы S подать $\widetilde{\sigma}_{n+2}$, а затем $\widetilde{\sigma}_{n+3}$, то значение на выходе исправной схемы при переходе от одного набора к другому должно измениться, поскольку в данном случае изменится значение на левом входе одного лишь крайнего левого элемента «⊕». Но если хотя бы один элемент «⊕» в горизонтальной нижней цепочке окажется в неисправном состоянии, то значение на выходе этого элемента, а значит, и на выходе крайнего правого элемента «⊕» (т. е. на выходе схемы) при указанном переходе останется неизменным. А это, очевидно, означает, что последние два набора из T позволяют обнаружить неисправность любого элемента «⊕».

Значит, множество наборов T является единичным проверяющим тестом. Теорема доказана.

Заметим, что при доказательстве теоремы строится *неизбыточная* схема, которая при переходе любого одного элемента в неисправное состояние реализует нетривиальную функцию неисправности.

Глава VII

ОГРАНИЧЕННО-ДЕТЕРМИНИРОВАННЫЕ ФУНКЦИИ И РЕАЛИЗАЦИЯ ИХ АВТОМАТАМИ

§ 1. Детерминированные и ограниченно-детерминированные функции

Возьмём множество E_2^C всех двузначных последовательностей $\delta = \{\delta(1), \delta(2), \ldots, \delta(i), \ldots\}$, где все $\delta(i) \in E_2$, $E_2 = \{0, 1\}$. Обозначим через P_C множество всех функций $f(x_1, \ldots, x_n)$, определённых на наборах $(\alpha_1, \ldots, \alpha_n)$, где $\alpha_i \in E_2^C$ $(i = 1, \ldots, n)$, и принимающих значения из E_2^C. Таким образом, функции из P_C преобразуют наборы двузначных последовательностей в двузначные последовательности.

Пример. Рассмотрим функцию $f(x)$ такую, что $f(\alpha) = f(\{\alpha(1), \alpha(2), \ldots\}) = \{\alpha(1), \alpha(3), \alpha(5), \ldots\}$. Очевидно, что $f(x) \in P_C$.

Набор переменных $(x_1, \ldots, x_n)$ в векторной записи обозначим через X и вместо $f(x_1, \ldots, x_n)$ будем писать также $f(X)$. При этом переменная X принимает значение α из множества наборов $\{(\alpha_1, \ldots, \alpha_n)\}$, компонентами которых являются двузначные последовательности: $\alpha_i = \{\alpha_i(1), \alpha_i(2), \ldots, \alpha_i(m), \ldots\}$, $i = 1, \ldots, n$. Вместе с тем α можно рассматривать как последовательность векторов (наборов из нулей и единиц длины n) $\alpha = \{\alpha(1), \alpha(2), \ldots\}$, где $\alpha(j) = (\alpha_1(j), \ldots, \alpha_n(j))$, $j = 1, 2, \ldots$. При этом можно считать, что последовательность $\alpha = \{\alpha(1), \alpha(2), \ldots\}$ описывает значения последовательности переменных $\{X(1), X(2), \ldots\}$, где $X(m) = (x_1(m), \ldots, x_n(m))$.

Функция $f(X)$ из P_C называется *детерминированной*, если каково бы ни было натуральное число m и каковы бы ни были последовательности α и β, такие, что $\alpha(1) = \beta(1), \ldots, \alpha(m) = \beta(m)$, в последовательностях $\gamma = f(\alpha)$ и $\delta = f(\beta)$ первые m членов также совпадают, т. е. $\gamma(1) = \delta(1), \ldots, \gamma(m) = \delta(m)$.

Множество всех детерминированных функций обозначим через $P_д$.

Для детерминированной функции значение $\gamma(m)$ m-го члена $(m = 1, 2, \ldots)$ последовательности γ полностью определяется значениями m членов последовательности α, т. е. $\gamma_m = f_m(\alpha(1), \ldots, \alpha(m))$. Поэтому детерминированную функцию $f(X)$ можно определить с помощью последовательности булевых функций $f_1(X(1)), f_2(X(1), X(2)), \ldots, f_m(X(1), X(2), \ldots, X(m)), \ldots$, где

$$\begin{aligned} X(1) &= (x_1(1), x_2(1), \ldots, x_n(1)), \\ X(2) &= (x_1(2), x_2(2), \ldots, x_n(2)), \\ &\ldots \\ X(m) &= (x_1(m), x_2(m), \ldots, x_n(m)). \end{aligned}$$

Детерминированная функция $f(x_1, \ldots, x_n)$ может интерпретироваться как функция, описывающая работу дискретного преобразователя с n входами и одним выходом (рис. 17).

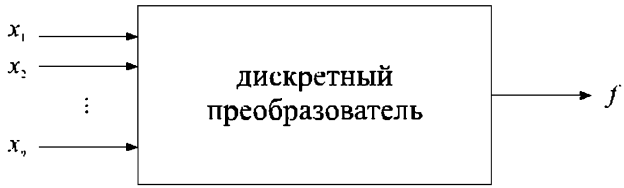


Рис. 17

На входы этого преобразователя в моменты времени $t = 1, 2, \ldots, m, \ldots$ подаются соответствующие значения членов (входных) последовательностей

$$\begin{aligned} \alpha_1 &= \{\alpha_1(1), \alpha_1(2), \ldots, \alpha_1(m), \ldots\}, \\ \alpha_2 &= \{\alpha_2(1), \alpha_2(2), \ldots, \alpha_2(m), \ldots\}, \\ &\ldots \\ \alpha_n &= \{\alpha_n(1), \alpha_n(2), \ldots, \alpha_n(m), \ldots\}. \end{aligned}$$

В эти же моменты времени на выходе появляются соответствующие значения членов (выходной) последовательности $\gamma = \{\gamma(1), \gamma(2), \ldots, \gamma(m), \ldots\}$. Тем самым мы имеем дело с некоторой функцией $f(x_1, \ldots, x_n)$ из $P_д$. Очевидно, что в дискретном преобразователе значение $\gamma(m)$ зависит только от значений входных последовательностей в моменты времени $t = 1, \ldots, m$. Поэтому функция f на выходе дискретного преобразователя есть детерминированная функция. Дискретный преобразователь указанного вида обычно называют *автоматом, реализующим функцию* $f(x_1, \ldots, x_n)$.

Пусть $f(X)$ — функция из $P_{\text{д}}$. Рассмотрим бесконечное корневое ориентированное дерево, представленное на рис. 18.

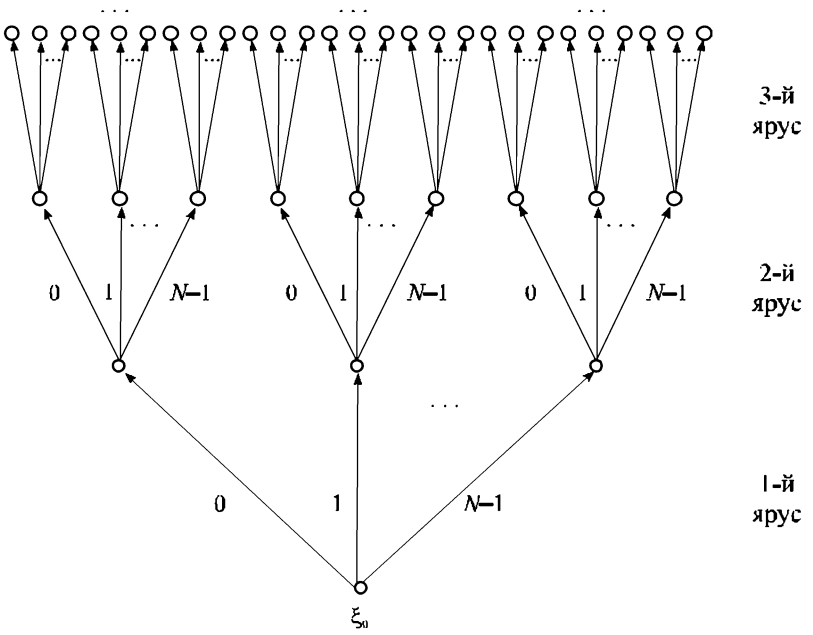


Рис. 18

Из корня ξ_0 этого дерева исходит пучок из $N = 2^n$ дуг, образующих первый ярус. Каждая из дуг первого яруса ведёт в вершину, из которой в свою очередь исходит пучок из N дуг, образующих в совокупности 2-й ярус, и т. д. Вершины, являющиеся концами дуг m-го яруса, также причисляются к m-му ярусу. Вершина ξ_0 считается вершиной нулевого яруса. Дуги каждого пучка нумеруются слева направо числами $0, 1, \ldots, N-1$ или их записями в двоичной системе счисления.

Бесконечный простой (т. е. без повторяющихся вершин) путь с началом в корне ξ_0 будем называть *ветвью* рассматриваемого дерева. Очевидно, что каждой ветви дерева можно сопоставить последовательность $\alpha = \{\alpha(1), \alpha(2), \ldots\}$ номеров дуг, входящих в эту ветвь, если идти по ней, начиная от корня. И наоборот, любой бесконечной последовательности чисел из множества $\{0, 1, \ldots, N-1\}$ однозначно соответствует некоторая ветвь дерева. Таким образом, существует взаимно однозначное соответствие между множеством всех ветвей дерева и множеством всех бесконечных последовательностей чисел из множе-

ства $\{0, 1, \ldots, N-1\}$, которые мы естественным образом можем занумеровать наборами из нулей и единиц длины n.

Как отмечалось выше, детерминированная функция $f(X)$ может быть задана с помощью последовательности булевых функций $f_1(X(1)), f_2(X(1), X(2)), \ldots, f_m(X(1), X(2), \ldots, X(m)), \ldots$. Используя эту последовательность, каждой дуге дерева можно однозначно приписать либо 0, либо 1. Для этого возьмём произвольную дугу m-го яруса ($m = 1, 2, \ldots$) и рассмотрим путь, ведущий из корня дерева к этой дуге. Очевидно, этот путь определён однозначно и характеризуется набором $(\alpha(1), \ldots, \alpha(m))$ номеров дуг пути, отсчитываемых от корня. Рассматриваемой дуге припишем число $\gamma(m) = f_m(\alpha(1), \ldots, \alpha(m))$. Полученное дерево будем называть *занумерованным деревом*.

Таким образом, имея произвольную детерминированную функцию, можно построить соответствующее ей занумерованное дерево. С другой стороны, возьмём бесконечное ориентированное дерево с корнем, из каждой вершины которого исходят ровно 2^n дуг. Занумеруем его произвольным образом, т. е. каждой дуге припишем либо 0, либо 1. Ясно, что полученное занумерованное дерево определит, причём однозначно, некоторую детерминированную функцию, зависящую от переменных $x_1, \ldots, x_n$.

Пусть $f(x_1, \ldots, x_n)$ — произвольная детерминированная функция. Рассмотрим соответствующее ей занумерованное дерево. Пусть ξ — произвольная его вершина из m-го яруса. В неё ведёт из корня ξ_0 путь $\alpha(1), \ldots, \alpha(m)$. Совокупность всех ветвей, исходящих из ξ, порождает некоторое дерево с корнем ξ (являющееся поддеревом исходного дерева). Так как исходное дерево занумеровано, то поддерево с корнем ξ также является занумерованным, и если в этом поддереве ввести свою нумерацию ярусов, начиная с первого, то поддереву будет соответствовать детерминированная функция $f^{\xi}(X)$. Эту функцию можно определить и с помощью последовательности булевых функций следующим образом.

Пусть $f_1(X(1)), f_2(X(1), X(2)), \ldots$ — последовательность булевых функций, задающая детерминированную функцию $f(X)$. Тогда последовательность $f_1^{\xi}(X(1)), f_2^{\xi}(X(1), X(2)), \ldots$ задаёт детерминированную функцию $f^{\xi}(X)$, если для любого $i = 1, 2, \ldots$

$$f_i^{\xi}(X(1), \ldots, X(i)) = f_{m+i}(\alpha(1), \ldots, \alpha(m), X(1), \ldots, X(i)).$$

Два поддерева исходного дерева, имеющие корни ξ_1 и ξ_2, называются *эквивалентными*, если $f^{\xi_1}(X) \equiv f^{\xi_2}(X)$.

Это отношение эквивалентности позволяет в исходном дереве множество всех вершин, а тем самым и множество всех поддеревьев, разбить на классы эквивалентности.

Число r классов эквивалентности, на которые разбивается множество всех поддеревьев данного занумерованного дерева, называется *весом дерева* и, соответственно, *весом детерминированной функции*.

Для занумерованных деревьев можно ввести нумерацию вершин. Сначала перенумеруем классы эквивалентности числами $0, 1, \ldots$ так, чтобы класс, в который попадает исходное дерево, имел номер 0. Далее, взяв произвольную вершину ξ, определяем класс, в который попадает дерево с корнем ξ. Пусть $\varkappa$ – номер этого класса. Тогда вершине ξ приписывается номер $\varkappa$.

Рассмотрим дерево с занумерованными дугами и вершинами. Возьмём произвольную ветвь; она проходит через вершины $\xi_0, \xi_1, \ldots, \xi_i, \ldots, \xi_j, \ldots$. Пусть этим вершинам приписаны соответственно номера $0, \varkappa_1, \ldots, \varkappa_i, \ldots, \varkappa_j, \ldots$. Допустим, что для всех пар (i, j) $(i \neq j)$, таких, что $\varkappa_i = \varkappa_j$, индекс j является наименьшим. Произведём усечение данной ветви, сохранив её начальный отрезок до вершины ξ_j. Производя эту операцию усечения для каждой ветви, получим *усечённое дерево*. Для случая функции конечного веса r на каждой ветви происходит повторение номеров вершин и индекс j, определяющий усечение, удовлетворяет неравенству $j \leqslant r$. Поэтому для этих функций усечённое дерево будет конечным.

Легко видеть, что усечённое дерево с занумерованными дугами и вершинами позволяет полностью восстановить исходное дерево.

Детерминированная функция называется *ограниченно-детерминированной* (или о.-д. функцией), если она имеет конечный вес.

Класс всех о.-д. функций обозначим через $P_{\text{од}}$.

В случае о.-д. функций полное (бесконечное) занумерованное дерево можно всегда свести к конечному дереву с занумерованными дугами и вершинами. Если в этом усечённом дереве произвести отождествление вершин с одинаковыми номерами, то получим так называемую *диаграмму переходов*, или, как её ещё называют, *диаграмму Мура*.

Для о.-д. функций $f(X)$ веса r диаграмма переходов имеет r вершин; из каждой вершины исходит N $(N = 2^n)$ дуг, дугам приписаны пары $(0, \gamma'), (1, \gamma''), \ldots, (N-1, \gamma^{(N)})$, выделена начальная

вершина. Очевидно, по диаграмме переходов о.-д. функция восстанавливается однозначно.

Теорема 25. *Число $p(n,r)$ о.-д. функций из $P_{\text{од}}$, зависящих от переменных $x_1,\dots,x_n$ и имеющих вес r, не превосходит $(2r)^{r\cdot 2^n}$.*

Доказательство. Возьмём диаграмму Мура для о.-д. функции $f(x_1,\dots,x_n)$ веса r. В ней из каждой вершины исходит 2^n дуг, причём i-я дуга входит в одну из r вершин и ей приписана пара (i,γ), где $\gamma\in\{0,1\}$. Таким образом, $p(n,r)$ не превосходит числа диаграмм Мура вышеуказанного вида. Данные диаграммы могут быть получены следующим образом.

Возьмём r вершин, занумерованных числами $0,1,\dots,r-1$ (0 — выделенная вершина); из каждой вершины исходит $N=2^n$ дуг, занумерованных числами $0,1,\dots,N-1$. Всего имеем rN дуг. Каждая дуга может входить в любую из r вершин, и ей может быть приписано любое из двух чисел (0 или 1). Поэтому $p(n,r)\leqslant(2r)^{rN}=(2r)^{r\cdot 2^n}$. Теорема доказана.

§ 2. Способы задания ограниченно-детерминированных функций

Пусть $f(X)=f(x_1,\dots,x_n)$ — произвольная о.-д. функция. Как уже отмечалось, областью определения этой функции является множество всех последовательностей вида $\{\alpha(1),\alpha(2),\dots,\alpha(t),\dots\}$, где $\alpha(t)$ для любого $t=1,2,\dots$ есть двоичная запись некоторого числа из множества $\{0,1,\dots,2^n-1\}$. Множеством значений о.-д. функции $f(X)$ является множество всех последовательностей вида $\{\gamma(1),\gamma(2),\dots,\gamma(t),\dots\}$, где $\gamma(t)\in\{0,1\}$ для любого $t=1,2,\dots$.

Используя то обстоятельство, что о.-д. функция $f(X)$ является математической моделью функционирования реального дискретного преобразователя (автомата), последовательность $\{\alpha(1),\alpha(2),\dots,\alpha(t),\dots\}$ будем называть входной последовательностью, а последовательность $\{\gamma(1),\gamma(2),\dots,\gamma(t),\dots\}$ — выходной последовательностью автомата (реализующего $f(X)$); числа $\alpha(t)$ и $\gamma(t)$ будем называть *сигналами*, поступающими в момент времени t соответственно на входы и на выход рассматриваемого автомата, а вершины диаграммы Мура для функции $f(X)$ — *состояниями* автомата.

Рассмотрим диаграмму Мура для о.-д. функции $f(X)$, реализуемой автоматом A. Предположим, что в момент времени $t-1$ автомат находился в состоянии $\varkappa(t-1)$; тогда при по-

ступлении в момент времени t на входы автомата сигнала $\alpha(t)$ автомат перейдёт по дуге $\alpha(t)$, выходящей из вершины $\varkappa(t-1)$ в диаграмме Мура, в состояние $\varkappa(t)$ и выдаст на выходе сигнал $\gamma(t)$ (рис. 19). Таким образом, пара $(\gamma(t), \varkappa(t))$ однозначно определяется парой $(\alpha(t), \varkappa(t-1))$.

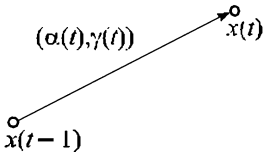


Рис. 19

Пусть в момент времени t переменная $X(t)$ описывает значение входного сигнала $\alpha(t)$, переменная $Q(t)$ описывает значения состояния $\varkappa(t)$, переменная $Z(t)$ описывает значение выходного сигнала $\gamma(t)$. Мы приходим к следующим уравнениям:

$$Z(t) = F(X(t), Q(t-1)),$$
$$Q(t) = G(X(t), Q(t-1)),$$

где F, G — некоторые функции, а $Q(0) = 0$.

Данные уравнения называются *каноническими уравнениями*. Можно перейти от векторной записи канонических уравнений к скалярной. Пусть $l = \lceil \log r \rceil$; тогда

$$\begin{cases} z(t) = & F'(x_1(t), \dots, x_n(t), q_1(t-1), \dots, q_l(t-1)), \\ q_1(t) = & G_1(x_1(t), \dots, x_n(t), q_1(t-1), \dots, q_l(t-1)), \\ & \dots \\ q_l(t) = & G_l(x_1(t), \dots, x_n(t), q_1(t-1), \dots, q_l(t-1)), \\ q_1(0) = & \dots = q_l(0) = 0. \end{cases}$$

Здесь $F', G_1, \dots, G_l$ — булевы функции, определённые на множестве $\mathcal{E}$ из $r \cdot 2^n$ наборов длины $n + l$, поскольку переменные $x_1, \dots, x_n$ пробегают значения из E_2, а вектор (двоичное число) $(q_1, \dots, q_l)$ принимает r значений (например, двоичные записи чисел $0, 1, \dots, r-1$). Доопределив функции $F', G_1, \dots, G_l$ на всём множестве булевых наборов длины $n + l$, мы получим соответственно функции $\phi, g_1, \dots, g_l$ из P_2, и канонические уравнения примут вид

$$\begin{cases} z(t) = & \phi(x_1(t), \dots, x_n(t), q_1(t-1), \dots, q_l(t-1)), \\ q_1(t) = & g_1(x_1(t), \dots, x_n(t), q_1(t-1), \dots, q_l(t-1)), \\ & \dots \\ q_l(t) = & g_l(x_1(t), \dots, x_n(t), q_1(t-1), \dots, q_l(t-1)), \\ q_1(0) = & \dots = q_l(0) = 0. \end{cases} \tag{1}$$

Доопределение частичных (недоопределённых) булевых функций $F', G_1, \dots, G_l$ до (полностью определённых) булевых

функций $\phi, g_1, \ldots, g_l$ обычно осуществляется так, чтобы правые части в системе (1) канонических уравнений имели наиболее простой вид, т. е. чтобы сложность реализации булевых функций $\phi, g_1, \ldots, g_l$ была по возможности наименьшей.

Поскольку для булевых функций имеется возможность табличного представления, то и о.-д. функции допускают табличное задание с помощью так называемых таблиц переходов. *Таблица переходов* для о.-д. функции $f(X)$, задаваемой каноническими уравнениями (1), — это табл. 13. В левых $(n+l)$ столбцах указываются всевозможные наборы значений переменных $x_1(t), \ldots, x_n(t)$, $q_1(t-1), \ldots, q_l(t-1)$, а в правых $(l+1)$ столбцах — соответствующие значения $z(t), q_1(t), \ldots, q_l(t)$ булевых функций $\phi, g_1, \ldots, g_l$.

Таблица 13

$x_1(t)$	...	$x_n(t)$	$q_1(t-1)$	...	$q_l(t-1)$	$z(t)$	$q_1(t)$	...	$q_l(t)$
0	...	0	0	...	0	$\phi(0, \ldots, 0)$	$g_1(0, \ldots, 0)$	...	$g_l(0, \ldots, 0)$
...	...	...	...	...	...	...	...	...	...
1	...	1	1	...	1	$\phi(1, \ldots, 1)$	$g_1(1, \ldots, 1)$	...	$g_l(1, \ldots, 1)$

§ 3. Схемы автоматов из функциональных элементов и элементов задержки

При построении схем автоматов (либо самих автоматов) предполагается возможным использование функциональных элементов и так называемых элементов задержки.

Элементом единичной задержки (или просто элементом задержки) называется автомат с одним входом и одним выходом (рис. 20), который реализует о.-д. функцию, задаваемую системой канонических уравнений

$$\begin{cases} z(t) = q(t-1), \\ q(t) = x(t), \\ q(0) = 0. \end{cases}$$

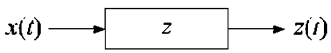


Рис. 20

Из канонических уравнений видно, что элемент задержки представляет собой автомат с двумя состояниями, в котором сигнал, поданный на вход в момент времени t, появляется на выходе в следующий момент времени $t+1$ $(t=1,2,\ldots)$.

Возьмём произвольную о.-д. функцию $f(X)$, задаваемую системой канонических уравнений (1) из § 2. Построим вначале схему из функциональных элементов (конъюнкторов, дизъюнкторов и инверторов), имеющую $(n+l)$ входов, $(l+1)$ выходов и реализующую булевы функции $\phi(x_1(t),\ldots$ $\ldots, x_n(t), q_1(t-1),\ldots, q_l(t-1)), g_1(x_1(t),\ldots, x_n(t), q_1(t-1),\ldots$ $\ldots, q_l(t-1)),\ldots, g_l(x_1(t),\ldots, x_n(t), q_1(t-1),\ldots, q_l(t-1))$ из правых частей системы. На первые n входов этой схемы подаются переменные $x_1(t),\ldots, x_n(t)$, на последние l входов подаются $q_1(t-1),\ldots, q_l(t-1)$ (рис. 21).

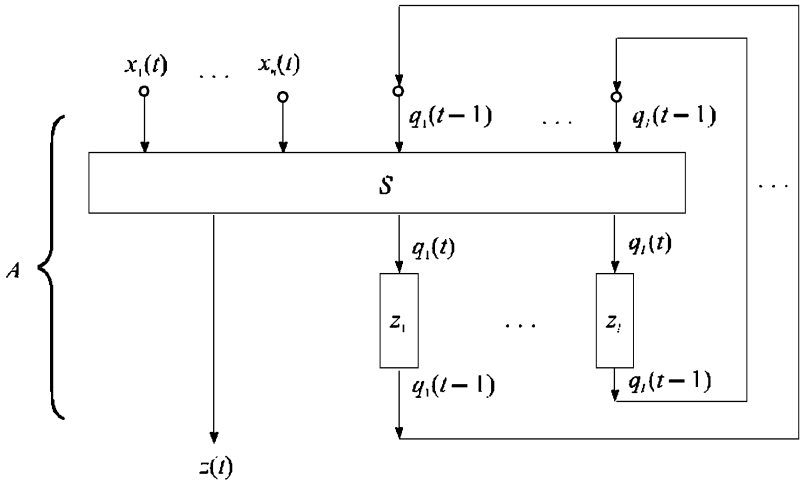


Рис. 21

Далее добавим в схему l элементов задержки $z_1,\ldots, z_l$, соединив их входы с соответствующими выходами схемы S, на которых реализуются $q_1(t),\ldots, q_l(t)$. Выходы элементов задержки, на которых реализуются $q_1(t-1),\ldots, q_l(t-1)$, соединим с соответствующими входами схемы S. В результате и получим схему автомата A, реализующего о.-д. функцию $f(X)$; на (внешние) входы этого автомата подаются $x_1(t),\ldots, x_n(t)$, на выходе получаем $z(t)$ — значение о.-д. функции $f(X)$.

Глава VIII

АЛГОРИТМЫ

§ 1. Алгоритмы. Машины Тьюринга. Задание машины системой команд

Для математики типичной является ситуация, когда требуется найти достаточно эффективный способ решения для бесконечного множества однотипных задач. Такими задачами являются, например:

а) возведение в квадрат целого числа;

б) извлечение квадратного корня из числа с заданной точностью;

в) решение системы линейных уравнений;

г) выяснение тождественной истинности формул для функций алгебры логики.

Для этих и многих других задач известны эффективные алгоритмы решения, которые, как оказалось, имеют следующие важные общие черты.

1) *Элементарность* шагов алгоритма: решение задачи распадается на ряд этапов, число которых может быть большим, но каждый из них достаточно прост.

2) *Детерминированность*: после завершения очередного этапа однозначно определено, что́ нужно делать на следующем этапе.

3) *Массовость*: алгоритм должен быть пригодным для решения всех задач заданного типа.

В общем случае, не претендуя на математическую строгость определения, понятие «алгоритм» можно ввести (или, скорее, пояснить) так: алгоритм — это точное предписание, которое задаёт вычислительный процесс (называемый в этом случае алгоритмическим), начинающийся с произвольного исходного данного (или из некоторой совокупности возможных исходных данных) и направленный на получение результата, полностью определённого этим исходным данным.

Если подходящий алгоритм решения задачи удаётся найти, то этим обычно дело и заканчивается.

Однако выяснилось, что для каких-то задач подходящие эффективные алгоритмы найти никак не удаётся. В разряд «алгоритмически неразрешимых» попали, например, задача установления тождественной истинности формул исчисления предикатов, задача о разрешимости в целых числах системы диофантовых уравнений (алгебраических уравнений с целочисленными коэффициентами) и ряд других задач. Следствием безуспешных поисков алгоритмов решения этих задач явилась гипотеза о том, что таких алгоритмов нет. Но доказать эту гипотезу, используя лишь некоторое, скорее интуитивное, нежели формальное, понятие алгоритма, оказалось невозможно. Поэтому в тридцатые годы прошлого столетия разными математиками были предложены несколько вариантов формального определения алгоритма. В дальнейшем выяснилось, что все эти определения по существу эквивалентны. Последнее обстоятельство позволяет надеяться, что главные моменты из интуитивного представления об эффективном алгоритме в предложенных формальных определениях учтены и отражены правильно.

Одно из формальных определений алгоритма связано с использованием математической модели вычислительного устройства, которая носит название машины Тьюринга (по имени предложившего данную модель английского математика А. Тьюринга).

Каждая *машина Тьюринга* имеет *ленту*, *читающую и пишущую головку* и работает с некоторым конечным (внешним) *алфавитом* $A = \{a_0, a_1, \ldots, a_r\}$. Лента бесконечна в обе стороны и разбита на *клетки* (рис. 22). Машина работает в дискретные моменты времени ($t = 1, 2, 3, \ldots$).

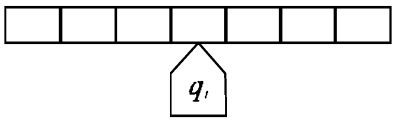


Рис. 22

Среди букв алфавита A имеется *пустой* символ a_0 (его будем обозначать также через 0). В каждый момент времени в каждой клетке записана одна из букв алфавита A.

Читающая и пишущая головка имеет конечное множество *внутренних состояний* $Q = \{q_0, q_1, \ldots, q_n\}$ (которые будем считать также состояниями самой машины). В каждый момент времени она находится в одном из этих состояний и обозревает одну из клеток ленты. В зависимости от своего состояния и буквы на ленте головка записывает в обозреваемую клетку новую букву (возможно, ту же самую), переходит к следующему моменту вре-

мени в новое состояние (в частности, то же самое) и сдвигается по ленте — либо на одну клетку влево, либо на одну клетку вправо, либо остаётся на месте. Среди внутренних состояний выделено одно, скажем, q_0, которое называется *заключительным*. Если после некоторого очередного сдвига головка попадает в заключительное состояние, то машина прекращает свою работу (останавливается).

Чтобы полностью определить работу машины, достаточно указать для начального момента времени её *конфигурацию*, в которую входят:

1) распределение букв по клеткам ленты;

2) клетка, обозреваемая головкой;

3) состояние головки.

Обычно рассматриваются конфигурации, в которых на ленте записано конечное число непустых символов. В этом случае можно считать, что на ленте записано слово конечной длины в алфавите A, содержащее все записанные на ленте непустые символы, причём в этом слове самая левая и самая правая буквы — непустые (внутри слова могут встречаться и пустые символы; наличие пустого символа в клетке содержательно означает, что в этой клетке ничего не записано).

В качестве начального состояния будем предполагать, скажем (как это обычно и делается), состояние q_1.

Далее будем предполагать, что в начальной конфигурации головка обозревает крайнюю правую клетку ленты с непустым символом.

Если машина, работая в соответствии с указанными правилами, в некоторый момент времени придёт в заключительное состояние, то она считается *применимой* к начальной конфигурации (или к начальному слову, записанному на ленте). Результатом работы машины при этом считается слово, которое окажется записанным на ленте в заключительной конфигурации.

Если же при работе машины ни в какой момент времени её головка (а вместе с ней и сама машина) не окажется в заключительном состоянии, то машина считается *неприменимой* к начальной конфигурации (и к начальному слову); результат её работы в этом случае не определён.

Поскольку каждая машина Тьюринга имеет конечный алфавит и конечное число состояний, то она полностью определяется конечным числом *команд* вида

$$qa \to q'a'S,$$

где:

q — состояние головки;

a — буква в обозреваемой головкой клетке;

q' — состояние головки в следующий момент;

a' — буква, записываемая вместо a в обозреваемую клетку;

S — сдвиг головки к следующему моменту:

Л — если сдвиг происходит влево,

П — если сдвиг происходит вправо,

С — если головка остается на месте.

При вычислениях числовых функций на машинах Тьюринга часто используется кодирование чисел, при котором целое неотрицательное число m изображается строчкой из $(m+1)$ единиц (палочек). Например, машина M, прибавляющая к произвольному числу единицу, может быть определена так.

Алфавит: $\{0, 1\}$.

Система команд:

$$q_1 1 \to q_1 1 \text{П},$$
$$q_1 0 \to q_0 1 \text{С}.$$

Работа данной машины при прибавлении единицы к числу 2 выглядит следующим образом:

$1\ 1\ \underset{q_1}{1}$ (начальная конфигурация)

$1\ 1\ 1\ \underset{q_1}{0}$

$1\ 1\ 1\ \underset{q_0}{1}$ (заключительная конфигурация).

Числовая функция $f(x_1, \ldots, x_r)$ называется *вычислимой* на машине Тьюринга, если существует машина Тьюринга M_f, применимая ко всем конфигурациям вида

$$\overbrace{1\,1\,1\,\ldots\,1}^{m_1+1}\,0\,\overbrace{1\,1\,\ldots\,1}^{m_2+1}\,0\,\ldots\,0\,\overbrace{1\,1\,\ldots\,\underset{q_1}{1}}^{m_r+1}$$

и приводящая к заключительной конфигурации

$$0\,\overbrace{1\,1\,\ldots\,\underset{q_0}{1}}^{m+1}\,0,$$

где $m = f(m_1, m_2, \ldots, m_r)$.

Нетрудно убедиться, что арифметические операции вычислимы в этом смысле. Возьмём, например, сложение двух неотрицательных чисел m_1 и m_2.

Начальная конфигурация:

$$0\overbrace{1\ldots1}^{m_1+1}0\overbrace{1\ldots\underset{q_1}{1}}^{m_2+1}0.$$

Заключительная конфигурация:

$$0\overbrace{1\ldots\ldots\underset{q_0}{1}}^{m_1+m_2+1}0.$$

Система команд:

$q_1 1 \to q_2 0\,Л,$
$q_2 0 \to q_0 0\,Л,$ (команда для случая, когда $m_2 = 0$)
$q_2 1 \to q_3 0\,Л,$ (убирается единица, которая будет перенесена на место «разделительного» нуля)
$q_3 1 \to q_3 1\,Л,$
$q_3 0 \to q_4 1\,П,$ («разделительный» нуль заменяется единицей)
$q_4 1 \to q_4 1\,П,$
$q_4 0 \to q_0 0\,Л.$

Нетрудно убедиться, что представленная машина действительно осуществляет арифметическое сложение любых двух неотрицательных целых чисел.

§ 2. Композиции машин. Тезис Тьюринга

Пусть M_1 и M_2 — машины Тьюринга в одном алфавите. Построим третью машину Тьюринга, работа которой будет равносильна последовательной работе машин M_1 и M_2 (сначала M_1, а затем M_2). Предположим, что машина M_1 имеет состояния q_0 (заключительное состояние), q_1 (начальное состояние), $q_2, \ldots, q_n$, а машина M_2 имеет состояния q_0 (заключительное), q_1 (начальное), $q_2, \ldots, q_m$. Объявим заключительное состояние машины M_1 начальным состоянием машины M_2. Более точно, определим машину M следующим образом. Пусть она имеет состояния $q_0, q_1, \ldots, q_{n+m}$, а её команды получаются из команд машин M_1 и M_2 так:

— в командах машины M_1 всюду заменим q_0 на q_{n+1};

— в командах машины M_2 всюду заменим q_i $(i \neq 0)$ на q_{n+i}.

Таким способом, в частности (при одинаковых машинах M_1 и M_2), можно построить машину, работа которой равносильна

двукратной (а в более общем случае и k-кратной) работе машины M.

Можно использовать машины с несколькими заключительными состояниями и строить более сложные композиции машин. Например, можно взять три машины M_0, M_1, M_2, где M_0 имеет два заключительных состояния $q_{0,1}$ и $q_{0,2}$, и построить машину M, работа которой будет равносильна работе машин M_0, M_1, M_2 в следующем смысле: если машина M_0 приходит в заключительное состояние $q_{0,1}$, то затем работает M_1, а если M_0 приходит в состояние $q_{0,2}$, то далее работает M_2.

Общепринятым является следующее предположение.

Тезис Тьюринга. *Любой интуитивно осуществимый алгоритм может быть реализован некоторой машиной Тьюринга.*

Это предположение невозможно доказать из-за того, что понятие интуитивно осуществимого алгоритма является неформальным. Вместе с тем это предположение подтверждается опытом математиков, занимающихся построением алгоритмов на протяжении тысячелетий. Еще один довод в пользу данного тезиса — это то, что предложенные разными авторами формальные уточнения понятия алгоритма оказались эквивалентными.

§ 3. Проблема самоприменимости. Теорема о самоприменимости

Опираясь на формальное уточнение понятия алгоритма по Тьюрингу, приведем пример алгоритмически неразрешимой задачи — проблемы самоприменимости, которая ставится следующим образом.

Будем рассматривать машины Тьюринга, внешний алфавит которых содержит буквы 1 и 0. Закодируем все машины Тьюринга словами в этом алфавите так, чтобы по системе команд эффективно (т. е. достаточно просто) строился код и наоборот, по коду эффективно восстанавливалась система команд; это можно сделать, например, так.

Пусть машина M имеет алфавит $\{a_0, a_1, \ldots, a_r\}$ и состояния $q_0, q_1, \ldots, q_n$. Все объекты (кроме стрелок), встречающиеся в командах машины, занумеруем целыми неотрицательными числами:

$$\begin{array}{cccccccccc} \text{С} & \text{Л} & \text{П} & a_0 & a_1 & \ldots & a_i & \ldots & a_r \\ 0 & 1 & 2 & 3 & 4 & \ldots & i+3 & \ldots & r+3 \end{array}$$

$$\begin{array}{ccccccc} q_0 & q_1 & \cdots & q_j & \cdots & q_n \\ r+4 & r+5 & \ldots & r+j+4 & \ldots & r+n+4. \end{array}$$

Как и выше, число i будем изображать набором из $(i+1)$ единиц.

Пусть $b \in \{\text{С}, \text{Л}, \text{П}, a_0, a_1, \ldots, a_r, q_0, q_1, \ldots, q_n\}$. Обозначим через $K(b)$ набор из одних единиц, длина которого соответствует номеру элемента b при указанной выше нумерации. Например, $K(\text{П}) = 111$; $K(a_4) = 11111111$.

Команде $qa \to q'a'S$ сопоставим слово

$$K(q)\,0\,K(a)\,0\,K(q')\,0\,K(a')\,0\,K(S),$$

являющееся по определению кодом команды. Выпишем подряд коды всех команд машины M, разделяя их парами нулей:

$$000\_\_\_\_00\_\_\_\_ \ldots \_\_\_\_00\_\_\_\_000;$$

запись можно упорядочить, например, так: сначала выпишем команды, отвечающие парам (стоящим слева от стрелки) q_1a_0, q_1a_1, ..., q_1a_r, затем команды, отвечающие парам q_2a_0, q_2a_1, ..., q_2a_r, и т. д. Полученное слово будем называть кодом машины M и обозначать через $K(M)$. Из самого способа построения кода видно, что он эффективно определяется системой команд. Нетрудно также заметить, что по коду машины эффективно определяется ее система команд. Действительно, просматривая в кодах команд вторые компоненты, можно найти максимальный номер буквы a_r, после чего по коду $K(q)$ состояния q легко определяется номер этого состояния.

Машина M, применимая к слову $K(M)$, т. е. к собственному коду, называется *самоприменимой*. Машина, не применимая к собственному коду, называется *несамоприменимой*.

Каждая машина Тьюринга является либо самоприменимой, либо несамоприменимой. Существуют как самоприменимые, так и несамоприменимые машины. Самоприменимыми являются, например, машины, у которых в правых частях всех команд стоит только заключительное состояние; такие машины применимы ко всем словам (в соответствующем алфавите) и, в частности, к собственным кодам. Примерами несамоприменимых машин могут служить машины, у которых в правых частях всех команд не встречается заключительное состояние; такие машины не применимы ни к каким словам, в том числе и к собственным кодам.

Проблема самоприменимости состоит в том, чтобы по машине установить, является ли она самоприменимой или нет. Точнее говоря, проблема заключается в нахождении алгоритма для решения данной задачи. Поскольку принят тезис Тьюринга, то и алгоритм нужно искать в виде соответствующей машины Тьюринга, т. е. требуется построить машину Тьюринга, которая

была бы применима к кодам всех машин Тьюринга и при этом заключительные конфигурации при работе над кодами самоприменимых машин и несамоприменимых машин эффективно различались бы. Потребуем, чтобы после работы над кодом самоприменимой машины заключительная конфигурация имела вид

$$\ldots\ldots \underset{q_0}{1} \ldots.$$

(вне обозреваемой клетки может стоять всё, что угодно), а после работы над кодом несамоприменимой машины заключительная конфигурация имела вид

$$\ldots\ldots \underset{q_0}{0} \ldots..$$

Теорема 26 (о самоприменимости). *Проблема самоприменимости алгоритмически неразрешима* (т. е. не существует машины Тьюринга, решающей эту проблему в указанном выше смысле).

Доказательство. Предположим, что такая машина M^*, решающая проблему самоприменимости, существует. В таком случае существует и машина M^{**}, обладающая следующими свойствами:

1) M^{**} применима к кодам несамоприменимых машин;

2) M^{**} неприменима к кодам самоприменимых машин.

Машина M^{**} получается из машины M^* следующим образом. Сохраняются все команды M^*; заключительное состояние q_0 машины M^* делается незаключительным («обычным») состоянием машины M^{**}; в качестве заключительного состояния машины M^{**} берётся новое состояние q_0'; добавляются две новые команды:

$$q_0 1 \to q_0 1 \mathrm{C};$$
$$q_0 0 \to q_0' 0 \mathrm{C}.$$

Очевидно, что так построенная машина M^{**} является либо самоприменимой, либо несамоприменимой.

В первом случае она применима к собственному коду, т. е. к коду самоприменимой машины, а это невозможно в силу свойства 2).

Во втором случае она неприменима к собственному коду, т. е. к коду несамоприменимой машины, а это невозможно в силу свойства 1).

Пришли к противоречию, которое возникло из допущения о существовании машины M^*, решающей проблему самоприменимости. Следовательно, такой машины M^* не существует. Теорема доказана.

Глава IX

КОДИРОВАНИЕ

§ 1. Алфавитное кодирование. Разделимые коды. Свойство префикса

Теория кодирования представляет собой один из разделов дискретной математики, в котором исследуются отображения конечных или счётных множеств объектов произвольной природы в множества последовательностей из чисел $0, 1, \ldots, q-1$, где q — некоторое натуральное число (в частности, $q = 2$). Такие отображения обычно называют кодированиями. Многие задачи теории кодирования сводятся к отысканию кодирования, обладающего определёнными свойствами и оптимального в некотором заранее заданном смысле. Критерий оптимальности кодирования обычно связывается с минимизацией длины образов объектов, тогда как требуемые свойства кодирования могут быть довольно разнообразными. В частности, такими свойствами могут быть существование однозначного обратного отображения (декодирования), возможность исправлять при декодировании различного рода ошибки, возможность простой реализации кодирования и декодирования и т. п. Указанные задачи теории кодирования и некоторые подходы к решению этих задач рассматриваются ниже.

Пусть задан алфавит сообщений $A = \{a_1, \ldots, a_r\}$, состоящий из конечного числа букв (символов). Конечная последовательность символов из A

$$\widetilde{a} = a_{i_1} a_{i_2} \ldots a_{i_n}$$

называется *словом* в алфавите A, а число n — *длиной* слова $\widetilde{a}$. Длину слова $\widetilde{a}$ будем обозначать через $l(\widetilde{a})$. Пусть $S = S(A)$ — множество всех непустых слов в алфавите A, а S' – некоторое подмножество множества S. Объект, порождающий слова из S', называется *источником сообщений*, а слова из S' — *сообщениями*.

Пусть задан (кодирующий) алфавит $B = \{b_1, \dots, b_q\}$. Через $\widetilde{b}$ обозначим слово в алфавите B и через $S(B)$ — множество всех непустых слов в алфавите B.

Пусть задано отображение F, которое каждому слову $\widetilde{a} \in S'$ однозначно ставит в соответствие слово $\widetilde{b} = F(\widetilde{a})$, $\widetilde{b} \in S(B)$. Слово $\widetilde{b}$ будем называть *кодом сообщения* $\widetilde{a}$ (а также *кодовым словом*), а переход от слова $\widetilde{a}$ к его коду, т. е. отображение F — *кодированием*.

Кодирование F называется *взаимно однозначным*, если каждый код сообщения является кодом ровно одного сообщения.

Пусть задано отображение Σ букв алфавита A в множество $S(B)$ вида

$$\Sigma : \begin{cases} a_1 \to \widetilde{b}_1, \\ a_2 \to \widetilde{b}_2, \\ \dots \\ a_r \to \widetilde{b}_r \end{cases}$$

($\widetilde{b}_1, \widetilde{b}_2, \dots, \widetilde{b}_r$ — некоторые слова в алфавите B). Указанное отображение называется *схемой* (кодирования). Эта схема Σ определяет *алфавитное кодирование* следующим образом: каждому слову $\widetilde{a} = a_{i_1} \dots a_{i_n}$ из $S'(A) = S(A)$ ставится в соответствие слово $\widetilde{b} = \widetilde{b}_{i_1} \dots \widetilde{b}_{i_n}$, называемое *кодом слова* $\widetilde{a}$. Слова $\widetilde{b}_1, \dots, \widetilde{b}_r$ называются *элементарными кодами*.

Множество $\{\widetilde{b}_1, \dots, \widetilde{b}_r\}$ элементарных кодов называется *алфавитным кодом* и обозначается через $\Sigma(A)$.

Алфавитное кодирование и соответствующий алфавитный код называются *разделимыми*, если из каждого равенства

$$\widetilde{b}_{i_1} \widetilde{b}_{i_2} \dots \widetilde{b}_{i_k} = \widetilde{b}_{j_1} \widetilde{b}_{j_2} \dots \widetilde{b}_{j_l}$$

(между словами кодирующего алфавита) следует, что $l = k$ и $j_t = i_t$, $t = 1, 2, \dots, k$.

Легко заметить, что алфавитное кодирование является взаимно однозначным тогда и только тогда, когда оно задаётся с помощью разделимого кода. Ясно также, что все слова разделимого кода различны и непусты.

Приведём простой достаточный признак взаимной однозначности алфавитного кодирования.

Пусть слово $\widetilde{b}$ имеет вид

$$\widetilde{b} = \widetilde{b}' \widetilde{b}''.$$

Тогда слово $\widetilde{b}'$ называется *началом*, или *префиксом слова* $\widetilde{b}$, а $\widetilde{b}''$ — *концом* слова $\widetilde{b}$. При этом пустое слово Λ и само слово $\widetilde{b}$ также считаются началами и концами слова $\widetilde{b}$. Все начала и концы слова $\widetilde{b}$, отличные от него самого, называются *собственными*.

Схема Σ *обладает свойством префикса*, если для любых i и j $(1 \leqslant i,\ j \leqslant r;\ i \neq j)$ слово $\widetilde{b}_i$ не является префиксом слова $\widetilde{b}_j$; соответствующий такой схеме код называется *префиксным*.

Теорема 27. *Если схема* Σ *обладает свойством префикса, то определяемое этой схемой алфавитное кодирование будет взаимно однозначным.*

Доказательство. Поскольку схема Σ обладает свойством префикса, то все элементарные коды в ней попарно различны, т. е. $\widetilde{b}_i \neq \widetilde{b}_j$ при $i = j$. Предположим, что некоторое слово $\widetilde{b}$ является кодом двух различных сообщений, т. е. допускает два разбиения на элементарные коды

$$\widetilde{b} = \widetilde{b}_{i_1} \ldots \widetilde{b}_{i_s},$$
$$\widetilde{b} = \widetilde{b}_{j_1} \ldots \widetilde{b}_{j_t}.$$

Пусть $\widetilde{b}_{i_1} = \widetilde{b}_{j_1}, \ldots, \widetilde{b}_{i_{m-1}} = \widetilde{b}_{j_{m-1}}$, $\widetilde{b}_{i_m} \neq \widetilde{b}_{j_m}$. В таком случае одно из слов $\widetilde{b}_{i_m}, \widetilde{b}_{j_m}$ является префиксом другого. Получаем противоречие. Теорема доказана.

Отметим, что условие теоремы 27 не является необходимым и существуют разделимые коды, не обладающие свойством префикса. Об этом свидетельствует, в частности, такой пример.

Пусть $A = \{a_1, a_2\}$, $B = \{b_1, b_2\}$. Рассмотрим схему

$$\Sigma : \begin{cases} a_1 \to b_1, \\ a_2 \to b_1 b_2. \end{cases}$$

Очевидно, схема Σ свойством префикса не обладает. Но тем не менее задаваемое этой схемой алфавитное кодирование является взаимно однозначным. Декодирование (т. е. нахождение по коду сообщения самого сообщения) осуществляется так. Код сообщения $\widetilde{b}$ разобьём на элементарные коды. Отметим, что непосредственно перед каждым вхождением буквы b_2 в слове $\widetilde{b}$ находится буква b_1. Это позволяет выделить все пары $(b_1 b_2)$. Оставшаяся часть слова $\widetilde{b}$ будет состоять из букв b_1. Если теперь каждую пару $(b_1 b_2)$ заменить на a_2, а каждую из оставшихся букв b_1 — на a_1, то получим слово $\widetilde{a}$, являющееся прообразом слова $\widetilde{b}$.

§ 2. Неравенство Крафта–Макмиллана

Пусть алфавитное кодирование задано схемой

$$\Sigma : \begin{cases} a_1 \to \widetilde{b}_1, \\ \dots \\ a_r \to \widetilde{b}_r. \end{cases}$$

Пусть алфавит B содержит q букв и $l_i = l(\widetilde{b}_i)$ — длина слова $\widetilde{b}_i$ $(i = 1, \dots, r)$.

Теорема 28 (неравенство Крафта–Макмиллана). *Если алфавитное кодирование со схемой Σ обладает свойством взаимной однозначности, то*

$$\sum_{i=1}^{r} \frac{1}{q^{l_i}} \leqslant 1. \tag{1}$$

Доказательство. Рассмотрим всевозможные слова в алфавите A, имеющие длину n. Все они могут быть порождены выражением

$$(a_1 + \dots + a_r)^n,$$

если перемножить скобки (например, слева) без применения закона коммутативности и рассматривать произведение $a_{i_1} a_{i_2} \dots a_{i_n}$ как запись слова в алфавите A. Здесь, очевидно, символ a_{i_1} будет принадлежать первой скобке, a_{i_2} — второй, ..., a_{i_n} — n-й скобке. Имеем

$$(a_1 + \dots + a_r)^n = \sum_{(i_1 i_2 \dots i_n)} a_{i_1} a_{i_2} \dots a_{i_n}.$$

Соответствующие этим словам коды получаются, если заменить символы $a_1, \dots, a_r$ элементарными кодами $\widetilde{b}_1, \dots, \widetilde{b}_r$, используя схему алфавитного кодирования. Получим

$$(\widetilde{b}_1 + \dots + \widetilde{b}_r)^n = \sum_{(i_1 i_2 \dots i_n)} \widetilde{b}_{i_1} \widetilde{b}_{i_2} \dots \widetilde{b}_{i_n}. \tag{2}$$

Равенству (2) поставим в соответствие равенство

$$\left(\frac{1}{q^{l_1}} + \dots + \frac{1}{q^{l_r}} \right)^n = \sum_{(i_1 \dots i_n)} \frac{1}{q^{l_{i_1} + \dots + l_{i_n}}}. \tag{3}$$

Очевидно, что здесь членам с одинаковыми знаменателями из правой суммы соответствуют в (2) слова $\widetilde{b}_{i_1} \widetilde{b}_{i_2} \dots \widetilde{b}_{i_n}$ одинаковой длины. Введём обозначения: $t = l_{i_1} + \dots + l_{i_n}$, $l = \max\limits_{1 \leqslant i \leqslant r} l_i$

и $\nu(n,t)$ — число слов $\widetilde{b}_{i_1}\widetilde{b}_{i_2}\dots\widetilde{b}_{i_n}$ из (2), имеющих длину t (если таких слов нет, то $\nu(n,t)=0$). Получим

$$\sum_{(i_1\dots i_n)}\frac{1}{q^{l_{i_1}+\dots+l_{i_n}}}=\sum_{t=1}^{nl}\frac{\nu(n,t)}{q^t}. \tag{4}$$

В силу взаимной однозначности алфавитного кодирования если $(i_1,\dots,i_n)\neq(j_1,\dots,j_n)$, т.е. $a_{i_1}\dots a_{i_n}\neq a_{j_1}\dots a_{j_n}$, то $\widetilde{b}_{i_1}\dots\widetilde{b}_{i_n}\neq\widetilde{b}_{j_1}\dots\widetilde{b}_{j_n}$; отсюда вытекает $\nu(n,t)\leqslant q^t$, и, следовательно,

$$\sum_{t=1}^{nl}\frac{\nu(n,t)}{q^t}\leqslant nl.$$

Из последнего неравенства и соотношений (3), (4) получаем

$$\sum_{i=1}^{r}\frac{1}{q^{l_i}}\leqslant\sqrt[n]{nl}.$$

Это неравенство справедливо при любом n. Поэтому оно справедливо и при переходе к пределу при $n\to\infty$. Окончательно имеем

$$\sum_{i=1}^{r}\frac{1}{q^{l_i}}\leqslant 1.$$

Теорема доказана.

Последнюю теорему дополняет

Теорема 29. *Если натуральные числа $l_1,\dots,l_r$ удовлетворяют неравенству* (1), *то существует алфавитное кодирование со схемой*

$$\Sigma':\begin{cases}a_1\to\widetilde{b}'_1,\\ \dots\\ a_r\to\widetilde{b}'_r,\end{cases}$$

обладающей свойством префикса и удовлетворяющей равенствам

$$l(\widetilde{b}'_1)=l_1,\dots,l(\widetilde{b}'_r)=l_r.$$

Доказательство. Разобьём параметры $l_1,\dots,l_r$ на классы так, что l_i и l_j принадлежат одному классу тогда и только тогда, когда $l_i=l_j$. Пусть m $(1\leqslant m\leqslant r)$ — число классов, а символами $\lambda_1,\dots,\lambda_m$ и $\nu_1,\dots,\nu_m$ обозначены соответственно представители и мощности классов. Можно считать, что

$$\lambda_1<\lambda_2<\dots<\lambda_m.$$

Неравенство Крафта–Макмиллана можно переписать в виде

$$\sum_{i=1}^{m} \frac{\nu_i}{q^{\lambda_i}} \leqslant 1.$$

Это неравенство порождает серию вспомогательных неравенств

$$\frac{\nu_1}{q^{\lambda_1}} \leqslant 1, \text{ или } \nu_1 \leqslant q^{\lambda_1},$$

$$\frac{\nu_1}{q^{\lambda_1}} + \frac{\nu_2}{q^{\lambda_2}} \leqslant 1, \text{ или } \nu_2 \leqslant q^{\lambda_2} - \nu_1 q^{\lambda_2 - \lambda_1},$$

$$\dots$$

$$\frac{\nu_1}{q^{\lambda_1}} + \frac{\nu_2}{q^{\lambda_2}} + \dots + \frac{\nu_m}{q^{\lambda_m}} \leqslant 1, \text{ или}$$

$$\nu_m \leqslant q^{\lambda_m} - \nu_1 q^{\lambda_m - \lambda_1} - \nu_2 q^{\lambda_m - \lambda_2} - \dots - \nu_{m-1} q^{\lambda_m - \lambda_{m-1}}.$$

Рассмотрим слова в алфавите B, имеющие длину λ_1. Так как $\nu_1 \leqslant q^{\lambda_1}$, то можно выбрать из них ν_1 слов длины λ_1. Обозначим их через $\widetilde{b}'_1, \dots, \widetilde{b}'_{\nu_1}$. Исключим из дальнейшего рассмотрения слова, начинающиеся с $\widetilde{b}'_1 \dots \widetilde{b}'_{\nu_1}$. Далее, возьмём множество слов в алфавите B, имеющих длину λ_2, которые не начинаются со слов $\widetilde{b}'_1, \dots, \widetilde{b}'_{\nu_1}$. Очевидно, что таких слов будет $q^{\lambda_2} - \nu_1 q^{\lambda_2 - \lambda_1}$. Так как $\nu_2 \leqslant q^{\lambda_2} - \nu_1 q^{\lambda_2 - \lambda_1}$, то из этого множества можно выбрать ν_2 слов — обозначим их через $\widetilde{b}'_{\nu_1+1}, \dots, \widetilde{b}'_{\nu_1+\nu_2}$. Исключим из дальнейшего рассмотрения также слова, начинающиеся с $\widetilde{b}'_{\nu_1+1}, \dots, \widetilde{b}'_{\nu_1+\nu_2}$, и т. д. Используя последовательно вспомогательные неравенства, в итоге построим слова $\widetilde{b}'_1, \dots, \widetilde{b}'_r$ ($r = \sum_{i=1}^{m} \nu_i$). Если их взять в качестве элементарных кодов, то получим искомую схему Σ'. Для Σ' выполнено свойство префикса и

$$l(\widetilde{b}'_1) = l_1, \dots, l(\widetilde{b}'_r) = l_r.$$

Теорема доказана.

Следствие 1. *Неравенство Крафта–Макмиллана является необходимым и достаточным условием существования разделимого кода с длинами элементарных кодов* $l_1, \dots, l_r$.

Это следствие с очевидностью вытекает из теорем 28 и 29 (достаточно лишь вспомнить, что префиксный код — это разделимый код).

Следствие 2. *Если существует разделимый код с заданными длинами элементарных кодов, то существует и префиксный код с теми же длинами элементарных кодов.*

Для доказательства следует применить сначала теорему 28, а затем теорему 29.

§ 3. Коды с минимальной избыточностью. Оптимальное кодирование Хаффмена

Предположим, что задан алфавит $A = \{a_1, \dots, a_r\}$ $(r \geqslant 2)$ и набор положительных вероятностей $p_1, \dots, p_r$ $(\sum\limits_{i=1}^{r} p_i = 1)$ появлений символов $a_1, \dots, a_r$ в сообщениях (предполагается, что символы $a_1, \dots, a_r$ появляются в сообщениях независимо). Пусть далее задан алфавит $B = \{b_1, \dots, b_q\}$ $(q \geqslant 2)$. Тогда можно построить целый ряд схем алфавитного кодирования вида

$$\Sigma : \begin{cases} a_1 \to \widetilde{b}_1, \\ \dots \\ a_r \to \widetilde{b}_r, \end{cases}$$

обладающих свойством взаимной однозначности. В частности, всегда можно взять в качестве элементарных кодов $\widetilde{b}_1, \dots, \widetilde{b}_r$ различные слова в алфавите B одинаковой длины l, где $l = \lceil \log_q r \rceil$.

Для каждой схемы Σ можно ввести среднюю длину $l_{\text{ср}}^{\Sigma}$, определяемую как математическое ожидание длины элементарного кода, т. е.

$$l_{\text{ср}}^{\Sigma} = \sum_{i=1}^{r} p_i l_i, \quad l_i = l(\widetilde{b}_i).$$

Очевидно, что величина $l_{\text{ср}}^{\Sigma}$ $(l_{\text{ср}}^{\Sigma} \geqslant 1)$ показывает, во сколько раз увеличивается средняя длина слова при кодировании со схемой Σ.

Пусть

$$l^* = \inf_{\Sigma} l_{\text{ср}}^{\Sigma},$$

где нижняя грань берётся по всем схемам Σ, обеспечивающим свойство взаимной однозначности. Легко заметить, что

$$1 \leqslant l^* \leqslant \lceil \log_q r \rceil$$

(верхнюю оценку даёт, например, упомянутое выше кодирование). Отсюда следует, что при построении кодов, у которых

величина $l_{\text{ср}}^{\Sigma}$ близка к l^*, можно не рассматривать коды с $l_{\text{ср}}^{\Sigma}$, бóльшим, чем $\lceil \log_q r \rceil$. Значит, для таких схем

$$p_i l_i \leqslant \lceil \log_q r \rceil.$$

Положим $p^* = \min\limits_{1 \leqslant i \leqslant r} p_i$. Поскольку $p_i > 0$ $(i = 1, \ldots, r)$, то $p^* > 0$ и

$$l_i \leqslant \frac{\lceil \log_q r \rceil}{p^*}$$

для всех i. Значит, имеется конечное число возможных значений $l_{\text{ср}}^{\Sigma}$, для которых $l^* \leqslant l_{\text{ср}}^{\Sigma} \leqslant \lceil \log_q r \rceil$. Следовательно, величина l^* достигается на некоторой схеме Σ^* и может быть определена как

$$l^* = \min_{\Sigma} l_{\text{ср}}^{\Sigma}.$$

Коды, определяемые схемой Σ с $l_{\text{ср}}^{\Sigma} = l^*$, называются *кодами с минимальной избыточностью*, или *оптимальными кодами Хаффмена*.

Очевидно, что коды с минимальной избыточностью дают в среднем минимальное увеличение длины слов при соответствующем кодировании. Ввиду этого представляет интерес задача о построении кодов с минимальной избыточностью. Ниже эта задача решается для двоичного кодирования, когда кодирующий алфавит B содержит два символа (скажем, 0 и 1); похожим способом задача решается и в общем случае (когда $B = \{b_1, \ldots, b_q\}$, $q \geqslant 3$).

В силу следствия 2 (из теорем 28 и 29) существует алфавитное кодирование со свойством префикса, дающее коды с минимальной избыточностью. Поэтому при построении кодов с минимальной избыточностью можно ограничиться рассмотрением префиксных кодов.

Префиксный код удобно задавать с помощью занумерованного ориентированного дерева с корнем, в котором из каждой вершины выходит не более двух дуг и в каждую вершину, кроме корня, входит по одной дуге (в корень не входит ни одна дуга). Каждой дуге дерева приписывается 0 или 1; двум дугам, выходящим из одной вершины, приписываются разные символы (0 и 1). Такие деревья будем называть *кодовыми*. Каждой вершине v кодового дерева соответствует единственный путь $P(v)$, ведущий в неё из корня дерева; вершине v сопоставляется слово из символов, приписанных дугам, составляющим путь $P(v)$, в порядке следования этих дуг. Нетрудно заметить, что если двум вершинам v_1 и v_2 приписаны соответственно двоичные слова

(наборы) $\widetilde{b}$ и $\widetilde{b}'$, причем слово $\widetilde{b}$ является префиксом слова $\widetilde{b}'$, то путь из корня дерева в вершину v_2 проходит через v_1. Отсюда следует, что слова, сопоставленные концевым вершинам кодового дерева, образуют префиксный код (на рис. 23,*а* это код 00, 01, 101, 1110, 1111) и, наоборот, всякому префиксному коду соответствует множество концевых вершин некоторого кодового дерева.

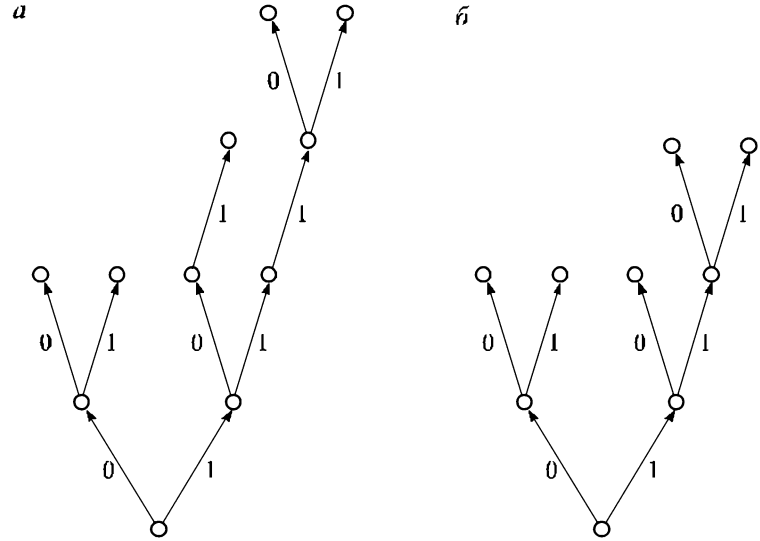


Рис. 23

Предположим, что из некоторой вершины кодового дерева D выходит ровно одна дуга. Если начало и конец этой дуги отождествить, а саму дугу удалить, то, очевидно, получим кодовое дерево D', которое, так же, как и исходное дерево D, задаёт префиксный код (с прежним алфавитом сообщений A), но в этом коде некоторые элементарные коды окажутся короче соответствующих элементарных кодов, определяемых исходным деревом D. Указанный переход от дерева D к дереву D' назовём операцией усечения; если, например, дважды применить операцию усечения к дереву на рис. 23,*а*, то получим кодовое дерево, изображённое на рис. 23,*б*.

Укажем некоторые свойства оптимальных префиксных кодов. Будем считать, что символы $a_1, a_2, \ldots, a_r$ алфавита сообщений A занумерованы в порядке убывания вероятностей их появления, т. е.

$$p_1 \geqslant p_2 \geqslant \ldots \geqslant p_r.$$

Свойство 1. *В оптимальном коде длины l_i элементарных кодов $\widetilde{b}_i$ не убывают:*

$$l_1 \leqslant l_2 \leqslant \ldots \leqslant l_r.$$

Действительно, если это не так, т. е. для некоторых i и j одновременно $p_i > p_j$ и $l_i > l_j$, то возьмём схему кодирования Σ', в которой символу a_i соответствует элементарный код $\widetilde{b}_j$, а символу a_j – элементарный код $\widetilde{b}_i$ (т. е. в исходной схеме кодирования Σ поменяем местами элементарные коды $\widetilde{b}_i$ и $\widetilde{b}_j$). В результате получим код с меньшей величиной $l_{\text{ср}}$, ибо

$$p_i l_i + p_j l_j = p_i l_j + p_j l_i + (p_i - p_j)(l_i - l_j) > p_i l_j + p_j l_i.$$

Но это противоречит предположению об оптимальности исходного кода. Свойство 1 установлено.

Свойство 2. *Существует оптимальный код, в котором двум наименее вероятным символам a_{r-1} и a_r соответствуют элементарные коды одинаковой длины, различающиеся лишь в последнем разряде.*

Рассмотрим произвольный оптимальный префиксный код и соответствующее ему кодовое дерево. Пусть $\widetilde{b}$ — вершина дерева, предшествующая концевой вершине (элементарному коду) $\widetilde{b}_r$ (рис. 24). Если из $\widetilde{b}$ выходит лишь одна дуга (входящая в $\widetilde{b}_r$), то к кодовому дереву (соответственно и к коду) можно применить операцию усечения, в результате чего дуга будет удалена, а вершина $\widetilde{b}_r$ отождествлена с $\widetilde{b}$. Поскольку операция усечения уменьшает $l_{\text{ср}}$, это противоречит оптимальности исходного кода. Значит, из вершины $\widetilde{b}$ выходят две дуги, и существует элементарный код $\widetilde{b}_i$, имеющий ту же длину l_r, что и $\widetilde{b}_r$. Отсюда и из того, что по свойству 1 $l_i \leqslant l_{r-1} \leqslant l_r$, вытекает равенство $l_i = l_{r-1} = l_r$. Если $i \neq r-1$, то в исходной схеме кодирования поменяем местами элементарные коды $\widetilde{b}_i$ и $\widetilde{b}_{r-1}$. В итоге получим обладающий нужным свойством код, который, так же, как и исходный, будет оптимальным.

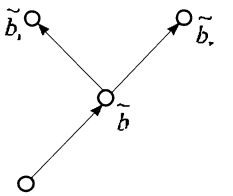


Рис. 24

Предположим теперь (как и выше), что источник сообщений S выдаёт сообщения в алфавите $A = \{a_1, \ldots, a_r\}$ и для вероятностей $p_1, \ldots, p_r$ появления символов $a_1, \ldots, a_r$ в сообщениях справедливо соотношение

$$p_1 \geqslant p_2 \geqslant \ldots \geqslant p_r > 0.$$

Рассмотрим источник S', полученный из S отождествлением («склеиванием») символов a_{r-1} и a_r. Он выдаёт символы $a_1, \dots, a_{r-2}$ с прежними вероятностями $p_1, \dots, p_{r-2}$, а новый символ a' («склеенный») — с вероятностью $p' = p_{r-1} + p_r$. Такой источник S' будем называть *редуцированным*. Предположим, что для редуцированного источника имеется некоторый код $\{\widetilde{b}_1, \widetilde{b}_2, \dots, \widetilde{b}_{r-2}, \widetilde{b}'\}$. По нему построим код для исходного источника S следующим образом. Для символов $a_1, \dots, a_{r-2}$ сохраним прежние коды $\widetilde{b}_1, \dots, \widetilde{b}_{r-2}$, а в качестве $\widetilde{b}_{r-1}$ и $\widetilde{b}_r$ возьмём соответственно слова $\widetilde{b}'0$ и $\widetilde{b}'1$. Очевидно, что если код для редуцированного источника S' — префиксный, то и построенный код для исходного источника – тоже префиксный.

Теорема 30 (о редукции). *Если префиксный код* $\{\widetilde{b}_1, \widetilde{b}_2, \dots, \widetilde{b}_{r-2}, \widetilde{b}'\}$ *для редуцированного источника* S' *является оптимальным, то и префиксный код* $\{\widetilde{b}_1, \widetilde{b}_2, \dots, \widetilde{b}_{r-2}, \widetilde{b}'0, \widetilde{b}'1\}$ *для исходного источника* S *также является оптимальным.*

Доказательство. Обозначим через $l'_{\text{ср}}$ и $l_{\text{ср}}$ соответственно средние длины кодов для редуцированного источника S' и для исходного источника S. По построению $l_{r-1} = l_r = l' + 1$, где l' — длина элементарного кода $\widetilde{b}'$, и $p' = p_{r-1} + p_r$, поэтому

$$l_{\text{ср}} = \sum_{i=1}^{r} p_i l_i = \sum_{i=1}^{r-2} p_i l_i + p_{r-1}(l' + 1) + p_r(l' + 1) = \\ = \sum_{i=1}^{r-2} p_i l_i + p' l' + p_{r-1} + p_r = l'_{\text{ср}} + p_{r-1} + p_r. \quad (1)$$

Если код для источника S не является оптимальным, то возьмём оптимальный код, обладающий свойством 2, и построим по нему код для редуцированного источника S', сопоставив символу a' вершину b', предшествующую вершинам $\widetilde{b}_{r-1}$ и $\widetilde{b}_r$, и сохранив для остальных символов $a_1, \dots, a_{r-2}$ те же элементарные коды $\widetilde{b}_1, \dots, \widetilde{b}_{r-2}$, что и в оптимальном коде для S. Средние длины $l_{\text{ср}}$ и $l'_{\text{ср}}$ для новых кодов удовлетворяют (1), а поскольку новая величина $l_{\text{ср}}$ (для оптимального кода) меньше соответствующей прежней, то и новая величина $l'_{\text{ср}}$ будет меньше прежней. А это противоречит оптимальности кода для исходного редуцированного источника. Теорема доказана.

Теорема 30 даёт алгоритм построения оптимальных кодов. В этом алгоритме сначала производится свёртывание искомо-

го кода в соответствии с правилами перехода от источника S к редуцированному источнику S'. В процессе свёртывания один за другим получаются всё более простые (короткие) коды и в конце концов — код из однобуквенных элементарных кодов. Фактически это означает преобразование набора вероятностей до тех пор, пока не получится набор вероятностей (из двух чисел), для которого оптимальный код находится тривиально. После этого найденный код развёртывается в соответствии с теоремой о редукции в последовательность оптимальных кодов, которая заканчивается искомым кодом.

Пример построения оптимального кода в случае $r = 6$ и набора вероятностей $P = (0{,}4; 0{,}25; 0{,}15; 0{,}1; 0{,}05; 0{,}05)$ представлен в табл. 14.

Таблица 14

A	P	P^I	P^{II}	P^{III}	P^{IV}	Σ^{IV}	Σ^{III}	Σ^{II}	Σ^I	Σ
a_1	0,4	0,4	0,4	0,4	0,6	0	1	1	1	1
a_2	0,25	0,25	0,25	0,35	0,4	1	00	01	01	01
a_3	0.15	0,15	0,2	0.25			01	000	001	001
a_4	0,1	0,1	0,15					001	0000	0000
a_5	0,05	0,1							0001	00010
a_6	0,05									00011

§ 4. Самокорректирующиеся коды. Коды Хэмминга

Самокорректирующиеся коды ниже рассматриваются для некоторого частного случая (равномерного) кодирования.

Пусть $A = \{0, 1\}$ — алфавит сообщений, содержащий два символа. Пусть, далее, $\{\widetilde{a}_1, \ldots, \widetilde{a}_s\}$ — множество всех слов вида $\widetilde{a} = \alpha_1\alpha_2 \ldots \alpha_m$ в алфавите A, имеющих фиксированную длину m ($s = 2^m$).

Предположим, что в канале связи, по которому передаются слова (сообщения), действует источник помех, который может вызывать ошибки не более чем в p символах одного слова (сообщения). Это значит, что двоичная последовательность, полученная на выходе канала связи, отличается от двоичной последовательности, поступившей на вход этого канала, не более чем в p позициях. Ясно, что если передавать исходное сообщение $\alpha_1 \ldots \alpha_m$ (без предварительного кодирования), то на выходе канала связи невозможно будет установить, какое сообщение фак-

тически было передано. В связи с этим возникает вопрос: нельзя ли осуществить кодирование слов $\widetilde{a}$ из множества $\{\widetilde{a}_1, \dots, \widetilde{a}_s\}$, т. е. слов вида $\alpha_1 \dots \alpha_m$, словами $\beta_1 \dots \beta_n$ длины n так, чтобы по коду $\beta'_1 \dots \beta'_n$, полученному на выходе канала при передаче кода $\beta_1 \dots \beta_n$, можно было однозначно восстановить этот код и, значит, исходное сообщение $\alpha_1 \dots \alpha_m$? Коды, обладающие данными свойствами, называют *самокорректирующимися кодами* относительно рассматриваемого источника помех, или *кодами, исправляющими p ошибок*.

Рассмотрим коды, исправляющие одну ошибку. Будем считать, что кодовые слова имеют длину n, т. е. представляют собой двоичные наборы (векторы), содержащие n разрядов (компонент).

Найдём такое натуральное число k, что

$$2^{k-1} \leqslant n < 2^k;$$

нетрудно заметить, что $k = \lfloor \log n \rfloor + 1 = \lceil \log(n+1) \rceil$. Для произвольного i из множества $\{0, 1, \dots, 2^k - 1\}$ через $\widetilde{e}^{(k)}(i)$ обозначим двоичный набор $(e_1, \dots, e_k)$ длины k, удовлетворяющий условию

$$i = \sum_{j=1}^{k} e_j 2^{k-j},$$

т. е. являющийся двоичной записью числа i с помощью k цифр (например, $\widetilde{e}^{(6)}(35) = (100011)$, $\widetilde{e}^{(8)}(35) = (00100011)$; здесь и ниже, где это удобно, запятые в изображении булевых наборов мы опускаем). Сложение чисел по модулю 2 будем обозначать знаками $\oplus$ и $\sum$. Пусть B^n – множество всех двоичных слов (наборов) длины n в алфавите $\{0, 1\}$. Суммой наборов $\widetilde{\alpha} = (\alpha_1, \dots, \alpha_n)$ и $\widetilde{\beta} = (\beta_1, \dots, \beta_n)$ из B^n будем считать набор $(\alpha_1 \oplus \beta_1, \alpha_2 \oplus \beta_2, \dots, \alpha_n \oplus \beta_n)$; для обозначения операции сложения наборов также будем использовать знаки $\oplus$ и $\sum$. Умножение произвольного набора $\widetilde{\beta} = (\beta_1, \dots, \beta_n)$ из B^n на элемент α из $\{0, 1\}$ введём следующим образом: $\alpha\widetilde{\beta} = (\alpha\beta_1, \alpha\beta_2, \dots, \alpha\beta_n)$.

На множестве B^n определим вектор-функцию $\widetilde{h}(\widetilde{\beta})$, полагая для произвольного набора $\widetilde{\beta} = (\beta_1, \beta_2, \dots, \beta_n) \in B^n$

$$\widetilde{h}(\widetilde{\beta}) = \sum_{i=1}^{n} \beta_i \widetilde{e}^{(k)}(i). \tag{1}$$

Очевидно, что $\widetilde{h}(\widetilde{\beta})$ — это двоичный набор длины k, полученный в результате сложения наборов, являющихся двоичны-

ми записями (с помощью k цифр) номеров единичных разрядов набора $\widetilde{\beta}$. Рассмотрим код H_n, состоящий из всех наборов $\widetilde{\beta} = (\beta_1, \beta_2, \ldots, \beta_n) \in B^n$ таких, что

$$\widetilde{h}(\widetilde{\beta}) = (\underbrace{0, 0, \ldots, 0}_{k}); \tag{2}$$

этот код называется *кодом Хэмминга*.

Пример. Положим $n = 6$, $\widetilde{\beta} = (110011)$ и $\widetilde{\gamma} = (110001)$. Тогда $k = 3$, $\widetilde{h}(\widetilde{\beta}) = (0,0,1) \oplus (0,1,0) \oplus (1,0,1) \oplus (1,1,0) = (0,0,0)$ и $\widetilde{h}(\widetilde{\gamma}) = (0,0,1) \oplus (0,1,0) \oplus (1,1,0) = (1,0,1)$. Следовательно, $\widetilde{\beta} \in H_6$, $\widetilde{\gamma} \notin H_6$.

Теорема 31. *Код Хэмминга исправляет одну ошибку.*

Доказательство. Для произвольного двоичного набора $\widetilde{\alpha} = (\alpha_1, \alpha_2, \ldots, \alpha_k)$ через $N(\widetilde{\alpha})$ обозначим число, двоичной записью которого является набор $\widetilde{\alpha}$, т. е. положим

$$N(\widetilde{\alpha}) = \sum_{i=1}^{k} \alpha_i 2^{k-i}.$$

Будем считать $N(\widetilde{\alpha})$ *числовым значением набора* $\widetilde{\alpha}$. Рассмотрим однозначное отображение (декодирование), при котором каждому набору $\widetilde{\beta} \in B^n$ ставится в соответствие набор $\widetilde{\beta}$, если $\widetilde{\beta} \in H_n$, или набор, полученный из $\widetilde{\beta}$ заменой разряда с номером $N(\widetilde{h}(\widetilde{\beta}))$ на противоположный, если $\widetilde{\beta} \notin H_n$. Убедимся, что если в произвольном кодовом наборе произойдёт любая одна ошибка, то эта ошибка будет исправлена при указанном декодировании.

Действительно, пусть в некотором наборе $\widetilde{\beta} = (\beta_1, \beta_2, \ldots, \beta_n) \in H_n$ произошла ошибка в j-м разряде, в результате чего получился набор $\widetilde{\beta}' = (\beta_1, \beta_2, \ldots, \beta_{j-1}, \beta_j \oplus 1, \beta_{j+1}, \ldots, \beta_n)$. Так как $\widetilde{\beta} \in H_n$, то

$$\widetilde{h}(\widetilde{\beta}') = \sum_{i=1}^{n} \beta_i \widetilde{e}^{(k)}(i) \oplus \widetilde{e}^{(k)}(j) =$$

$$= \widetilde{h}(\widetilde{\beta}) \oplus \widetilde{e}^{(k)}(j) = (0, \ldots, 0) \oplus \widetilde{e}^{(k)}(j) = \widetilde{e}^{(k)}(j).$$

Произведя в наборе $\widetilde{\beta}'$ замену разряда с номером $N(\widetilde{h}(\widetilde{\beta}')) = N(\widetilde{e}^{(k)}(j)) = j$, получим исходный набор $\widetilde{\beta}$. Теорема доказана.

Пример. Пусть в наборе $\widetilde{\beta} = (110011) \in H_6$ произошла ошибка в 5-м разряде, в результате чего получился набор $\widetilde{\beta}' = (110001)$. Так как $\widetilde{h}(\widetilde{\beta}') = (0,0,1) \oplus (0,1,0) \oplus (1,1,0) =$

$= (1, 0, 1)$, то числовое значение набора $\widetilde{h}(\widetilde{\beta}')$ равняется 5, и это значение определяет номер искажённого разряда.

Из определения вектор-функции $\widetilde{h}(\widetilde{\beta})$ (соотношения (1)) вытекает, что j-й разряд $h_j(\widetilde{\beta})$ набора $\widetilde{h}(\widetilde{\beta}) = (h_1(\widetilde{\beta}), \ldots, h_k(\widetilde{\beta}))$ есть сумма по модулю 2 тех и только тех разрядов набора $\widetilde{\beta}$, у которых двоичная запись (с помощью k цифр) номеров содержит 1 в j-м разряде. Условие принадлежности набора $\widetilde{\beta}$ коду Хэмминга H_n (соотношение (2)) можно записать в виде

$$\begin{cases} h_1(\widetilde{\beta}) = 0, \\ \ldots \\ h_k(\widetilde{\beta}) = 0. \end{cases} \tag{3}$$

Двоичная запись числа 2^{k-j} $(j = 1, 2, \ldots, k)$ с помощью k цифр, т. е. набор $\widetilde{e}^{(k)}(2^{k-j})$ содержит, очевидно, ровно одну единицу в j-м разряде, поэтому разряд $\beta_{2^{k-j}}$ входит лишь в j-е уравнение из (3). Прибавив (по модулю 2) к обеим частям равенства $h_j(\widetilde{\beta}) = 0$ разряд $\beta_{2^{k-j}}$, получим эквивалентное ему соотношение $h'_j(\widetilde{\beta}) = \beta_{2^{k-j}}$, где $h'_j(\widetilde{\beta})$ получается из $h_j(\widetilde{\beta})$ удалением слагаемого $\beta_{2^{k-j}}$. Условие (3) можно переписать в виде

$$\begin{cases} h'_1(\widetilde{\beta}) = \beta_{2^{k-1}}, \\ \ldots \\ h'_k(\widetilde{\beta}) = \beta_{2^0}. \end{cases} \tag{4}$$

Суммы $h'_1(\widetilde{\beta}), \ldots, h'_k(\widetilde{\beta})$ содержат лишь те разряды набора $\widetilde{\beta}$, номера которых отличны от степеней двух; эти разряды назовём *информационными*. Из (4) видно, что информационные $(n - k)$ разрядов можно взять произвольными, и тогда остальные k разрядов $\beta_{2^0}, \beta_{2^1}, \ldots, \beta_{2^{k-1}}$ однозначно определяются из условия (4); разряды $\beta_{2^0}, \beta_{2^1}, \ldots, \beta_{2^{k-1}}$ назовём *контрольными*. Из сказанного вытекает следующее утверждение.

Теорема 32. *Код Хэмминга H_n содержит 2^{n-k} кодовых наборов, где $k = \lceil \log(n + 1) \rceil$.*

§ 5. Геометрические свойства самокорректирующихся кодов. Оценки Хэмминга и Гильберта

Будем рассматривать единичный n-мерный куб B^n как метрическое пространство, где для любых двух двоичных наборов

(точек) $\widetilde{\beta} = (\beta_1, \ldots, \beta_n)$ и $\widetilde{\gamma} = (\gamma_1, \ldots, \gamma_n)$ расстояние $\rho(\widetilde{\beta}, \widetilde{\gamma})$ между ними равно числу разрядов, в которых они различаются, т. е.

$$\rho(\widetilde{\beta}, \widetilde{\gamma}) = \sum_{i=1}^{n} |\beta_i - \gamma_i|.$$

Пусть $\widetilde{\beta}$ — некоторый набор из B^n. Множество наборов $\widetilde{\beta}'$, таких, что $\rho(\widetilde{\beta}, \widetilde{\beta}') \leqslant p$, называется *шаром радиуса p с центром в $\widetilde{\beta}$.*

Теорема 33. *Множество двоичных наборов из B^n составляет код, исправляющий p ошибок, в том и только в том случае, когда расстояние между любыми двумя наборами этого множества не меньше, чем $2p + 1$.*

Доказательство. Если условие теоремы выполнено и при передаче кодового набора $\widetilde{\beta}$ по каналу связи произошло $p' \leqslant p$ ошибок, то принятый набор $\widetilde{\beta}'$ будет отстоять от любого другого кодового набора $\widetilde{\gamma}$ не менее, чем на $2p + 1 - p' > p$, и переданный набор $\widetilde{\beta}$ определяется однозначно.

Если же расстояние между некоторыми кодовыми наборами $\widetilde{\beta}$ и $\widetilde{\gamma}$ не превосходит $2p$, то найдётся набор $\widetilde{\delta}$, отстоящий от $\widetilde{\beta}$ и от $\widetilde{\gamma}$ не более, чем на p. При получении набора $\widetilde{\delta}$, очевидно, нельзя однозначно установить, из какого набора ($\widetilde{\beta}$ или $\widetilde{\gamma}$) он произошёл при имевших место p' ошибках ($p' \leqslant p$). Теорема доказана.

Из этой теоремы и теоремы 31 вытекает

Следствие 1. *Расстояние между любыми двумя различными кодовыми наборами из H_n не меньше трёх.*

Отсюда и из теоремы 32 получаем ещё

Следствие 2. *При $n = 2^t - 1$, где t — целое число, единичный n-мерный куб может быть разбит на единичные шары* (т. е. шары радиуса 1).

Последняя теорема допускает следующую геометрическую интерпретацию.

Около каждого кодового набора опишем шар радиуса p. Все наборы, которые могут получиться из кодового набора $\widetilde{\beta}$ в результате не более, чем p ошибок, содержатся в шаре с центром $\widetilde{\beta}$. Если расстояние между кодовыми наборами не меньше $2p + 1$, то шары не пересекаются, и по искажённому набору соответствующий кодовый набор (центр шара) восстанавливается однозначно. Если же расстояние между какими-либо кодовыми наборами не превосходит $2p$, то соответствующие этим наборам

шары пересекаются и по наборам, попавшим в пересечение, центры шаров однозначно указать нельзя.

Подобные рассуждения позволяют достаточно просто получить оценки для числа кодовых наборов в самокорректирующихся кодах.

Пусть K_p^n — произвольный самокорректирующийся код, исправляющий p ошибок и представляющий собой некоторое множество наборов из B^n. Из теоремы 33 следует, что шары радиуса p, описанные около кодовых наборов из K_p^n, не пересекаются. Каждый такой шар содержит $1 + n + C_n^2 + \ldots + C_n^p$ наборов (отличающихся от центра не более чем в p разрядах), откуда следует

$$|K_p^n|(1 + n + C_n^2 + \ldots + C_n^p) \leqslant 2^n,$$

или

$$|K_p^n| \leqslant \frac{2^n}{1 + n + C_n^2 + \ldots + C_n^p};$$

последняя оценка носит название *верхней границы Хэмминга*.

Код K_p^n называется *максимальным*, если он содержит наибольшее число кодовых наборов (при заданных n и p).

Пусть $K_{p,\max}^n$ — произвольный максимальный код из n-разрядных наборов, исправляющий p ошибок. Найдём нижнюю оценку мощности этого кода.

Построим код, исправляющий p ошибок, следующим образом. В качестве первого кодового набора возьмём произвольный n-разрядный двоичный набор $\widetilde{\beta}_1$. Затем к $\widetilde{\beta}_1$ добавим какой-нибудь набор $\widetilde{\beta}_2$, лежащий вне шара радиуса $2p$ с центром в $\widetilde{\beta}_1$. На третьем шаге в код включим набор $\widetilde{\beta}_3$, не принадлежащий шарам радиуса $2p$, описанным около $\widetilde{\beta}_1$ и $\widetilde{\beta}_2$, и т. д. Построение завершается, когда после очередного l-го шага не останется наборов вне шаров радиуса $2p$ с центрами в $\widetilde{\beta}_1, \widetilde{\beta}_2, \ldots, \widetilde{\beta}_l$. Расстояние между любыми двумя наборами из множества $\{\widetilde{\beta}_1, \ldots, \widetilde{\beta}_l\}$ по построению не менее $2p + 1$, и поэтому построенный код согласно теореме 33 исправляет p ошибок.

Каждый шар радиуса $2p$ содержит $1 + n + C_n^2 + \ldots + C_n^{2p}$ наборов. Шары с центрами в $\widetilde{\beta}_1, \ldots, \widetilde{\beta}_l$ содержат все 2^n наборов из B^n, поэтому

$$l(1 + n + C_n^2 + \ldots + C_n^{2p}) \geqslant 2^n$$

(иначе существовал бы набор, лежащий вне рассматриваемых шаров с центрами в $\widetilde{\beta}_1, \ldots, \widetilde{\beta}_l$), откуда получаем

$$l \geqslant \frac{2^n}{1 + n + C_n^2 + \ldots + C_n^{2p}}.$$

Учитывая, что мощность максимального кода $K_{p,\max}^n$ не меньше мощности l построенного кода, получаем оценку

$$|K_{p,\max}^n| \geqslant \frac{2^n}{1 + n + C_n^2 + \ldots + C_n^{2p}},$$

которая носит название *нижней границы Гильберта*.

Из теоремы 32 и верхней границы Хэмминга вытекает

Следствие 3. *При* $n = 2^t - 1$, *где* t — *целое число, код Хэмминга является максимальным.*

Действительно, в этом случае $k = \lceil \log(n+1) \rceil = t$ и согласно теореме 32 в H_n входят 2^{n-t} кодовых наборов. С другой стороны, согласно верхней границе Хэмминга имеем

$$|H_n| \leqslant \frac{2^n}{n+1} = 2^{n-t},$$

т. е. множество наборов H_n имеет максимальную мощность.

Глава X

ДИСКРЕТНЫЕ ЭКСТРЕМАЛЬНЫЕ ЗАДАЧИ

§ 1. Задача на покрытие. Точное решение задачи на покрытие

Характерной особенностью дискретных экстремальных задач, как правило, бывает проявляющаяся в том или ином виде дискретность решений таких задач. Скажем, в задачах синтеза схем из функциональных элементов дискретность проявляется, например, в том, что любая схема содержит целое число элементов; в задачах поиска кратчайших путей на графах дискретность можно усмотреть в том, что всякое ребро графа либо входит в некоторый путь, либо не входит в него, и т. п. Экстремальность рассматриваемых задач обычно заключается в экстремальности искомых решений: ищут схему, реализующую заданную булеву функцию и содержащую при этом наименьшее возможное число функциональных элементов; выбирают путь, соединяющий две заданные вершины графа и имеющий наименьшую возможную длину; в заданной сети строят целочисленный поток наибольшей величины и т. п.

При решении дискретных экстремальных задач обычно бывает нетрудно построить некоторые достаточно простые (в содержательном смысле) алгоритмы переборного характера для нахождения нужного решения, однако такие алгоритмы фактически непригодны для практического использования из-за неприемлемо большой трудоёмкости. Гораздо труднее подыскать подходящие алгоритмы, использующие относительно небольшое или хотя бы даже просто приемлемое число элементарных операций. Собственно, этим обстоятельством и объясняется тот факт, что лишь для относительно небольшого числа дискретных экстремальных задач известны подходящие для практического использования алгоритмы поиска точных решений. В связи с этим для многих содержательных и важных в прикладном отношении задач ограничиваются поиском приближённых (в том или ином смысле) ре-

шений; при этом большую актуальность приобретает и проблема оценки качества получаемых решений. Сказанное проиллюстрируем ниже на конкретных примерах и первой рассмотрим задачу на покрытие.

Покрытием множества $A = \{a_1, \ldots, a_m\}$ называется любая система $(A_1, A_2, \ldots, A_n)$ подмножеств множества A, объединение которых равно A; подмножества $A_1, \ldots, A_n$ считаются при этом *элементами покрытия* $(A_1, \ldots, A_n)$. Покрытие считается *тупиковым*, если при удалении из него любого одного элемента оно перестаёт быть покрытием.

При заданной системе подмножеств $(A_1, \ldots, A_n)$ задача нахождения минимального покрытия (или просто, как говорят, задача на покрытие) заключается в нахождении покрытия, являющегося частью заданного покрытия $(A_1, \ldots, A_n)$, содержащей наименьшее число элементов покрытия (т. е. подмножеств из числа $A_1, \ldots, A_n$).

Покрытие можно представить булевой матрицей (таблицей) $\|\alpha_{ij}\|$. Строки этой матрицы соответствуют элементам $a_1, \ldots, a_m$ множества A, а столбцы — элементам покрытия $A_1, \ldots, A_n$. Элемент α_{ij} $(i = 1, \ldots, m;\ j = 1, \ldots, n)$ данной матрицы равен 1 в том и только в том случае, когда элемент a_i входит в подмножество A_j, т. е.

$$\alpha_{ij} = \begin{cases} 1, & \text{если } a_i \in A_j, \\ 0, & \text{если } a_i \notin A_j; \end{cases}$$

если $\alpha_{ij} = 1$, то считаем, что j-й столбец *покрывает* i-ю строку. Всякую задачу на покрытие, очевидно, можно свести к задаче на покрытие булевой матрицы, заключающейся в том, чтобы для заданной матрицы найти наименьшее число её столбцов, в совокупности покрывающих все строки матрицы (при этом подразумевается, что в матрице нет строк из одних нулей).

Один из способов решения задачи на покрытие заключается в следующем. Сопоставим i-й строке матрицы $\|\alpha_{ij}\|$ $(i = 1, \ldots, m)$ дизъюнкцию

$$D_i = A_{j(i,1)} \vee A_{j(i,2)} \vee \ldots \vee A_{j(i,r(i))},$$

которая в качестве слагаемых $A_{j(i,1)}, A_{j(i,2)}, \ldots, A_{j(i,r(i))}$ включает символы (имена) всех столбцов заданной матрицы, покрывающих i-ю строку (символы $A_1, \ldots, A_n$ в данном случае рассматриваем как символы булевых переменных). Составим затем конъюнкцию всех дизъюнкций D_i $(i = 1, \ldots, m)$:

$$\&_{i=1}^{m} (A_{j(i,1)} \vee A_{j(i,2)} \vee \ldots \vee A_{j(i,r(i))}).$$

В полученном выражении («произведении сумм») раскроем скобки (используя дистрибутивный закон) и проведём упрощения с использованием законов коммутативности, ассоциативности, идемпотентности ($A_jA_j = A_j$) и поглощения ($B \vee BC = B$). В итоге придём к дизъюнкции конъюнкций («сумме произведений»), в которой:

а) каждая конъюнкция содержит попарно различные символы;

б) ни одна из конъюнкций не входит (в качестве сомножителя) в другую конъюнкцию.

Конъюнкции полученного в итоге выражения, как нетрудно заметить, определяют все тупиковые покрытия исходной матрицы, а содержащая наименьшее число символов (столбцов) конъюнкция и даёт решение задачи — указанные в этой конъюнкции столбцы матрицы составляют минимальное покрытие.

Однако указанный способ нахождения минимального покрытия уже при относительно небольших размерах рассматриваемых матриц (скажем, 10^2 строк и столько же столбцов) требует выполнения весьма большого числа элементарных операций и потому для решения прикладных задач фактически непригоден.

§ 2. Градиентный алгоритм поиска приближённого решения. Оценка сложности градиентного покрытия

Для приближённого решения задачи на покрытие известен простой эвристический метод, который носит название *градиентного алгоритма* (другие названия этого же метода: *жадный алгоритм*; *метод наискорейшего спуска*; *метод вычерпывания*). Метод заключается в пошаговом отборе таких столбцов таблицы, каждый из которых вносит (на соответствующем шаге) наибольший вклад в покрытие остающихся строк.

Более точно этот алгоритм применительно к некоторой заданной булевой матрице M можно представить так.

1) Возьмём в качестве R исходную матрицу M, а в качестве P пустое множество.

2) В R выберем столбец S_R, покрывающий наибольшее число строк, и отнесём выбранный столбец к P (если указанных столбцов S_R несколько, то выбираем из них столбец с наименьшим номером).

3) Вычеркнем из матрицы R все строки, покрытые столбцом S_R. Если все строки матрицы R оказались вычеркнутыми (покрытыми), то процедура построения на этом заканчивается и в качестве покрытия берётся полученное множество P. Если R

содержит ещё какие-то строки (непокрытые), то возвращаемся к п. 2).

Проанализируем приближённое решение задачи на покрытие, полученное с помощью градиентного алгоритма.

Под *градиентным покрытием* матрицы M будем далее подразумевать покрытие данной матрицы, полученное с помощью градиентного алгоритма. Под *сложностью покрытия* будем подразумевать число столбцов в этом покрытии.

Пусть M — произвольная булева матрица с m строками и n столбцами, содержащая в каждой строке и в каждом столбце не менее чем по одной единице. Обозначим через $l_{\min}$ сложность минимального покрытия этой матрицы, а через $l_{\text{ГА}}$ — сложность её градиентного покрытия.

Пусть $Q = \{S_1, \ldots, S_{l_{\min}}\}$ — некоторое минимальное покрытие матрицы M, $A(l)$ — множество строк, покрытых после l проходов градиентного алгоритма, $B(l)$ — множество строк, оставшихся непокрытыми после l-го прохода градиентного алгоритма. Ниже будем считать $l_{\min} \geqslant 2$ (при $l_{\min} = 1$ имеем тривиальный случай, когда минимальное покрытие состоит из одного столбца и находится с помощью градиентного алгоритма за один проход).

Лемма 8. $|A(l)| \geqslant \dfrac{m}{l_{\min}} \displaystyle\sum_{i=0}^{l-1} \left(1 - \frac{1}{l_{\min}}\right)^i$.

Доказательство проведём индукцией по l. При $l = 1$, очевидно, $|A(l)| \geqslant \frac{m}{l_{\min}}$.

Предположим, что неравенство леммы верно для номеров проходов $1, 2, \ldots, l-1$; докажем его для l-го прохода. Если $|A(l)| = m$, то лемма доказана (формально: $\sum_{i=0}^{\infty} \left(1 - \frac{1}{l_{\min}}\right)^i = l_{\min}$). В противном случае существует непустое подмножество Q' ($Q' \subset Q$) столбцов, не вошедших в градиентное покрытие P. Множество $B(l-1)$ этим подмножеством Q' покрывается. При этом по крайней мере для одного столбца из Q' количество покрываемых им строк не менее $\dfrac{1}{l_{\min}}|B(l-1)|$, т. е. не менее $\dfrac{m - |A(l-1)|}{l_{\min}}$. Поэтому

$$|A(l)| \geqslant |A(l-1)| + \frac{m - |A(l-1)|}{l_{\min}} =$$

$$= \frac{m}{l_{\min}} + \left(1 - \frac{1}{l_{\min}}\right)|A(l-1)| \geqslant \{\text{по предположению индукции}\} \geqslant$$

$$\geqslant \frac{m}{l_{\min}} + \frac{m}{l_{\min}} \sum_{i=1}^{l-1} \left(1 - \frac{1}{l_{\min}}\right)^i =$$

$$= \frac{m}{l_{\min}} \sum_{i=0}^{l-1} \left(1 - \frac{1}{l_{\min}}\right)^i = m\left(1 - \left(1 - \frac{1}{l_{\min}}\right)^l\right).$$

Лемма доказана.

Следствие. $|B(l)| \leqslant m\left(1 - \frac{1}{l_{\min}}\right)^l$.

Обозначим через l^* минимальное натуральное число, такое, что $|B(l^*)| \leqslant l_{\min}$.

Лемма 9. $l_{\text{ГА}} \leqslant l^* + l_{\min}$.

Доказательство. На каждом шаге градиентного алгоритма покрывается по крайней мере одна новая строка. Поскольку $|B(l^*)| \leqslant l_{\min}$, то до получения покрытия осталось сделать не более $l_{\min}$ шагов; отсюда и получается оценка леммы.

Теорема 34 (о сложности градиентного покрытия).

$$l_{\text{ГА}} \leqslant l_{\min}\left(1 + \frac{1}{l_{\min}}\left\lceil \frac{\ln(m/l_{\min})}{\ln(l_{\min}/(l_{\min}-1))}\right\rceil\right).$$

Доказательство. Оценим сверху l^*. Решим относительно l вспомогательное уравнение $m\left(1 - \frac{1}{l_{\min}}\right)^l = l_{\min}$; если решение этого уравнения обозначим через L, то для L получаем выражение

$$L = \frac{\ln(m/l_{\min})}{\ln(l_{\min}/(l_{\min}-1))}.$$

Поскольку $\lceil L \rceil \geqslant L$, а $1 - \frac{1}{l_{\min}} < 1$, то

$$m\left(1 - \frac{1}{l_{\min}}\right)^{\lceil L \rceil} \leqslant m\left(1 - \frac{1}{l_{\min}}\right)^L = l_{\min}.$$

Отсюда и из следствия к лемме 8 получаем

$$|B(\lceil L \rceil)| \leqslant m\left(1 - \frac{1}{l_{\min}}\right)^{\lceil L \rceil} \leqslant l_{\min},$$

т. е.

$$l^* \leqslant \lceil L \rceil = \left\lceil \frac{\ln(m/l_{\min})}{\ln(l_{\min}/(l_{\min}-1))} \right\rceil.$$

Отсюда и из леммы 9 получаем

$$l_{\text{ГА}} \leqslant \left\lceil \frac{\ln(m/l_{\min})}{\ln(l_{\min}/(l_{\min}-1))} \right\rceil + l_{\min} = \\ = l_{\min}\left(1 + \frac{1}{l_{\min}} \left\lceil \frac{\ln(m/l_{\min})}{\ln(l_{\min}/(l_{\min}-1))} \right\rceil\right).$$

Теорема доказана.

Для отношения $\frac{l_{\text{ГА}}}{l_{\min}}$ — отклонения градиентного покрытия от минимального — получаем оценку

$$\frac{l_{\text{ГА}}}{l_{\min}} \leqslant 1 + \frac{1}{l_{\min}} \left\lceil \frac{\ln(m/l_{\min})}{\ln(l_{\min}/(l_{\min}-1))} \right\rceil.$$

Более грубые оценки можно получить, если использовать разложение $\ln(1+z) = z - \frac{z^2}{2} + \frac{z^3}{3} - \frac{z^4}{4} + \ldots$ при $|z| < 1$. Из этого разложения следует, что $\ln\left(1 - \frac{1}{l_{\min}}\right) \leqslant -\frac{1}{l_{\min}}$, а

$$\ln \frac{l_{\min}}{l_{\min}-1} = \ln \frac{1}{1-\frac{1}{l_{\min}}} = -\ln\left(1 - \frac{1}{l_{\min}}\right) \geqslant \frac{1}{l_{\min}}.$$

Используя последнее неравенство и оценку теоремы 34, получаем для сложности градиентного покрытия оценку

$$l_{\text{ГА}} \leqslant l_{\min}\left(1 + \frac{1}{l_{\min}}\left(\frac{\ln(m/l_{\min})}{\frac{1}{l_{\min}}} + 1\right)\right) = \\ = l_{\min}\left(\ln(m/l_{\min}) + 1 + \frac{1}{l_{\min}}\right),$$

а для отклонения —

$$\frac{l_{\text{ГА}}}{l_{\min}} \leqslant \ln(m/l_{\min}) + 1 + \frac{1}{l_{\min}}.$$

Полученные оценки достаточно хороши; об этом свидетельствует следующий пример.

Пусть M_0 есть матрица $2^k \times k$, состоящая из всевозможных двоичных наборов длины k; пусть, далее, M_i — матрица $2^k \times 3$, в которой i-й столбец состоит из единиц, а два остальных — из нулей ($i = 1, 2, 3$). Рассмотрим матрицу M, составленную из подматриц M_0, M_1, M_2, M_3:

$$M = \left(\begin{array}{c|c} M_1 & M_0 \\ \hline M_2 & M_0 \\ \hline M_3 & M_0 \end{array}\right).$$

В этой матрице $m = 3 \cdot 2^k$ строк и $n = k + 3$ столбцов. Отметим, что первые три столбца содержат по 2^k единиц, а все остальные — по $3 \cdot 2^{k-1}$ единиц. Градиентный алгоритм на первых k шагах первые три столбца не выберет; его применение, очевидно, приведёт к тривиальному покрытию, содержащему все $k + 3$ столбцов, т. е. в данном примере $l_{\text{ГА}} = k + 3$.

В то же время минимальное покрытие образуют первые три столбца, т. е. $l_{\min} = 3$. Отсюда получаем

$$\frac{l_{\text{ГА}}}{l_{\min}} = \frac{k}{3} + 1,$$

что по порядку равно правой части из оценки для отклонения:

$$\frac{l_{\text{ГА}}}{l_{\min}} \leqslant \ln(m/l_{\min}) + 1 + \frac{1}{l_{\min}} = \ln \frac{3 \cdot 2^k}{3} + 1 + \frac{1}{3} = k \ln 2 + \frac{4}{3}.$$

Таким образом, полученную выше оценку сложности покрытия градиентным алгоритмом в общем случае существенно улучшить нельзя.

§ 3. Задача о минимальном остовном дереве

Приведём теперь примеры дискретных экстремальных задач, для которых известны эффективные, т. е. вполне доступные по числу выполняемых элементарных операций алгоритмы. Первой рассмотрим задачу о поиске минимального остовного дерева.

Пусть G — неориентированный связный граф. Остовным деревом, или *остовом* графа G называется всякий связный подграф графа G, являющийся деревом и содержащий все вершины графа G. Остов графа, очевидно, можно получить последовательным удалением циклических рёбер графа, что делается достаточно просто. Ниже рассматривается более сложная задача — поиск остова во взвешенном графе, которая может иметь в ряде случаев содержательную интерпретацию.

Граф G называется *взвешенным*, если каждому ребру e этого графа приписано неотрицательное число $c(e)$ — *вес* ребра (разным рёбрам, естественно, могут быть приписаны равные веса).

Рассмотрим задачу нахождения во взвешенном графе остова с минимальным общим весом рёбер, т. е. остовного дерева, у которого сумма весов всех его рёбер минимальна; такое дерево называется *минимальным* остовным деревом (или *минимальным* остовом). Опишем два алгоритма поиска минимального остовного дерева для связного взвешенного графа G с n вершинами.

Первый алгоритм (жадный). На первом шаге выбираем ребро с наименьшим весом и включаем его в искомое дерево (вместе с инцидентными ему вершинами). На каждом следующем очередном шаге выбираем новое ребро с наименьшим весом, не образующее циклов с уже выбранными рёбрами. Этот процесс продолжается до тех пор, пока не будет выбрано $(n-1)$ рёбер, образующих (в соответствии с одним из характеристических свойств дерева) остовное дерево. Подобный алгоритм называется градиентным (или жадным). Очевидно, указанный алгоритм требует конечного числа шагов. Докажем, что полученное с его помощью остовное дерево является минимальным.

Пусть с использованием градиентного алгоритма получено остовное дерево D; предположим, что D не является минимальным. Пусть градиентный алгоритм включает в D рёбра $e_1, e_2, \ldots, e_{n-1}$ так, что эти рёбра идут в порядке неубывания их весов. Возьмём минимальное остовное дерево $D_{\min}$, которое включает в себя рёбра $e_1, e_2, \ldots, e_{i-1}$ для наибольшего возможного i. Очевидно, $i \geqslant 1$; с другой стороны, $i \leqslant n-1$, т. к. D не является минимальным остовным деревом. При добавлении ребра e_i к дереву $D_{\min}$ появляется простой цикл Z, причём единственный, ибо любые две вершины (в том числе и инцидентные добавленному ребру e_i) в дереве $D_{\min}$ соединены единственной простой цепью — иначе уже в дереве $D_{\min}$ был бы цикл. Этот цикл Z должен включать некоторое ребро e, отличное от $e_1, \ldots, e_i$, поскольку $e_1, \ldots, e_i$ в дереве D цикла не образуют. Поскольку e и $e_1, \ldots, e_{i-1}$ принадлежат дереву $D_{\min}$, в котором нет циклов, и поскольку градиентный алгоритм на каждом шаге добавляет ребро наименьшей стоимости, которое не образует циклов, то вес e должен быть не меньше веса e_i (иначе к $e_1, e_2, \ldots, e_{i-1}$ вместо e_i было бы добавлено ребро e). Если вес e больше, чем вес e_i, то остовное дерево $D'_{\min}$, получающееся из $D_{\min}$ изъятием ребра e и добавлением e_i, было бы остовным деревом с меньшим весом, чем у $D_{\min}$ — получаем противоречие. Получается, что веса e и e_i должны быть равны, в связи с чем $D'_{\min}$ будет минимальным остовным деревом. Но это противоречит нашему предположению о том, что i – наибольшее из возможных. Поэтому дерево D (построенное градиентным алгоритмом) будет минимальным остовным деревом.

Второй алгоритм (алгоритм ближайшего соседа). Этот алгоритм не требует ни сортировки рёбер по весам, ни проверки на цикличность каждого добавляемого к строящемуся дереву ребра.

Построение минимального остовного дерева начинается с произвольной вершины v_i в заданном графе G. Пусть (v_i, v_j) — ребро с наименьшим весом, инцидентное v_i; ребро (v_i, v_j) включаем в искомое дерево. Затем среди всех рёбер, инцидентных либо v_i, либо v_j, выбираем ребро с наименьшим весом и включаем его в частично построенное дерево. В результате этого в дерево добавляется новая вершина, скажем, v_k. Повторяя процесс, ищем ребро с наименьшим весом, соединяющее либо v_i, либо v_j, либо v_k с некоторой новой вершиной графа (отличной от v_i, v_j, v_k). Процесс продолжается до тех пор, пока все вершины из G не будут включены в дерево, т. е. пока дерево не станет остовным. Доказательство того, что алгоритм приводит к минимальному остовному дереву, аналогично доказательству для жадного алгоритма.

§ 4. Поиск кратчайшего и надёжного путей в графе

Пусть G — взвешенный граф. Пусть Z — какой-нибудь путь, соединяющий вершины v и w; *длиной пути* Z из v в w будем считать сумму весов всех входящих в данный путь рёбер. Возможность прикладной интерпретации задачи поиска кратчайшего пути, т. е. пути наименьшей длины, во взвешенном графе очевидна. Заметим, что без ограничения общности можно рассматривать в данной задаче графы без петель (отсутствие их в кратчайшем пути очевидно) и без кратных рёбер; при наличии кратных рёбер, соединяющих вершины v и w, следует оставить в графе лишь одно ребро — с наименьшим весом.

Опишем «отмечающий» алгоритм поиска кратчайшего пути между двумя вершинами v и w взвешенного графа G (заметим, что этот алгоритм применим как в случае неориентированного, так и в случае ориентированного графа, когда перемещение по дугам допускается только в направлении ориентации).

Вначале вершине v присваивается окончательная метка 0, а каждой из остальных вершин присваивается временная метка ∞. Далее при каждом проходе алгоритма одной вершине с временной меткой присваивается окончательная метка и поиск продолжается. Присвоение меток происходит следующим образом.

1) Каждой вершине j, не имеющей окончательной метки, присваивается новая временная метка — минимум из её прежней временной метки и суммы числа ω_{ij} с окончательной меткой вершины i, где i — вершина, которой была присвоена окончательная

метка на предыдущем шаге, а ω_{ij} — вес ребра, связывающего i-ю вершину с j-й; если такое ребро (для ориентированного графа — дуга, ведущая из i-й вершины в j-ю) отсутствует, то считается $\omega_{ij} = \infty$.

2) Находится наименьшая из всех временных меток, которая и становится окончательной меткой своей вершины. В случае равенства выбирается любая из них, например (для определённости), имеющая наименьший номер.

Процесс продолжается до тех пор, пока вершина w не получит окончательную метку. Первой помеченной окончательной меткой вершиной будет v, которая находится на нулевом расстоянии от себя. Следующей вершиной, получившей окончательную метку, будет вершина, ближайшая к v. Третьей вершиной, получившей окончательную метку, будет вторая по близости, и т. д. Легко видеть, что окончательная метка каждой вершины задаёт длину кратчайшего пути от начала v до данной вершины; это утверждение формально легко доказать индукцией по числу вершин с окончательной меткой. Действительно, выполнение базиса индукции — для вершины v — очевидно. Пусть $v, g, \ldots, p$ — первые k вершин, ближайшие к v; пусть, далее, q есть $(k+1)$-я по близости к v вершина. Возьмём кратчайший путь, связывающий v с q; смежная с q вершина из этого пути принадлежит, очевидно, множеству вершин $\{v, g, \ldots, p\}$. Но тогда вершиной q может быть только та вершина, которая выбирается при $(k+1)$-м проходе алгоритма — иначе получим противоречие с предположением о том, что q есть $(k+1)$-я по близости к v вершина.

В качестве примера на рис. 25 представлен граф, на котором требуется найти кратчайший путь между вершинами a и g. В табл. 15 отражены результаты применения отмечающего алгоритма для решения поставленной задачи; в этой таблице и на рис. 25 окончательные метки вершин заключены в скобки.

Искомый кратчайший путь между вершинами a и g находим, идя обратно от g по вершинам с окончательными метками. На первом шаге выбираем вершину, смежную с g и такую, что (её метка) + (расстояние до g) даёт метку вершины g; аналогично осуществляются второй и последующие шаги. Таким способом получим последовательность вершин g, e, b, c, d, a, из которой восстанавливаем кратчайший путь от a до g:

$$a \to d \to c \to b \to e \to g$$

(в данном случае путь указан входящими в него вершинами).

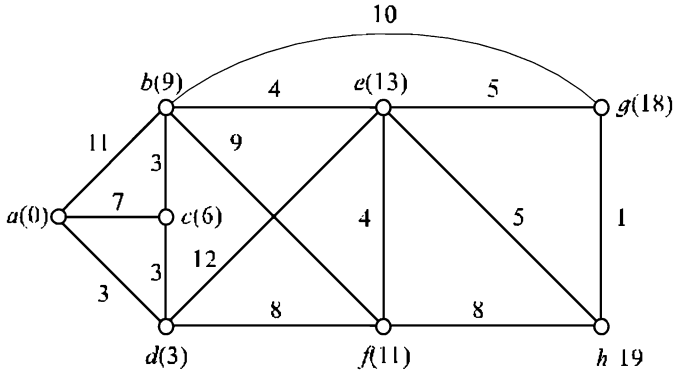


Рис. 25

Таблица 15

№ прохода алгоритма	a	b	c	d	e	f	g	h
1	(0)	∞	∞	∞	∞	∞	∞	∞
2	(0)	11	7	3	∞	∞	∞	∞
	(0)	11	7	(3)	∞	∞	∞	∞
3	(0)	11	6	(3)	15	11	∞	∞
	(0)	11	(6)	(3)	15	11	∞	∞
4	(0)	9	(6)	(3)	15	11	∞	∞
	(0)	(9)	(6)	(3)	15	11	∞	∞
5	(0)	(9)	(6)	(3)	13	11	19	∞
	(0)	(9)	(6)	(3)	13	(11)	19	∞
6	(0)	(9)	(6)	(3)	13	(11)	19	19
	(0)	(9)	(6)	(3)	(13)	(11)	19	19
7	(0)	(9)	(6)	(3)	(13)	(11)	18	19
	(0)	(9)	(6)	(3)	(13)	(11)	(18)	19

Заметим, что при использовании отмечающего алгоритма после расстановки окончательных меток для всех вершин графа будут найдены кратчайшие расстояния от заданной исходной вершины до всех вершин графа.

Выше предполагалось, что вес любого пути в графе — аддитивная функция, т. е. он равен сумме весов рёбер, образующих путь из одной вершины в другую. Но возможен и случай, когда требуется найти минимум мультипликативной функции. В этом случае можно прологарифмировать нашу функцию, решить за-

дачу для её логарифма, а затем полученный результат пропотенцировать. В качестве примера рассмотрим задачу о наиболее надёжном пути применительно к каналам связи.

Пусть задан граф G, в котором вершины задают некоторые пункты, а рёбра (или дуги) — каналы связи между соответствующими пунктами. Каждому ребру (или дуге) e_i приписано число $p(e_i)$ — вероятность безотказной работы канала связи e_i, т. е. надёжность канала e_i, причём $0 < p(e_i) < 1$. Предположим, что каналы связи выходят из строя статистически независимо друг от друга. В этом случае надёжность пути (связи) Z между пунктами v и w, т. е. вероятность безотказной связи между v и w будет равна произведению надёжностей соответствующих каналов, т. е. весов рёбер, входящих в Z. Обозначим надёжность пути Z через $P(Z)$; тогда

$$P(Z) = \prod_{e_i \in Z} p(e_i).$$

Нам требуется найти надёжный путь Z^*, т. е. такой, для которого

$$P(Z^*) = \max_Z P(Z),$$

где максимум берётся по всем путям, ведущим из v в w. Последнее равенство можно представить в виде

$$\frac{1}{P(Z^*)} = \min_Z \frac{1}{P(Z)}$$

или (после логарифмирования)

$$\ln \frac{1}{P(Z^*)} = \min_Z \ln \frac{1}{P(Z)} = \min_Z \ln \frac{1}{\prod\limits_{e_i \in Z} p(e_i)} = \min_Z \sum_{e_i \in Z} \ln \frac{1}{p(e_i)}.$$

Отсюда видно, что наша исходная задача отыскания надёжного пути сводится к задаче отыскания кратчайшего пути между заданными вершинами v и w на графе, который получается из исходного заменой веса $p(e_i)$ каждого ребра e_i на вес $\ln \frac{1}{p(e_i)}$.

§ 5. Точное решение задачи на покрытие методом динамического программирования

Опишем ещё один способ решения задачи на покрытие, связанный с методом динамического программирования. Суть этого способа заключается в том, что процесс поиска решения исходной задачи разбивается на ряд этапов, на которых нахо-

дятся решения аналогичных задач (на покрытие), но меньшей размерности. При поиске решения задачи большей размерности используются решения задач меньшей размерности; ключевую роль в данном случае играет рекуррентное соотношение, связывающее решение задачи большей размерности с решениями задач меньшей размерности.

Обозначим через M булеву $(m \times n)$-матрицу $||\alpha_{ij}||$, $i = 1, \ldots, m$; $j = 1, \ldots, n$. Через M_j обозначим подматрицу, составленную из строк матрицы M, не покрытых столбцом A_j, $j = 1, \ldots, n$; другими словами, подматрица M_j получается из матрицы M удалением всех тех строк, в которых на пересечении с j-м столбцом стоит единица. Сложность минимального покрытия матрицы M будем обозначать через $l_{\min}(M)$.

Убедимся, что для произвольной матрицы M выполняется соотношение

$$l_{\min}(M) = 1 + \min_{1 \leqslant j \leqslant n} l_{\min}(M_j). \tag{1}$$

Действительно, минимальное покрытие подматрицы M_j вместе со столбцом A_j даёт покрытие (не обязательно минимальное) всей матрицы M, откуда следует верхняя оценка

$$l_{\min}(M) \leqslant 1 + l_{\min}(M_j).$$

С другой стороны, если P — минимальное покрытие матрицы M и A_j — некоторый столбец из P, то после удаления A_j из P получим покрытие подматрицы M_j. Это следует из того, что j-й столбец в подматрице M_j — нулевой и никаких строк в ней не покрывает, а делают это все оставшиеся в P столбцы. Следовательно,

$$l_{\min}(M) - 1 \geqslant l_{\min}(M_j) \geqslant \min_{1 \leqslant j \leqslant n} l_{\min}(M_j)$$

и выполняется нижняя оценка

$$l_{\min}(M) \geqslant 1 + \min_{1 \leqslant j \leqslant n} l_{\min}(M_j).$$

Из полученных верхней и нижней оценок вытекает соотношение (1).

Вместе с соотношением (1) установлено, что при некотором j (при котором достигается минимум $l_{\min}(M_j)$) минимальное покрытие $P(M)$ матрицы M может быть получено из минимального покрытия $P(M_j)$ матрицы M_j добавлением к нему столбца A_j:

$$P(M) = P(M_j) \cup \{A_j\}.$$

Опираясь на рекуррентное соотношение (1), укажем по индукции способ построения минимального покрытия $P(M)$ матрицы M с использованием минимальных покрытий подматриц M^A, где A на s-м шаге будет пробегать все возможные сочетания из m строк по s ($s = 1, 2, \ldots, m$), а через M^A обозначена матрица со строками из A.

а) *Базис индукции*. Пусть $s = 1$. Для одноэлементного A, очевидно, $l_{\min}(M^A) = 1$; для каждого A зафиксируем какое-нибудь минимальное (из одного столбца) покрытие $P(M^A)$.

б) *Индуктивный переход*. Пусть для всех матриц M^A, таких, что $|A| \leqslant s - 1$ ($|A|$ — число строк подмножества A), уже определено значение $l_{\min}(M^A)$ и найдено минимальное покрытие $P(M^A)$. Возьмём произвольное множество A из s строк. Для него из соотношения (1) следует равенство

$$l_{\min}(M^A) = 1 + \min_{1 \leqslant j \leqslant n} l_{\min}(M_j^A).$$

где минимум можно искать только по ненулевым в M^A столбцам. Но любая подматрица M_j^A при этом содержит уже меньше, чем s строк, ибо $M_j^A = M^{A'}$ для некоторого $A' \subset A$. Для всех таких подматриц M_j^A на предыдущих шагах уже найдены как значения $l_{\min}(M_j^A)$, так и минимальные покрытия $P(M_j^A)$. Осталось только определить значение j, при котором $l_{\min}(M_j^A)$ оказывается наименьшим, и положить

$$l_{\min}(M^A) = 1 + l_{\min}(M_j^A),$$
$$P(M^A) = P(M_j^A) \cup \{A_j\}.$$

После выполнения подобной процедуры на m-м шаге найдём как значение $l_{\min}(M)$, так и какое-то минимальное покрытие $P(M)$.

Нетрудно заметить, что при двоичном кодировании подмножеств (и подматриц) все вычисления, необходимые для индуктивного перехода, можно выполнить с общим числом «элементарных» операций (типа арифметических), не превосходящим по порядку $mn \cdot 2^m$. Это означает, что в классе «широких» матриц, для которых $m = O(\log n)$, для представленного метода поиска решения задачи на покрытие справедлива полиномиальная верхняя оценка трудоёмкости. При тривиальном «переборном» способе решения задачи на покрытие число просматриваемых подмножеств столбцов по порядку равно 2^n, т. е. переборный алгоритм решения имеет экспоненциальную сложность (отно-

сительно «размерности» задачи, оцениваемой в данном случае числом $m + n$).

§ 6. Приближённое решение задачи об упаковке в контейнеры

Пусть задано конечное множество предметов $P = \{p_1, \ldots, p_n\}$. Для каждого предмета p_i известен его размер (например, объём предмета) $r_i = r(p_i)$; предполагается, что размер предмета r_i есть рациональное число, удовлетворяющее условию $0 < r_i \leqslant 1$, $i = 1, 2, \ldots, n$. Имеются одинаковые контейнеры, каждый размера 1 (к последнему условию, очевидно, можно прийти естественным образом — подходящей нормировкой размеров предметов и контейнеров). Требуется упаковать все предметы в контейнеры так, чтобы: а) для каждого контейнера сумма размеров помещённых в него предметов не превосходила единицы; б) было использовано наименьшее возможное число контейнеров.

Рассмотрим два алгоритма поиска приближённых решений поставленной задачи и оценим близость доставляемых этими алгоритмами решений к минимальным.

Первый алгоритм очень простой. Первый предмет помещается в первый контейнер. Очередной предмет p_i помещается в уже частично заполненный контейнер с наименьшим возможным номером; если это сделать невозможно, то берётся пустой контейнер, в него помещается p_i, а контейнер получает очередной номер.

Пусть в результате работы указанного алгоритма было использовано l_1 контейнеров; среди них не может быть двух или более контейнеров, заполненных не более, чем наполовину. Действительно, если контейнеры k_i и k_j заполнены не более, чем наполовину и $i < j$, т. е. контейнер k_i начал заполняться раньше, чем k_j, то первый же предмет, попавший в k_j, должен был быть помещённым в k_i (или в контейнер с меньшим номером), поскольку места там для него было достаточно. Следовательно, существуют $(l_1 - 1)$ контейнеров, заполненных более, чем наполовину, причём этих контейнеров не хватило для упаковки всех предметов. Отсюда вытекает неравенство

$$\frac{1}{2}(l_1 - 1) < \sum_{i=1}^{n} r_i;$$

кроме того, очевидно, выполняется неравенство

$$\sum_{i=1}^{n} r_i \leqslant l_{\min},$$

где $l_{\min}$ — наименьшее возможное число контейнеров, достаточное для упаковки всех предметов.

Из этих неравенств получаем $\frac{1}{2}(l_1 - 1) < l_{\min}$, или $l_1 - 1 < 2l_{\min}$; учитывая целочисленность параметров l_1 и $l_{\min}$, получаем окончательную оценку для точности решения, получаемого первым алгоритмом:

$$l_1 \leqslant 2l_{\min}.$$

Второй алгоритм учитывает достаточно простое эвристическое соображение: для более экономного использования пространства в контейнерах можно попытаться в первую очередь упаковывать наиболее крупные предметы, заполняя затем оставшиеся в контейнерах свободные места более мелкими. В этом случае вначале все предметы из P перенумеровываются в порядке убывания размеров, т. е. так, чтобы выполнялись неравенства $r_1 \geqslant r_2 \geqslant \ldots \geqslant r_n$, а уже затем осуществляется собственно упаковка в контейнеры с использованием первого алгоритма. Обозначим через l_2 число используемых контейнеров при упаковке предметов в соответствии со вторым алгоритмом.

Теорема 35 (об эффективности второго алгоритма упаковки). *Для произвольной задачи упаковки в контейнеры справедлива оценка*

$$l_2 < \frac{3}{2} l_{\min} + 1.$$

Доказательство. Предположим, что в результате проведённой (в соответствии со вторым алгоритмом) упаковки все l_2 контейнеров, кроме, быть может, одного, оказались заполненными не менее, чем на $\frac{2}{3}$. Тогда $\frac{2}{3}(l_2 - 1) < \sum_{i=1}^{n} r_i$ (поскольку имеется $l_2 - 1$ контейнеров, каждый из которых заполнен не менее, чем на $\frac{2}{3}$, и потребовался ещё один — l_2-й контейнер). Отсюда (учитывая ещё и очевидное неравенство $l_{\min} \geqslant \sum_{i=1}^{n} r_i$) получаем $\frac{2}{3}(l_2 - 1) < l_{\min}$, или, окончательно, $l_2 < \frac{3}{2} l_{\min} + 1$.

Рассмотрим теперь другой случай, когда после упаковки по крайней мере два контейнера оказались заполненными не более, чем на $\frac{2}{3}$. Множество P всех предметов разобьём на три подмножества A, B и C, полагая $A = \{p|\ r(p) > \frac{1}{2}\}$, $B = \{p|\ \frac{1}{3} < r(p) \leqslant \frac{1}{2}\}$, $C = \{p|\ r(p) \leqslant \frac{1}{3}\}$ (через $\{p|\ r(p) > \frac{1}{2}\}$

обозначаем подмножество предметов p, удовлетворяющих условию $r(p) > \frac{1}{2}$; аналогичный смысл имеют и другие обозначения); пусть $|A| = k$, $|B| = m$. Отметим, что в последнем контейнере находятся только предметы p_i из $A \cup B$, для которых $r(p_i) > \frac{1}{3}$: иначе предмет p_i из последнего контейнера с $r(p_i) \leqslant \frac{1}{3}$ можно было бы упаковать в некоторый контейнер K_j с номером $j < l_2$, заполненный не более чем на $\frac{2}{3}$. Это означает, что первый предмет, положенный в последний контейнер K_{l_2}, имел размер, бóльший $\frac{1}{3}$, и что для всех предметов из C при упаковке нашлись места в уже частично заполненных контейнерах. Следовательно, общее число занятых контейнеров определилось уже при укладке предметов из $A \cup B$, и это число с учётом ограничений на размеры предметов из A и из B не превосходит $k + \lceil \frac{m}{2} \rceil$.

При укладке (любым алгоритмом) предметов из A потребуется, очевидно, k контейнеров, а при укладке предметов из $A \cup B$ потребуется не менее, чем $\frac{k+m}{2}$ контейнеров (потому что в один контейнер войдет не более двух предметов из $A \cup B$).

В итоге получаем неравенства $l_2 \leqslant k + \lceil \frac{m}{2} \rceil$, $l_{\min} \geqslant k$, $l_{\min} \geqslant \frac{k+m}{2}$, из которых следует

$$l_2 < k + \frac{m}{2} + 1 = \frac{k}{2} + \frac{k+m}{2} + 1 \leqslant \frac{l_{\min}}{2} + l_{\min} + 1 = \frac{3}{2} l_{\min} + 1.$$

Теорема доказана.

§ 7. Классы P и NP. Полиномиальная сводимость задач

Дискретную экстремальную задачу П в общем случае можно представить как множество $\{Z_1, Z_2, \ldots\}$ индивидуальных (конкретных) задач, возникающих из П при конкретном выборе объектов и числовых параметров, используемых при постановке задачи П. При этом для каждой индивидуальной задачи $Z \in \Pi$ определена совокупность R допустимых решений r данной задачи, причём каждому решению r, $r \in R$, отвечает некоторое число $l(r)$ — сложность данного решения. Требуется найти алгоритм решения задачи П, который по произвольному входу, определяющему конкретную индивидуальную задачу Z, $Z \in \Pi$, находил бы решение r с оптимальным (максимальным или минимальным) значением $l(r)$. Именно такие структура и постановка были у рассмотренных выше дискретных экстремальных задач.

Существует также множество дискретных задач, решения которых могут и не обладать свойством экстремальности (но

обладать каким-либо иным свойством). Подобные задачи возникают, например, при решении вопроса об изоморфизме двух графов, при выяснении эквивалентности формул или схем из функциональных элементов и во многих других случаях.

Большинство таких дискретных задач допускает нахождение требуемого решения или ответа на поставленный вопрос через некоторую переборную процедуру. Например, поиск минимальной схемы из функциональных элементов для заданной булевой функции можно вести перебором всевозможных схем в порядке возрастания их сложности вплоть до первой схемы, реализующей заданную функцию; эквивалентность двух логических формул можно установить, перебирая все возможные наборы значений переменных из этих формул и сравнивая значения реализуемых формулами функций на каждом из наборов, и т. п.

Из соображений удобства математическая теория дискретных задач строится для задач распознавания свойств. Например, дискретные экстремальные задачи несколько видоизменяются и ставятся в форме вопроса: верно ли, что для данной индивидуальной задачи Z, $Z \in \Pi$, и значения l существует такое решение r, $r \in R$, что выполняется свойство $S(Z, r)$, заключающееся в том, что $l(r) \leqslant l$ (или, наоборот, $l(r) \geqslant l$ в задаче на максимум)? В такой постановке по-прежнему требуется построить алгоритм, который по предъявленным индивидуальной задаче Z и свойству $S(Z, r)$ находил бы правильный ответ — «да» или «нет». Если удастся построить подходящий алгоритм для решения последней задачи, то его, скорее всего, несложно будет приспособить и для решения исходной оптимизационной задачи нахождения оптимального значения $l(r)$: например, при целочисленных значениях $l(r)$, ограниченных степенной оценкой относительно «размера» индивидуальной задачи (скажем, длины кода, полностью определяющего данную задачу), поиск оптимального значения $l_{\text{опт}}$ можно эффективно организовать способом «деления отрезка пополам». Для других задач (с решениями не экстремального характера) соответствующие свойства обычно также легко усматриваются из самой постановки задачи. Скажем, в задаче о тождественной истинности формул над множеством $\{\&, \vee, {}^{-}, 0, 1\}$ в качестве Z может выступать любая конкретная формула Φ над $\{\&, \vee, {}^{-}, 0, 1\}$, в качестве r — набор $\widetilde{\sigma}$ значений переменных из Φ, а в качестве свойства $S(\Phi, \widetilde{\sigma})$ — равенство нулю значения функции ϕ, реализуемой формулой Φ, на наборе $\widetilde{\sigma}$ (если указанный набор $\widetilde{\sigma}$ существует, то это означает, что формула Φ тождественно истинной не является).

Исходные данные рассматриваемых индивидуальных задач будем предполагать закодированными в виде, например, двоичных наборов (из нулей и единиц). В детальное описание способов кодирования можно не вдаваться, в частности, потому, что если, скажем, при двух разных кодированиях исходных данных одной и той же задачи Z получаются кодирующие наборы длины n_1 и n_2 соответственно и можно подобрать такие полиномы (многочлены) P_1 и P_2, что $n_1 \leqslant P_2(n_2)$ и $n_2 \leqslant P_1(n_1)$, то такого рода полиномиальные «растяжения» или «сжатия» входной информации не отразятся (как это будет видно из определения класса P) на качественной оценке эффективности алгоритма. С учётом этого обстоятельства в качестве размера входа, т. е. длины кодирующего набора для исходных данных задачи, можно считать любое такое число n_1, что для любого реального кодирования исходных данных задачи Z получается набор длины n_2, причём для некоторых полиномов P_1 и P_2 выполняются приведённые выше неравенства.

Главным показателем качества алгоритма, решающего дискретную задачу, является время работы (или число выполняемых элементарных операций) алгоритма при решении задачи; формальное уточнение понятия алгоритма можно связывать, например, с машиной Тьюринга, время работы алгоритма — со временем работы машины Тьюринга и т. п.

Класс P составляют такие дискретные задачи (поставленные в форме распознавания), для которых существуют алгоритмы, решающие эти задачи с полиномиальной оценкой времени их работы относительно размера входа. Для дискретной задачи П из класса P это означает, что существуют решающий эту задачу алгоритм A и полином P_A, такие, что для любой индивидуальной задачи Z, $Z \in \Pi$, время решения задачи Z алгоритмом A не превосходит $P_A(|Z|)$, где $|Z|$ — размер входа индивидуальной задачи Z.

Отметим, что приведённое определение класса P оказывается устойчивым относительно выбора конкретной модели вычислительного устройства, адекватной понятию алгоритма, т. е. можно считать, что за одну единицу времени выполняется одна элементарная операция, которой может быть, скажем, вычисление значения булевой функции f на наборе $\widetilde{\alpha} = (\alpha_1, \ldots, \alpha_n)$ (при фиксированном n), арифметическая операция над n-разрядными двоичными числами, выполнение одной команды на машине Тьюринга и т. п. Переход от одной модели к другой приводит лишь к полиномиальному «ускорению» или «замедлению» процесса

вычислений, а это сказывается лишь на выборе полинома P_A из определения.

Примерами дискретных задач из P являются, как нетрудно заметить, задачи о минимальном остовном дереве, о поиске кратчайшего и надёжного путей в графе.

Перейдём к определению класса NP. Пусть имеется дискретная задача распознавания П, а для индивидуальной задачи Z, $Z \in \Pi$, и рассматриваемого свойства S ответом служит «да». Это означает, что имеется допустимое решение r, $r \in R$, обладающее свойством $S(Z, r)$; наличие решения r свидетельствует о правильности ответа «да» в данной индивидуальной задаче и может служить обоснованием такого ответа. Кроме того, можно поставить такой вопрос: возможно ли проверить и убедиться в том, что r является обоснованием ответа «да», за полиномиальное время? С учётом ответа на этот вопрос можно определить следующий класс дискретных задач.

Класс NP составляют такие дискретные задачи распознавания, для которых имеется алгоритм проверки обоснования ответа «да» с полиномиальной оценкой времени работы. Для дискретной задачи П из класса NP это означает, что существуют такие алгоритм A и полином P_A, что для любой индивидуальной задачи Z, $Z \in \Pi$, с ответом «да» можно указать такое решение r, по которому с помощью алгоритма A за время, не превосходящее $P_A(|Z|)$, можно установить, что r — допустимое решение и для него выполняется соответствующее свойство $S(Z, r)$.

Другими словами, $\Pi \in NP$, если за время, полиномиальное по $|Z|$, можно обосновать ответ «да» в любой индивидуальной задаче Z, $Z \in \Pi$, где он имеет место.

Примерами дискретных задач из NP могут служить задача на покрытие и задача об упаковке в контейнеры.

Вопрос о соотношении между классами P и NP (в частности, вопрос о том, совпадают или не совпадают эти классы) является открытым и составляет одну из крупнейших проблем современной математики. Отсюда должно быть ясно и то, что открытым является и вопрос о существовании алгоритмов, позволяющих решать многие важные в прикладном отношении дискретные экстремальные задачи за полиномиальное время (или, как ещё говорят, алгоритмов с полиномиальной сложностью). В связи с этим становится актуальным уже просто сравнение различных дискретных задач по трудности их решения; в основе такого сравнения лежит известная в математической логике и в теории алгоритмов ещё с тридцатых годов прошлого столетия идея сводимости задач.

Считается, что задача Π_1 сводится к задаче Π_2, если алгоритм решения задачи Π_2 можно использовать и для решения задачи Π_1. Ниже дадим более точное определение *полиномиальной сводимости*.

Пусть Π_1 и Π_2 — дискретные задачи распознавания. Будем говорить, что задача Π_1 (*полиномиально*) *сводится* к задаче Π_2, и обозначать этот факт как $\Pi_1 \propto \Pi_2$, если существуют алгоритм A и полином P_A, такие, что по любой индивидуальной задаче Z, $Z \in \Pi_1$, алгоритм A за время, не превосходящее $P_A(|Z|)$, строит такую индивидуальную задачу Z', $Z' \in \Pi_2$, что ответы в задачах Z и Z' совпадают.

Другими словами, $\Pi_1 \propto \Pi_2$, если можно построить сохраняющий ответы алгоритм, с полиномиальной сложностью преобразующий индивидуальные задачи из Π_1 в индивидуальные задачи из Π_2. Легко заметить, что отношение $\propto$ рефлексивно и транзитивно.

Важность введённого понятия сводимости обусловлена прежде всего следующими двумя соображениями.

Во-первых, если $\Pi_1 \propto \Pi_2$ и $\Pi_2 \in P$, то и $\Pi_1 \in P$. Действительно, пусть A — алгоритм, сводящий Π_1 к Π_2, A_2 — полиномиальный алгоритм, решающий задачу Π_2, а P_A и P_2 — полиномы, ограничивающие сложность алгоритмов A и A_2. Ясно, что последовательное применение алгоритмов A и A_2 определяет алгоритм A_1, решающий задачу Π_1 с оценкой времени работы на входе Z, $Z \in \Pi_1$, имеющей вид $P_2(P_A(|Z|))$, т. е. опять же полиномиальной (композиция двух полиномов дает полином).

Во-вторых, сводимость позволяет получать сравнительные оценки (качественного характера) для сложности решения дискретных задач типа «если $\Pi_1 \propto \Pi_2$, то задача Π_2 не проще, чем задача Π_1». Продемонстрируем это на конкретном примере.

В качестве задачи Π_1 возьмём задачу о выполнимости, которая формулируется так. Пусть $X = \{x_1, \ldots, x_n\}$ — множество булевых переменных. Элементарная дизъюнкция D над X определяется следующим образом:

$$D = x_{i_1}^{\sigma_1} \vee x_{i_2}^{\sigma_2} \vee \ldots \vee x_{i_r}^{\sigma_r}.$$

Пусть K есть конъюнкция некоторых элементарных дизъюнкций, называемая *конъюнктивной нормальной формой* (к. н. ф.):

$$K(x_1, \ldots, x_n) = D_1 D_2 \ldots D_m.$$

Считается, что K *выполнима*, если существует такой набор $\widetilde{\alpha} = (\alpha_1, \alpha_2, \ldots, \alpha_n)$ значений переменных $x_1, x_2, \ldots, x_n$, что $K(\widetilde{\alpha}) = 1$ (на наборе $\widetilde{\alpha}$ все дизъюнкции $D_1, \ldots, D_m$ обращаются в

единицу). Задача Π_1 заключается в том, чтобы по заданной к. н. ф. K выяснить, выполнима она или нет.

Если к. н. ф. K выполнима, т. е. $K(\widetilde{\alpha}) = 1$ для некоторого набора $\widetilde{\alpha} = (\alpha_1, \ldots, \alpha_n)$, то легко указать алгоритм, который за полиномиальное относительно $|K| = mn$ время может проверить выполнение равенства $K(\widetilde{\alpha}) = 1$. А это свидетельствует о том, что задача о выполнимости принадлежит классу NP.

В качестве задачи Π_2 возьмём задачу на покрытие и докажем следующее утверждение.

Теорема 36 (о сводимости задачи о выполнимости к задаче на покрытие). *Задача о выполнимости полиномиально сводится к задаче на покрытие.*

Доказательство. В соответствии с определением сводимости нужно указать сохраняющий ответы полиномиальный алгоритм, который по произвольной индивидуальной задаче о выполнимости строит индивидуальную задачу на покрытие (с указанием граничного значения l).

Пусть дана к. н. ф. $K(x_1, \ldots, x_n) = D_1 D_2 \cdot \ldots \cdot D_m$. По K строится таблица T_K (табл. 16) из $2n$ столбцов и $m + n$ строк. Столбцы этой таблицы обозначим символами $x_1, x_2, \ldots, x_n, \overline{x}_1, \overline{x}_2, \ldots$ $\ldots, \overline{x}_n$. Верхние n строк таблицы — вспомогательные. Нижние m строк соответствуют элементарным дизъюнкциям $D_1, \ldots, D_m$, которые и будут именами данных строк. Если x_j^σ $(j = 1, \ldots, n;$ $\sigma = 0, 1)$ входит в дизъюнкцию D_i, $i = 1, \ldots, m$, то в пересечении строки D_i и столбца x_j^σ ставится 1; в противном случае в указанной клетке матрицы ставится 0. Нижние m строк таблицы T_K, очевидно, полностью определяют заданную к. н. ф. K. Верхние n строк, как видно из табл. 16, полностью определяются уже одним числом n (эти строки фактически не содержат информации об индивидуальной к. н. ф. K).

Ниже (табл. 17) в качестве примера приведена таблица T_K для к. н. ф. $K = (x_1 \vee \overline{x}_2)(\overline{x}_1 \vee \overline{x}_3)(x_2 \vee x_3)$.

Нетрудно указать алгоритм, который строит таблицу T_K по к. н. ф. K за полиномиальное время относительно $|K| = mn$.

Отметим теперь, что i-я вспомогательная строка таблицы T_K покрывается лишь двумя столбцами x_i и $\overline{x}_i$, отвечающими i-й переменной, $i = 1, \ldots, n$. А это означает, что если какие-то n столбцов образуют покрытие всех вспомогательных строк, то для каждого i в покрытии присутствует либо столбец x_i, либо столбец $\overline{x}_i$, но не оба сразу (иначе какая-то переменная не будет участвовать в покрытии и соответствующая ей вспомогательная строка таблицы не будет покрыта). Отсюда следует, что любое

Таблица 16

	x_1	x_2	x_3	...	x_n	$\overline{x}_1$	$\overline{x}_2$	$\overline{x}_3$	...	$\overline{x}_n$
	1	0	0	...	0	1	0	0	...	0
	0	1	0	...	0	0	1	0	...	0
	0	0	1	...	0	0	0	1	...	0
	...	...	...	...	...	...	...	...	...	...
	0	0	0	...	1	0	0	0	...	1
D_1										
D_2										
⋮										
D_m										

Таблица 17

	x_1	x_2	x_3	$\overline{x}_1$	$\overline{x}_2$	$\overline{x}_3$
	1	0	0	1	0	0
	0	1	0	0	1	0
	0	0	1	0	0	1
D_1	1	0	0	0	1	0
D_2	0	0	0	1	0	1
D_3	0	1	1	0	0	0

покрытие таблицы T_K из не более, чем n столбцов (если оно существует) содержит n столбцов, отвечающих разным переменным, т. е. имеет вид $\{x_1^{\alpha_1}, x_2^{\alpha_2}, \ldots, x_n^{\alpha_n}\}$.

Рассмотрим построенную индивидуальную задачу на покрытие, в которой требуется ответить на вопрос: возможно ли покрыть таблицу T_K не более, чем n столбцами? Покажем, что ответы в этой индивидуальной задаче совпадают с ответами в задаче о выполнимости K.

В задаче на покрытие при ответе «да» покрытие существует и, как было установлено выше, имеет вид $\{x_1^{\alpha_1}, \ldots, x_n^{\alpha_n}\}$. Каждая строка D_j, $j \in \{1, \ldots, m\}$, покрывается некоторым столбцом $x_i^{\alpha_i}$ из покрытия, а это означает, что D_j содержит слагаемое $x_i^{\alpha_i}$ и $D_j = 1$ при $x_i = \alpha_i$. Следовательно, при $x_1 = \alpha_1$, ..., $x_n = \alpha_n$ получим $K(\alpha_1, \ldots, \alpha_n) = 1$, т. е. к. н. ф. K выполнима.

С другой стороны, если к. н. ф. K выполнима и $K(\alpha_1, \ldots, \alpha_n) = 1$, то, как нетрудно заметить, набор столбцов $\{x_1^{\alpha_1}, \ldots, x_n^{\alpha_n}\}$ образует покрытие таблицы T_K (в приведённом выше

примере так связаны между собой, скажем, «выполняющий» набор значений переменных $(0, 0, 1)$ и «покрывающий» набор столбцов $\{\overline{x}_1, \overline{x}_2, x_3\}$). Это означает, что при ответе «нет» в задаче на покрытие невозможен ответ «да» в соответствующей задаче о выполнимости. Теорема доказана.

Понятие полиномиальной сводимости задач позволяет очертить класс труднорешаемых дискретных задач, для которых пока ещё не найдены эффективные алгоритмы решения (и даже более того — остается открытым вопрос о самом существовании таких алгоритмов).

Задача П, $\Pi \in NP$, называется *NP-полной*, если к ней полиномиально сводится любая задача из класса NP.

Из этого определения следует, что если для некоторой задачи П установлена ее NP-полнота, то эта задача может быть отнесена к труднорешаемым, и если $P \neq NP$, то $\Pi \in NP \setminus P$ (заметим, что вопрос о том, совпадают ли классы P и NP или же $P \neq NP$, до сих пор остаётся открытым).

Непосредственное доказательство NP-полноты для какой-то дискретной задачи П из класса NP может оказаться, скорее всего, достаточно трудным и громоздким; впервые такое доказательство было получено для задачи о выполнимости. Но теперь для того, чтобы установить NP-полноту задачи П из NP, нет необходимости действовать только по определению NP-полной задачи и проводить «прямое» доказательство; достаточно для уже известной подходящей NP-полной задачи Π' установить отношение $\Pi' \propto \Pi$, из которого и будет следовать (с учётом транзитивности отношения $\propto$) NP-полнота задачи П. Именно таким образом расширяется список NP-полных задач, насчитывающий к настоящему времени свыше тысячи задач. К NP-полным задачам относятся, в частности, рассматривавшиеся выше задачи на покрытие и об упаковке в контейнеры.

ЗАДАЧИ

К главе I

1. Найти число размещений k одинаковых шаров по n ячейкам при условии:

а) в каждой ячейке находится не более одного шара;

б) в каждой ячейке может находиться более одного шара;

в) в каждой ячейке находится более одного шара.

2. Доказать равенства:

а) $\binom{n}{k}\binom{k}{r} = \binom{n-r}{k-r}\binom{n}{r}$;

б) $\binom{m}{h} = \binom{m-1}{h} + \binom{m-1}{h-1}$;

в) $\sum_{i=0}^{k} \binom{n-i-1}{k-i} = \binom{n}{k}$, $n > k$;

г) $\sum_{n=k}^{m} \binom{n}{k} = \binom{m+1}{k+1}$;

д) $\sum_{k=0}^{n} k\binom{n}{k} = n \cdot 2^{n-1}$;

е) $\sum_{k=0}^{n} \frac{1}{k+1}\binom{n}{k} = \frac{2^{n+1}-1}{n+1}$;

ж) $\sum_{k=0}^{n} \frac{(-1)^k}{k+1}\binom{n}{k} = \frac{1}{n+1}$;

з) $\sum_{i=0}^{k} \binom{n}{k-i}\binom{m}{i} = \binom{n+m}{k}$;

и) $\sum_{k=1}^{n} \frac{(-1)^{k-1}}{k}\binom{n}{k} = 1 + \frac{1}{2} + \ldots + \frac{1}{n}$.

3. Доказать, что:

а) $\binom{n}{k}$ возрастает по n при фиксированном k;

б) $\binom{n-i}{k-i}$ убывает по i при фиксированных n и k.

4. Найти максимальное значение $\binom{n}{k}$ при фиксированном n.

5. Найти минимальное значение суммы

$$\binom{n_1}{k}+\binom{n_2}{k}+\ldots+\binom{n_s}{k},$$

если $n_1+n_2+\ldots+n_s=n$ и n делится на s.

6. Показать, что число разбиений целого числа n на k частей равно числу разбиений n на части, наибольшая из которых равна k.

7. Пусть A — некоторое множество, B и C — его подмножества, а χ_B и χ_C — характеристические функции подмножеств B и C. Доказать равенства:

а) $\chi_{\overline{B}}=1-\chi_B$;

б) $\chi_{B\setminus C}=\chi_B-\chi_B\chi_C$;

в) $\chi_{B\cap C}=\chi_B\chi_C$;

г) $\chi_{B\cup C}=\chi_B+\chi_C-\chi_B\chi_C$;

д) $|B|=\sum_{x\in A}\chi_B(x)$.

8. Имеются m предметов и n различных свойств, которыми могут обладать эти предметы. Пусть m_r — число предметов, обладающих в точности r свойствами из n. Доказать, что

$$m_r=\sum_{k=0}^{n-r}(-1)^k\binom{k+r}{r}\sum_{1\leqslant i_1<\ldots<i_{k+r}\leqslant n}m(i_1,\ldots,i_{k+r}),$$

где $m(i_1,\ldots,i_{k+r})$ — число предметов, обладающих i_1-м, ..., i_{k+r}-м свойствами $(r\geqslant 1)$.

9. Найти производящую функцию для последовательности $\{a_n\}$, если:

а) $a_n=1$;

б) $a_n=\binom{n}{k}$;

в) $a_n=\binom{n+k-1}{k}$.

10. С помощью тождеств, связывающих производящие функции, вывести тождества для биномиальных коэффициентов:

а) $(1+x)^n(1-x)^n=(1-x^2)^n$,

$$\sum_{i=0}^{k}(-1)^i\binom{n}{k-i}\binom{n}{i}=\begin{cases}(-1)^{k/2}\binom{n}{k/2} & \text{при чётном } k,\\ 0 & \text{при нечётном } k;\end{cases}$$

б) $(1+x)^n(1+x)^{-m}=(1+x)^{n-m}$,

$$\sum_{i=0}^{k}(-1)^{k-i}\binom{n}{i}\binom{m+k-i-1}{k-i}=\binom{n-m}{k}.$$

11. Найти a_n по рекуррентным соотношениям и начальным условиям:

а) $a_{n+2}-5a_{n+1}+6a_n=0$, $a_0=10$, $a_1=26$;

б) $a_{n+4}-5a_{n+2}+4a_n=0$, $a_0=10$, $a_1=26$, $a_2=56$, $a_3=122$.

12. Найти n-й член последовательности Фибоначчи $\{F_n\}$, задаваемой рекуррентным соотношением $F_{n+2}=F_{n+1}+F_n$ и начальными условиями $F_1=F_2=1$.

К главе II

В задачах 1–13 рассматриваются неориентированные графы.

Степенью вершины графа называется число рёбер, инцидентных этой вершине.

1. Показать, что в любом графе без петель и кратных рёбер, содержащем не менее двух вершин, найдутся две вершины с одинаковыми степенями.

2. Пусть $\sigma(G)$ — наименьшая из степеней вершин графа G, не имеющего петель и кратных рёбер и содержащего n вершин ($n\geqslant 2$).

а) Доказать, что если $\sigma(G)\geqslant\dfrac{n-1}{2}$, то граф связен.

б) Показать, что в предыдущем утверждении нельзя заменить $\dfrac{n-1}{2}$ на $\left\lfloor\dfrac{n-1}{2}\right\rfloor$.

3. Доказать, что если в графе степень каждой вершины больше единицы, то в нём есть цикл.

4. Доказать, что каждое дерево с числом вершин $n\geqslant 2$ является двудольным графом.

5. Найти число попарно неизоморфных 4-вершинных графов без петель и кратных рёбер.

6. Доказать, что при любом натуральном $n \geqslant 3$ существует n-вершинный связный граф без петель и кратных рёбер, содержащий $n-1$ вершин с попарно различными степенями.

7. Доказать, что если в графе имеются ровно две вершины с нечётными степенями, то найдётся цепь, соединяющая эти вершины.

8. Доказать, что если в n-вершинном ($n \geqslant 2$) дереве нет вершин степени 2, то в нём содержится не меньше чем $\frac{n}{2}+1$ висячих вершин.

9. Корневое дерево D имеет k висячих вершин ($k \geqslant 2$), отличных от корня, и не имеет вершин степени 2, отличных от корня. Доказать, что общее число вершин в дереве не превосходит $2k$.

10. Доказать, что в связном графе любые две простые цепи максимальной длины имеют общую вершину. Верно ли, что у них всегда есть общее ребро?

11. Доказать, что любой граф (без петель и параллельных рёбер) допускает такую геометрическую реализацию в трёхмерном пространстве, при которой его рёбрам отвечают прямолинейные отрезки.

12. Доказать, что ранг матрицы инциденций связного графа с n вершинами (без петель) равен $n-1$ (подразумевается, что арифметические операции при вычислении определителей производятся по модулю 2).

Граф называется *эйлеровым*, если в нём имеется цикл, содержащий все вершины и все рёбра данного графа.

13. Доказать, что для связного графа следующие утверждения эквивалентны:

1) G — эйлеров граф;

2) каждая вершина графа G имеет чётную степень;

3) множество рёбер графа G можно разбить на простые циклы.

14. Найти число неизоморфных ориентированных графов, содержащих:

а) 2 вершины и 2 дуги;

б) 2 вершины и 3 дуги;

в) 3 вершины и 2 дуги;

15. Доказать, что если для каждой вершины ориентированного графа число исходящих из этой вершины дуг (обычно называемое полустепенью исхода этой вершины) положительно, то в нём существует ориентированный цикл. (Петля считается ориентированным циклом длины 1.)

К главе III

1. Найти число попарно различных булевых функций, получающихся из функции $\bigvee\limits_{1\leqslant i<j\leqslant n} x_i x_j$ подстановкой констант вместо переменных $x_1, \ldots, x_n$.

2. Найти число булевых наборов длины n, на которых функция $f(x_1, \ldots, x_n)$ обращается в единицу, если:

а) $f(x_1, \ldots, x_n) = \sum\limits_{k=1}^{n} \sum\limits_{1\leqslant i_1<\ldots<i_k\leqslant n} x_{i_1} \cdot \ldots \cdot x_{i_k}$;

б) $f(x_1, \ldots, x_n) = \sum\limits_{k=1}^{n} x_{i_1} \cdot \ldots \cdot x_{i_k}$, $i_a \neq i_b$ при $a \neq b$.

3. Множество A состоит из булевых функций, каждая из которых существенно зависит от всех своих переменных. Доказать, что всякая формула Φ над A, содержащая попарно различные переменные, каждая из которых входит в Φ ровно один раз, реализует булеву функцию, существенно зависящую от всех своих переменных.

4. Пусть $f(x_1, \ldots, x_n)$ — булева функция, существенно зависящая от всех своих n переменных. Доказать, что найдутся переменная x_i и булева константа σ ($\sigma \in \{0, 1\}$) такие, что функция $f(x_1, \ldots, x_{i-1}, \sigma, x_{i+1}, \ldots, x_n)$ существенно зависит от всех своих $n-1$ переменных.

Булева функция $f(x_1, \ldots, x_n)$ называется *симметрической*, если при любой перестановке переменных получается функция, равная $f(x_1, \ldots, x_n)$.

5. а) Показать, что любая булева функция от трёх переменных может быть получена из симметрической функции от семи переменных отождествлением переменных.

б) Указать взаимно однозначное соответствие между множеством булевых функций от трёх переменных и множеством симметрических функций от семи переменных.

в) Показать, что любая булева функция может быть получена из симметрической функции отождествлением переменных.

6. Известно, что булева функция $f(x_1, x_2, x_3)$ существенно зависит от каждой из трёх своих переменных. Доказать, что в этом случае найдутся переменная x_i ($i \in \{1, 2, 3\}$) и булева константа σ ($\sigma \in \{0, 1\}$), такие, что при подстановке в $f(x_1, x_2, x_3)$ константы σ вместо переменной x_i получается симметрическая функция, существенно зависящая от каждой из оставшихся двух переменных.

7. Можно ли функцию $x\&y$ реализовать формулой над множеством $\{x \to y\}$? (Другими словами, можно ли конъюнкцию выразить через импликацию?)

8. а) Доказать, что любую булеву функцию от n переменных, отличную от константы 0, можно представить в виде дизъюнктивной нормальной формы (д. н. ф.), содержащей не более 2^{n-1} элементарных конъюнкций вида $x_{i_1}^{\sigma_1} \cdot \ldots \cdot x_{i_k}^{\sigma_k}$ ($k \leqslant n$; $i_p \neq i_q$ при $p \neq q$).

б) Указать булеву функцию от n переменных, для которой любая д. н. ф., представляющая эту функцию, содержит не менее 2^{n-1} элементарных конъюнкций.

9. Доказать, что булева функция $f(x_1, \ldots, x_n)$ обращается в единицу на нечётном числе наборов значений своих переменных в том и только в том случае, когда полином Жегалкина этой функции содержит конъюнкцию максимальной длины $x_1 x_2 \ldots x_n$.

Пусть B — произвольное множество булевых функций. *Замыканием* множества B (обозначается $[B]$) называется множество всех тех булевых функций, каждая из которых может быть реализована формулой над B.

10. Доказать следующие свойства замыкания:
а) $B \subseteq [B]$;
б) $[B_1] \cup [B_2] \subseteq [B_1 \cup B_2]$;
в) $[B_1 \cap B_2] \subseteq [B_1] \cap [B_2]$;
г) $[B] = [[B]]$;
д) если $B_1 \subseteq B_2$, то $[B_1] \subseteq [B_2]$.

Булева функция $f(x_1, \ldots, x_n)$ *сохраняет константу* 0 (*константу* 1), если $f(0, \ldots, 0) = 0$ (соответственно если $f(1, \ldots, 1) = 1$). Множество всех булевых функций, сохраняющих константу 0 (константу 1), обозначается через T_0 (соответственно T_1).

Булева функция $f(x_1, \dots, x_n)$ считается: *самодвойственной*, если $f(x_1, \dots, x_n) = \overline{f}(\overline{x}_1, \dots, \overline{x}_n)$; *линейной*, если существуют такие булевы константы $\alpha_0, \alpha_1, \dots, \alpha_n$, что $f(x_1, \dots, x_n) = \alpha_0 \oplus \alpha_1 x_1 \oplus \dots \oplus \alpha_n x_n$; *монотонной*, если для любых двух наборов $\widetilde{\alpha} = (\alpha_1, \dots, \alpha_n)$ и $\widetilde{\beta} = (\beta_1, \dots, \beta_n)$ таких, что $\widetilde{\alpha} \leqslant \widetilde{\beta}$ (т. е. $\alpha_1 \leqslant \beta_1, \dots, \alpha_n \leqslant \beta_n$), выполняется неравенство $f(\widetilde{\alpha}) \leqslant f(\widetilde{\beta})$. Множества самодвойственных, линейных и монотонных булевых функций обозначаются соответственно через S, L и M.

В классах T_0, T_1, S, L, M выделим подмножества булевых функций, зависящих от n переменных $x_1, \dots, x_n$, и обозначим эти подмножества через T_0^n, T_1^n, S^n, L^n, M^n соответственно.

11. Доказать, что классы T_0, T_1, S, L, M замкнуты и отличны от P_2 (P_2 — множество всех булевых функций).

12. Найти число булевых функций в каждом из множеств T_0^n, T_1^n, S^n, L^n.

13. Найти число булевых функций $f(x_1, \dots, x_n)$, принадлежащих множеству:

а) $T_0^n \cup T_1^n \cup L^n$;

б) $T_1^n \cap S^n \cap L^n$;

в) $T_0^n \cup (L^n \cap M^n)$;

г) $(T_0^n \cup S^n) \setminus L^n$;

д) $(T_0^n \cup T_1^n \cup S^n) \cap L^n$;

е) $S^n \setminus (T_0^n \setminus L^n)$;

ж) $T_0^n \cup T_1^n \cup S^n \cup L^n$;

з) $(L^n \setminus S^n) \setminus M^n$.

14. Доказать, что если булева функция $f(x_1, \dots, x_n)$ не является самодвойственной, то, подставляя x и $\overline{x}$ вместо переменных $x_1, \dots, x_n$, можно получить константу (0 или 1).

15. Доказать, что если булева функция $f(x_1, \dots, x_n)$ — немонотонная, то, подставляя вместо её переменных функции 0, 1, x, можно получить функцию $\overline{x}$.

16. Доказать, что если булева функция $f(x_1, \dots, x_n)$ — нелинейная, то, подставляя вместо её переменных функции 0, 1, x, $\overline{x}$, y, $\overline{y}$, можно получить $x\&y$ или $\overline{x\&y}$.

Пусть A — замкнутый класс, а B — произвольное множество булевых функций из A. Множество B *полно в* A, если любая функция из A может быть реализована формулой над B; множество B считается *базисом в* A, если B полно в A, но любое собственное подмножество B полнотой в A не обладает.

17. Используя утверждения из задач 12, 15–17, доказать следующий критерий полноты (теорему Поста): *для того чтобы множество B булевых функций было полным в P_2, необходимо и достаточно, чтобы это множество не содержалось целиком ни в одном из классов T_0, T_1, S, L, M.*

18. Доказать, что если булева функция f монотонна и существенно зависит не менее, чем от двух переменных, то система $\{0, \overline{f}\}$ полна в P_2.

19. Выяснить, является ли множество B базисом в A, если:
а) $A = T_1$, $B = \{xy \sim z\}$;
б) $A = T_0$, $B = \{xy \vee z\}$;
в) $A = L$, $B = \{x \sim y, x \oplus y\}$;
г) $A = L$, $B = \{x_1 \oplus x_2 \oplus \ldots \oplus x_k, 1\}$, k — константа;
д) $A = T_0 \cap T_1$, $B = \{xy, xy \vee x\overline{z} \vee y\overline{z}\}$.

Булева функция f называется *шефферовой*, если любую булеву функцию можно выразить через f (т. е. если f образует базис в P_2).

20. Доказать, что функции $\overline{x_1 \& x_2}$ (*штрих Шеффера*; обозначается x_1/x_2) и $\overline{x_1 \vee x_2}$ (*стрелка Пирса*; обозначается $x_1 \downarrow x_2$) являются шефферовыми.

21. Выяснить, при каких n функция f является шефферовой:
а) $f(x_1, \ldots, x_n) = 1 \oplus x_1x_2 \oplus \ldots \oplus x_ix_{i+1} \oplus \ldots \oplus x_{n-1}x_n \oplus x_nx_1$;
б) $f(x_1, \ldots, x_n) = 1 \oplus x_1x_2 \oplus \ldots \oplus x_ix_{i+1} \oplus \ldots \oplus x_{n-1}x_n$;
в) $f(x_1, \ldots, x_n) = 1 \oplus \bigoplus\limits_{1 \leqslant i < j \leqslant n} x_ix_j$;
г) $f(x_1, \ldots, x_n) = 1 \oplus (x_1/x_2) \oplus \ldots \oplus (x_i/x_{i+1}) \oplus \ldots \oplus (x_{n-1}/x_n) \oplus (x_n/x_1)$;
д) $f(x_1, \ldots, x_n) = (x_1/x_2) \oplus \ldots \oplus (x_i/x_{i+1}) \oplus \ldots \oplus (x_{n-1}/x_n)$.

22. Найти число шефферовых функций, зависящих не более, чем от n переменных.

К главе IV

1. На множестве $M = \{0, 1, 2, 3, \ldots\}$ заданы предикаты:

$$S(x, y, z) = 1 \Leftrightarrow x + y = z,$$
$$P(x, y, z) = 1 \Leftrightarrow xy = z.$$

Записать формулы, задающие предикаты:
а) $x = 0$;

б) $x = 1$;
в) x — чётное число;
г) x — простое число;
д) $x = y$;
е) $x < y$;
ж) x делит y.

2. Существует ли интерпретация (модель), в которой формула Φ имеет значение 1:
а) $\Phi = (\forall x)(\forall y)(Q(x,x)\&\overline{Q}(x,y))$;
б) $\Phi = (\exists x)(\forall y)(Q(x,y) \to (\forall z)R(x,y,z))$;
в) $\Phi = (\forall x)(\exists y)\big(P(y) \to \big((P(x) \to Q(x)) \to ((Q(x) \to R(x)) \to$ $\to R(y))\big)\big)$?

3. Существует ли интерпретация, в которой формула Φ принимает значение 0:
а) $\Phi = (\exists x)P(x) \to (\forall x)P(x)$;
б) $\Phi = (\exists y)(\forall x)(P(x) \sim \overline{P}(y))$;
в) $\Phi = (\forall x)(\forall y)(\forall z)\,(P(x,x)\&\,((P(x,z) \to P(x,y)) \vee P(y,z))) \to$ $\to (\exists y)(\forall z)P(y,z)$?

4. Общезначима ли формула Φ:
а) $\Phi = (\forall x)(\exists y)P(x,y) \sim (\exists y)(\forall x)P(x,y)$;
б) $\Phi = (\exists x)(\forall y)P(x,y) \to (\forall y)(\exists x)P(x,y)$?

5. Для формулы Φ указать эквивалентную ей нормальную формулу:
а) $\Phi = (\forall x)(P(x) \to (\exists y)Q(x,y))$;
б) $\Phi = (\exists x)P(x) \vee (\forall y)\big(P(y) \to \overline{(\exists x)Q(x,y)}\big)$;
в) $\Phi = (\forall x)(P(x) \to Q(x,x)) \to ((\forall y)P(y) \sim (\exists y)Q(y,z))$;
г) $\Phi = (\exists x)(\forall y)P(x,y) \to (\exists x)(\forall z)Q(x,z)$.

6. Привести примеры формул: а) истинных в модели; б) истинных на множестве; в) тождественно истинных.

7. Привести примеры формул: а) эквивалентных (равносильных) в модели; б) эквивалентных на множестве; в) эквивалентных (равносильных).

8. Доказать эквивалентность формул F_1 и F_2:
а) $F_1 = (\forall x)(\forall y)P(x,y)$, $F_2 = (\forall y)(\forall x)P(x,y)$;
б) $F_1 = (\exists x)(\exists y)P(x,y)$, $F_2 = (\exists y)(\exists x)P(x,y)$.

9. Доказать, что формулы $(\exists x)(\forall y)P(x,y)$ и $(\forall y)(\exists x)P(x,y)$ неэквивалентны.

10. Множество M (т.е. предметная область) конечно. $P(x)$ — предикат, определённый на множестве M. Выразить операции навешивания кванторов через логические операции.

11. Указать предикаты P и Q такие, что:
а) $(\forall x)(P(x) \vee Q(x)) \neq (\forall x)P(x) \vee (\forall x)Q(x)$;
б) $(\exists x)(P(x)\&Q(x)) \neq (\exists x)P(x)\&(\exists x)Q(x)$;
в) $(\forall y)(\exists x)P(x,y) \to (\exists x)(\forall y)P(x,y) \not\equiv 1$.

К главе V

В главе V были определены:
а) схемы из функциональных элементов в базисе $\{\&, \vee, {}^-\}$;
б) функции, реализуемые схемами;
в) сложность $L(S)$ произвольной схемы S;
г) сложность $L(f)$ функции f;
д) функция Шеннона $L(n)$.

Если вместо базиса $\{\&, \vee, {}^-\}$ берётся какая-нибудь другая конечная система функций $F = \{f_1, \ldots, f_k\}$, то все перечисленные объекты и понятия вводятся совершенно аналогично тому, как это было сделано для $\{\&, \vee, {}^-\}$; небольшие и почти очевидные изменения в определениях, соответствующие рассматриваемому базису F, вносятся лишь на самом первом этапе — при определении схемы. Например, в случае схем в базисе $\{\&, \oplus, 1\}$ вершинам из V_2 приписывается символ 1 (константа 1), а каждой вершине из V_3 приписывается один из символов $\&$, $\oplus$ (одна из функций $x_1\&x_2$, $x_1 \oplus x_2$). Для схем в базисе $\{/\}$ множество вершин V_2 (из определения схемы) пусто, а каждой вершине из V_3 приписывается символ $/$ и т. п. В предлагающихся ниже задачах базис указывается нижним индексом (или, точнее, нижней системой индексов) при соответствующих обозначениях.

1. Доказать равенства:
а) $L_{\{\&,\vee,-\}}(1)=L_{\{\&,\vee,-\}}(0)=L_{\{\&,\vee,-\}}(x \to y)=L_{\{\&,\vee,-\}}(x/y)= = L_{\{\&,\vee,-\}}(\overline{xy}) = 2$;
б) $L_{\{\&,\vee,-\}}(x \oplus y) = L_{\{\&,\vee,-\}}(x \sim y) = 4$.

2. Доказать равенства:
а) $L_{\{\&,\oplus,1\}}(x \sim y) = 3$;
б) $L_{\{\&,\oplus,1\}}(\overline{xy}) = L_{\{\&,\oplus,1\}}(x/y) = L_{\{\&,\oplus,1\}}(x \vee y) = 3$;
в) $L_{\{\&,\oplus,1\}}(x \to y) = 4$.

3. Доказать равенства:
а) $L_{\{/\}}(1) = L_{\{/\}}(x \to y) = L_{\{/\}}(x\&y) = 2$;

б) $L_{\{/\}}(x \vee y) = L_{\{/\}}(0) = 3$;
в) $L_{\{/\}}(\overline{x\,y}) = L_{\{/\}}(x \oplus y) = 4$;
г) $L_{\{/\}}(x \sim y) = 5$.

4. Показать, что

$$L_{\{\&,-\}}(x_1 \vee \ldots \vee x_n) = L_{\{\vee,-\}}(x_1 \& \ldots \& x_n) = 2n.$$

5. Показать, что для любой булевой функции $f(x_1, \ldots, x_n)$ выполняются неравенства:
а) $L_{\{\&,-\}}(f(x_1, \ldots, x_n)) \leqslant 2L_{\{\&,\vee,-\}}(f(x_1, \ldots, x_n)) + n$;
б) $L_{\{\vee,-\}}(f(x_1, \ldots, x_n)) \leqslant 2L_{\{\&,\vee,-\}}(f(x_1, \ldots, x_n)) + n$;

6. Найти сложность реализации системы из всех симметрических булевых функций от трёх переменных схемами в базисе $\{\&, \vee, 1\}$.

Пусть $N_n = \{f_1, \ldots, f_n\}$, где $f_i = x_1 \oplus x_2 \oplus \ldots \oplus x_{i-1} \oplus x_{i+1} \oplus \oplus x_{i+2} \oplus \ldots \oplus x_n$, $i = 1, 2, \ldots, n$.

7. Показать, что:
а) $L_{\{\oplus\}}(N_3) = 3$;
б) $L_{\{\oplus\}}(N_4) = 6$;
в) $L_{\{\oplus\}}(N_n) = 2n - 2$ при $n \geqslant 4$
(здесь через $L_{\{\oplus\}}(N_n)$ обозначаем сложность реализации системы булевых функций N_n схемами из функциональных элементов в базисе $\{x_1 \oplus x_2\}$, т. е. полагаем $L_{\{\oplus\}}(N_n) = \min L(S)$, где минимум берётся по всем схемам в базисе $\{\oplus\}$, реализующим N_n, а $L(S)$ — число функциональных элементов $\oplus$ в схеме S).

Пусть $Q_{n,k}$ — множество всех булевых функций от n переменных $x_1, \ldots, x_n$, представимых полиномами Жегалкина степени не выше k.

8. Показать, что:
а) для любой функции $f \in Q_{n,2}$ справедливо неравенство

$$L_{\{\&,\oplus,1\}}(f) \lesssim \frac{n^2}{\log n};$$

б) для каждой функции $f \in Q_{n,3}$ справедливо неравенство

$$L_{\{\&,\oplus,1\}}(f) \lesssim \frac{n^3}{\log n};$$

в) для каждого $n = 2, 3, \ldots$ существует такая функция $f_n \in \in Q_{n,2}$, что

$$L_{\{\&,\oplus,1\}}(f_n) \asymp \frac{n^2}{\log n}$$

(запись $a(n) \asymp b(n)$ означает, что найдутся константы c_1 и c_2, для которых одновременно выполняются неравенства $a(n) \leqslant c_1 b(n)$ и $b(n) \leqslant c_2 a(n)$).

9. Показать, что в любом конечном полном базисе можно построить схему из функциональных элементов, реализующую все булевы функции от n переменных и содержащую $2^{2^n} - n$ элементов; доказать минимальность этой схемы.

10. В базисе из всех двуместных булевых функций построить схему из функциональных элементов, которая содержит не более, чем $5n - 3$ элементов и вычисляет арифметическую сумму двух n-разрядных двоичных чисел.

11. Пусть C_n есть $(n, \lceil \log(n+1) \rceil)$-оператор, т. е. упорядоченный набор из $\lceil \log(n+1) \rceil$ булевых функций $(f_1(x_1, \ldots, x_n), \ldots, f_{\lceil \log(n+1) \rceil}(x_1, \ldots, x_n))$, который на произвольном наборе $\tilde{\sigma} = (\sigma_1, \ldots, \sigma_n)$ значений переменных $x_1, \ldots, x_n$ выдаёт набор $\tilde{\beta}(\tilde{\sigma}) = (f_1(\tilde{\sigma}), \ldots, f_{\lceil \log(n+1) \rceil}(\tilde{\sigma}))$, являющийся двоичной записью числа единиц в наборе $\tilde{\sigma}$. Доказать, что

$$L_{\{\&, \vee, -\}}(C_n) \lesssim n$$

(запись $a(n) \lesssim b(n)$ означает, что найдется константа c, для которой справедливо неравенство $a(n) < cb(n)$).

12. Доказать, что для произвольной симметрической булевой функции $f(x_1, \ldots, x_n)$ от n переменных справедлива верхняя оценка сложности

$$L_{\{\&, \vee, -\}}(f) \lesssim n.$$

13. Показать, что для любых двух конечных полных в P_2 базисов Б_1 и Б_2 и всех булевых функций $f \in P_2$ справедливо соотношение

$$L_{\text{Б}_1}(f) \asymp L_{\text{Б}_2}(f).$$

14. Пусть K_n — множество всех элементарных конъюнкций $x_1^{\sigma_1} \cdot \ldots \cdot x_n^{\sigma_n}$ длины n, а D_n — множество всех элементарных дизъюнкций $x_1^{\sigma_1} \vee \ldots \vee x_n^{\sigma_n}$ длины n. Доказать цепочку соотношений

$$L_{\{\&, \ \}}(K_n \cup D_n) \sim L_{\{\vee, \ \}}(K_n \cup D_n) \sim 2^{n+1}.$$

К главе VI

1. Найти длину минимального теста матрицы M:

а) $M = \begin{bmatrix} 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \end{bmatrix}$;

б) $M = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \end{bmatrix}$.

2. Найти длину минимального теста булевой матрицы M, если:

а) матрица M составлена из всевозможных столбцов высоты n;

б) матрица M составлена из всех тех столбцов высоты n, в каждом из которых содержится ровно k единиц.

3. В схеме S, реализующей функцию голосования $h(x_1, x_2, x_3) = x_1x_2 \vee x_1x_3 \vee x_2x_3$ и изображённой на рис. 26, возможны произвольные константные неисправности на выходах элементов. Указать все попарно различные минимальные полные диагностические тесты для заданной схемы.

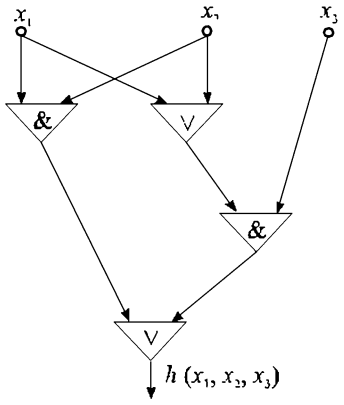


Рис. 26

4. Доказать, что в случае произвольных константных неисправностей на выходах элементов линейную булеву функцию $x_1 \oplus x_2 \oplus \ldots \oplus x_n$ можно реализовать:

а) схемой из функциональных элементов S_1 в базисе $\{\&, \vee, {}^-\}$, допускающей только тривиальный полный проверяющий тест (содержащий все 2^n входных наборов);

б) схемой S_2 в том же базисе $\{\&, \vee, {}^-\}$, допускающей полный проверяющий тест из четырёх наборов.

Для произвольной булевой функции $f(x_1, \ldots, x_n)$ через $Q(f)$ обозначим множество всех попарно различных функций, получающихся из $f(x_1, \ldots, x_n)$ подстановкой булевой константы (нуля или единицы) вместо какой-нибудь одной переменной. Всякое множество T_f булевых наборов длины n называется *единичным проверяющим тестом для функции* f, если для любой функции $g(\widetilde{x})$ из $Q(f)$ в T_f найдётся такой набор $\widetilde{\sigma}$, что $f(\widetilde{\sigma}) \neq g(\widetilde{\sigma})$. Всякое множество T_f^* булевых наборов длины n называется *единичным диагностическим тестом для функции* f, если T_f^* является единичным проверяющим тестом для f и, кроме того, для любых двух функций g и g' из $Q(f)$ в T_f^* найдётся такой набор $\widetilde{\sigma}$, что $g(\widetilde{\sigma}) \neq g'(\widetilde{\sigma})$.

5. Доказать, что для любой булевой функции от n переменных существует единичный диагностический тест, содержащий не более $2n$ наборов.

6. Указать булеву функцию от шести переменных, для которой любой единичный проверяющий тест содержит не менее восьми наборов.

7. Для произвольного натурального n указать булеву функцию от n переменных, для которой любой единичный проверяющий тест содержит не менее $2n - 2\lceil \log n \rceil$ наборов.

8. Указать булеву функцию от пяти переменных, для которой любой единичный диагностический тест содержит не менее десяти наборов.

К главе VII

Ниже рассматриваются функции из P_C, т. е. отображения вида $y = f(x)$, где $x = \{x(1), x(2), \ldots, x(i), \ldots\}$, $y = \{y(1), y(2), \ldots, y(i), \ldots\}$, а значения $x(i)$ и $y(i)$ принадлежат множеству $E_2 = \{0, 1\}$, $i = 1, 2, \ldots$.

1. Среди перечисленных ниже функций из P_C найти детерминированные и ограниченно-детерминированные функции:

а) $y(2t-1) = y(2t) = x(t)$, $t = 1, 2, \ldots$;

б) $y(t) = y(t+1) = x(t) \oplus x(t+1)$, $t = 1, 3, \ldots$;

в) $y(t) = \sum_{i=1}^{t} x(i)$, $t = 1, 2, \ldots$;

г) $y(t) = \begin{cases} 0 \text{ при } t = 1, \\ x(t-1), \text{ если } t \text{ — чётное число}, \\ \overline{x}(t) \vee y(t+1), \text{ если } t \text{ — нечётное число, большее двух;} \end{cases}$

д) $y(t) = \begin{cases} 0, \text{ если } \bigvee_{i=1}^{t+1} x(i) > \sum_{i=1}^{t+1} x(i), \\ 1 \text{ в противном случае;} \end{cases}$

е) $y(t) = \begin{cases} 0, \text{ если в наборе } (x(1), x(2), \ldots, x(t)) \\ \quad \text{нулей больше, чем единиц,} \\ 1 \text{ в противном случае.} \end{cases}$

2. Выяснить, являются ли перечисленные ниже функции ограниченно-детерминированными, и для каждой из ограниченно-детерминированных функций найти её вес:

а) $y(t) = \begin{cases} 1 \text{ при } t = 1, \\ \overline{x}(t) \text{ при } t \geqslant 2; \end{cases}$

б) $y(t) = \begin{cases} 1 \text{ при } t = 1, \\ x(t-1) \text{ при } t \geqslant 2; \end{cases}$

в) $y(t) = \begin{cases} x(1) \text{ при } t = 1, \\ x(t) \vee y(t-1) \text{ при } t \geqslant 2; \end{cases}$

г) $y(t) = \begin{cases} 0 \text{ при } t = 1, 2, \ldots, n, \\ x(t) \text{ при } t \geqslant n+1; \end{cases}$

д) $y(t) = \begin{cases} \overline{x}(t), \text{ если } t \text{ — нечётное число,} \\ x(t/2), \text{ если } t \text{ — чётное число;} \end{cases}$

е) $y(t) = \begin{cases} 0 \text{ при } t = 1, 2, \\ x(t-2) \text{ при } t \geqslant 3; \end{cases}$

ж) $y(t) = \begin{cases} 1, \text{ если } \sum_{i=1}^{t} x(i) = 10, \\ 0 \text{ в противном случае.} \end{cases}$

3. Построить диаграмму переходов, таблицу переходов, канонические уравнения и схему автомата в базисе $\{\&, \vee, \bar{}, z\}$ (содержащем функциональные элементы, реализующие $x\&y$, $x \vee y$, $\overline{x}$, и элемент единичной задержки z) для каждой из о.-д. функций:

а) $y(t) = \begin{cases} 0, \text{ если } t(\mathrm{mod}3) = 0, \\ 1, \text{ если } t(\mathrm{mod}3) = 1, \\ x(t), \text{ если } t(\mathrm{mod}3) = 2; \end{cases}$

б) $y(t) = \begin{cases} 0, \text{ если } t - \text{нечётное число}, \\ x(t), \text{ если } t = 4n + 2, \\ x(t-2), \text{ если } t = 4n, \text{ где } n = 0, 1, \ldots; \end{cases}$

в) $y(t) = \begin{cases} 1, \text{ если } t = 1, \\ 0, \text{ если } t = 2, \\ x(t-1), \text{ если } t \geqslant 3; \end{cases}$

г) $y(t) = \begin{cases} x(1), \text{ если } t - \text{нечётное число}, \\ x(2), \text{ если } t - \text{чётное число}; \end{cases}$

д) $y(t) = \begin{cases} 0, \text{ если } \sum\limits_{i=1}^{t} x(i)(\mathrm{mod}3) = 0, \\ 1, \text{ если } \sum\limits_{i=1}^{t} x(i)(\mathrm{mod}3) = 1, \\ x(t), \text{ если } \sum\limits_{i=1}^{t} x(i)(\mathrm{mod}3) = 2; \end{cases}$

е) $y(t) = \begin{cases} 0 \text{ при } t = 1, \\ x(t-1) \vee y(t-1) \text{ при } t \geqslant 2; \end{cases}$

ж) $y(t) = \begin{cases} 1, \text{ если } t = 1, \\ \overline{x}(t), \text{ если } x(t) = y(t-1), t \geqslant 2, \\ x(t-1), \text{ если } x(t) \neq y(t-1), t \geqslant 2; \end{cases}$

з) $y(t) = \begin{cases} \overline{x}(t) \text{ при } t = 1, 2, \\ \overline{x}(t) \vee y(t-1) \text{ при } t = 3, 4, \ldots. \end{cases}$

4. Построить схему автомата в базисе $\{\&, \vee, \bar{}, z\}$, реализующего упорядоченную пару о.-д. функций $y_1 = f_1(x)$, $y_2 = f_2(x)$, задаваемых следующим условием: $(y_1(t), y_2(t))$ есть двоичная запись числа $x(1) + \ldots + x(t)(\mathrm{mod}3)$, $t = 1, 2, \ldots$.

5. Построить диаграмму переходов и схему автомата в базисе $\{\&, \vee, {}^{-}, z\}$, реализующего о.-д. функцию $y = f(x_1, x_2)$, задаваемую условием

$$y(t) = \begin{cases} x_1(1) \text{ при } t = 1, \\ x_1(t) \oplus x_2(t-1) \text{ при } t = 2, 3, \ldots . \end{cases}$$

6. Построить канонические уравнения, таблицу переходов и диаграмму переходов для о.-д. функций, реализованных автоматами, представленными на схемах: а) рис. 27; б) рис. 28; в) рис. 29.

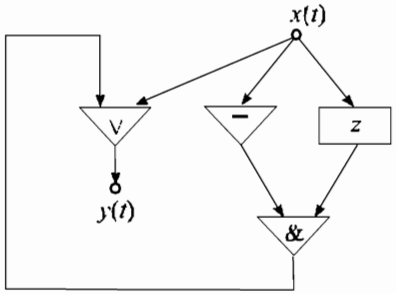


Рис. 27

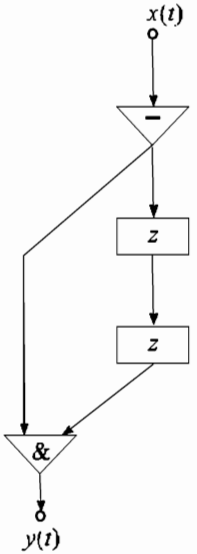


Рис. 28

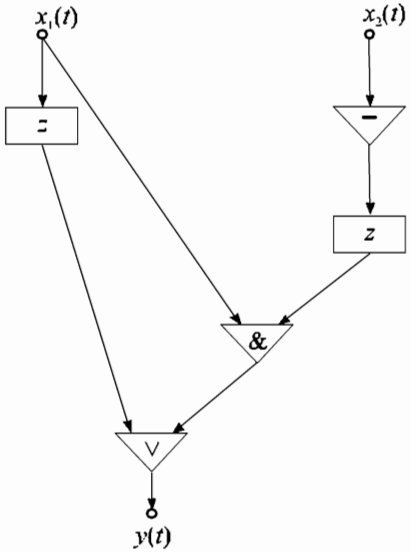


Рис. 29

К главе VIII

1. Выяснить, применима ли машина Тьюринга, задаваемая программой П, к слову $\widetilde{a}$. Если применима, то указать результат применения.

$$1)\quad \text{П}: \begin{cases} q_1 0 & \to q_1 0 \text{Л}, \\ q_1 1 & \to q_2 0 \text{Л}, \\ q_2 1 & \to q_1 0 \text{Л}, \\ q_2 0 & \to q_0 1 \text{С}, \end{cases}$$

а) $\widetilde{a} = 101^3$; б) $\widetilde{a} = 10^2 1^2$; в) $\widetilde{a} = 1^4$; г) $\widetilde{a} = 1^5$ (здесь и далее используется обозначение $a^m = \underbrace{aa \ldots a}_{m}$).

$$2)\quad \text{П}: \begin{cases} q_1 0 & \to q_2 1 \text{Л}, \\ q_1 1 & \to q_2 1 \text{Л}, \\ q_2 1 & \to q_1 1 \text{Л}, \\ q_2 0 & \to q_0 0 \text{С}, \end{cases}$$

а) $\widetilde{a} = 10^2 1^2$; б) $\widetilde{a} = 1^3 01^2$; в) $\widetilde{a} = 1^6$; г) $\widetilde{a} = 1^5$.

2. Построить машину Тьюринга, обладающую следующими свойствами:

а) машина имеет четыре команды, применима к словам $10^{2m}1$ ($m \geqslant 1$) и не применима к словам $10^{2m+1}1$ ($m \geqslant 0$);

б) машина имеет шесть команд, применима к словам 1^{3m} ($m \geqslant 1$) и не применима к словам 1^{3m+s} ($m \geqslant 0$ и $s = 1, 2$);

в) машина применима к словам $1^m 01^m$ ($m \geqslant 1$) и не применима к словам $1^m 01^s$ ($m \neq s$, $m \geqslant 1$ и $s \geqslant 1$).

Две машины Тьюринга *эквивалентны*, если они применимы к одним и тем же словам и на любом из этих слов результаты их работы совпадают. Предполагается, что если в начальной конфигурации на ленте записано непустое слово, то головка обозревает крайнюю правую клетку с непустым символом, т.е. с единицей для алфавита $A = \{0, 1\}$.

3. Выяснить, сколько существует неэквивалентных машин Тьюринга, программы которых содержат только по две команды вида $q_1 0 \to qaS$, $q_1 1 \to q'a'S'$, где $q, q' \in \{q_1, q_0\}$, $a, a' \in \{0, 1\}$, $S, S' \in \{\text{Л}, \text{П}, \text{С}\}$.

4. Построить машину Тьюринга, переводящую начальную конфигурацию K_1 в заключительную конфигурацию K_0:

а) $K_1 = 1^m q_1$, $K_0 = 1^m 01^m q_0$ ($m \geqslant 1$; в начальной и заключительной конфигурациях головка обозревает крайнюю правую клетку с единицей);
б) $K_1 = 1^m q_1$, $K_0 = (01)^m q_0$ ($m \geqslant 1$);
в) $K_1 = 1^m q_1$, $K_0 = 1^{2m} q_0$ ($m \geqslant 1$);
г) $K_1 = 1^m 01^s q_1$, $K_0 = 1^s 01^m q_0$ ($m, s \geqslant 1$).

5. Написать программы для машин Тьюринга, вычисляющих функции $x + y$, $x - y$, xy (x, y — натуральные числа).

6. Показать, что для всякой машины Тьюринга существует эквивалентная ей машина, в программе которой отсутствует символ C (т. е. головка в каждый момент времени обязательно сдвигается — либо влево, либо вправо).

7. По описанию машины Тьюринга написать её программу:
а) работая со словом $\ldots 00\,1^k\,00\ldots$ (из k записанных подряд единиц; $k \geqslant 1$), машина сдвигает его на две клетки влево, причём в заключительной конфигурации головка обозревает крайнюю правую единицу;
б) в процессе работы машины слово на ленте не изменяется, головка в каждый момент двигается влево и останавливается после прохождения первого же подслова из m следующих подряд нулей (m — заданное натуральное число).

8. Написать программу для машины Тьюринга, применимой к словам в алфавите $A = \{a_0, a_1, a_2\}$ вида $\ldots a_0 a_0 \widetilde{a} a_0 a_0 \ldots$, в которых подслово конечной длины $\widetilde{a}$ содержит только буквы a_1 и a_2 (отсутствующие вне $\widetilde{a}$); машина должна менять порядок букв в $\widetilde{a}$ на противоположный.

9. Пусть машина Тьюринга в каждый момент времени либо оставляет в обозреваемой клетке имеющийся там символ, либо записывает туда пустой символ. Доказать, что такая машина Тьюринга либо остановится, либо зациклится, либо с некоторого момента начнёт двигаться в одну сторону.

10. Построить машину Тьюринга, которая начальную конфигурацию

$$\ldots 00\,\overbrace{1\,1\ldots 1}^{m}\,q_1 00\ldots$$

переводит в заключительную конфигурацию

$$\ldots 0\,1\,0\,1\,0^2\,1\,0^3\,1\,0^4\ldots 0^{m-1}\,1\,0^m\,1^m\,q_0\,0\ldots .$$

К главе IX

1. Построить двоичный префиксный код $\sum$ с заданным набором $\widetilde{l}$ длин кодовых слов (элементарных кодов):

а) $\widetilde{l} = (1, 2, 3, 3)$;

б) $\widetilde{l} = (1, 3, 3, 4, 4, 4, 4)$;

в) $\widetilde{l} = (2, 2, 3, 3, 3, 4, 4)$;

2. Выяснить, может ли набор чисел $\widetilde{l}$ быть набором длин кодовых слов разделимого алфавитного кода в q-значном кодирующем алфавите:

а) $\widetilde{l} = (1, 2, 3, 3, 4)$, $q = 2$;

б) $\widetilde{l} = (1, 2, 2, 3, 3, 3)$, $q = 3$;

в) $\widetilde{l} = (1, 1, 2, 2, 3, 3, 3)$, $q = 3$;

г) $\widetilde{l} = (1, 1, 1, 2, 2, 2, 3, 3, 3, 3, 3)$, $q = 4$.

3. Алфавитный двоичный код $\sum(A)$ содержит более чем 2^l кодовых слов длины, не превышающей l. Выяснить, может ли код $\sum(A)$ быть:

а) взаимно-однозначным;

б) префиксным.

4. Используя теорему 30 (о редукции), построить двоичный код с минимальной избыточностью для набора вероятностей P:

а) $P = (0{,}6; 0{,}2; 0{,}1; 0{,}1)$;

б) $P = (0{,}4; 0{,}3; 0{,}1; 0{,}08; 0{,}08; 0{,}04)$;

в) $P = (0{,}3; 0{,}3; 0{,}1; 0{,}1; 0{,}04; 0{,}03; 0{,}03; 0{,}03; 0{,}03; 0{,}02; 0{,}02)$.

5. Указать набор вероятностей P, для которого существует оптимальный двоичный код Хаффмена с заданным набором $\widetilde{l}$ длин кодовых слов, и построить соответствующий код:

а) $\widetilde{l} = (1, 2, 4, 4, 4, 4)$;

б) $\widetilde{l} = (1, 3, 3, 4, 4, 4, 5, 5)$;

в) $\widetilde{l} = (2, 2, 3, 3, 3, 4, 5, 5)$;

6. Доказать, что в оптимальном двоичном коде Хаффмена число элементарных кодов максимальной длины чётно.

7. Выяснить, существует ли оптимальный двоичный код Хаффмена с заданным набором $\widetilde{l}$ длин кодовых слов:

а) $\widetilde{l} = (1, 3, 4, 4, 4)$;

б) $\widetilde{l} = (1, 2, 3, 4, 5)$;

в) $\widetilde{l} = (1, 2, 3, 4, 5, 5)$;

г) $\widetilde{l} = (1, 2, 3, 4, 4, 4, 4)$;
д) $\widetilde{l} = (1, 2, 3, 5, 5, 5, 5)$.

8. Рассматривается множество S оптимальных двоичных кодов Хаффмена для всевозможных наборов вероятностей $(p_1, \ldots, p_{15}, 2^{-20})$, удовлетворяющих условиям

$$p_1 \geqslant \ldots \geqslant p_{15} \geqslant 2^{-20}$$

и

$$p_1 + \ldots + p_{15} + 2^{-20} = 1.$$

Пусть $l(\sum)$ — наибольшая длина слова из кода $\sum$. Найти $\max\limits_{\sum \in S} l(\sum)$ и $\min\limits_{\sum \in S} l(\sum)$.

9. Построить код Хэмминга H_6.

10. Найти кодовое слово $\widetilde{\beta}$ из кода Хэмминга для сообщения $\widetilde{\alpha}$:

а) $\widetilde{\alpha} = 001$;
б) $\widetilde{\alpha} = 101$;
в) $\widetilde{\alpha} = 1000$;
г) $\widetilde{\alpha} = 1011$;
д) $\widetilde{\alpha} = 1110001$;
е) $\widetilde{\alpha} = 11011001$;
ж) $\widetilde{\alpha} = 0011010111$;
з) $\widetilde{\alpha} = 110101110010$.

11. По каналу связи, искажающему передаваемое слово не более, чем в одном разряде, было передано кодовое слово из кода Хэмминга, отвечающее сообщению $\widetilde{\alpha}$. Восстановить исходное сообщение $\widetilde{\alpha}$, если принято было сообщение $\widetilde{\beta}$:

а) $\widetilde{\beta} = 010011$;
б) $\widetilde{\beta} = 110001$;
в) $\widetilde{\beta} = 001100111$;
г) $\widetilde{\beta} = 101110111$;
д) $\widetilde{\beta} = 00011000101000$;
е) $\widetilde{\beta} = 00111000101000$.

К главе X

1. Найти минимальное остовное дерево для взвешенного графа, изображённого на:
а) рис. 30; б) рис. 31.

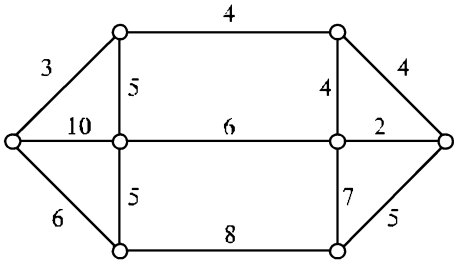


Рис. 30

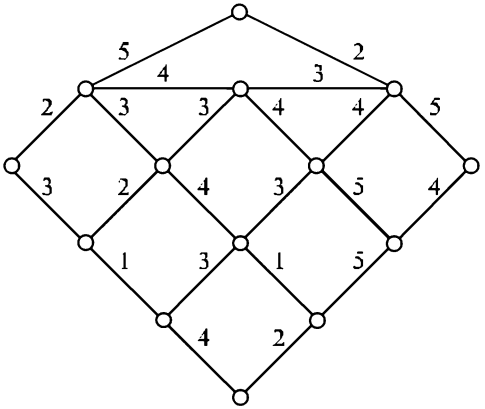


Рис. 31

2. Найти кратчайший путь между вершинами v и v' графа, изображённого на рис. 32.

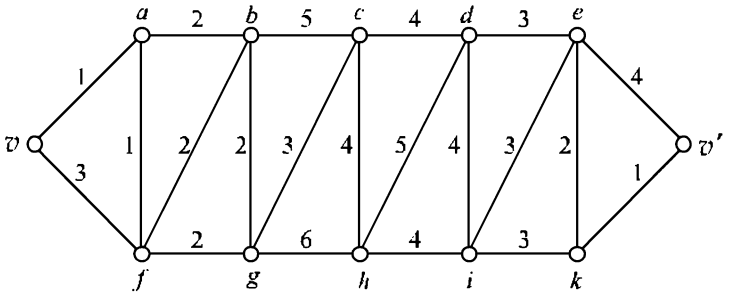


Рис. 32

3. Найти максимальный поток между полюсами α и β сети, изображённой на:

а) рис. 33; б) рис. 34.

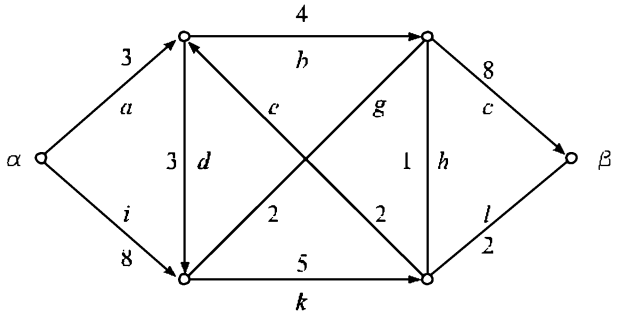


Рис. 33

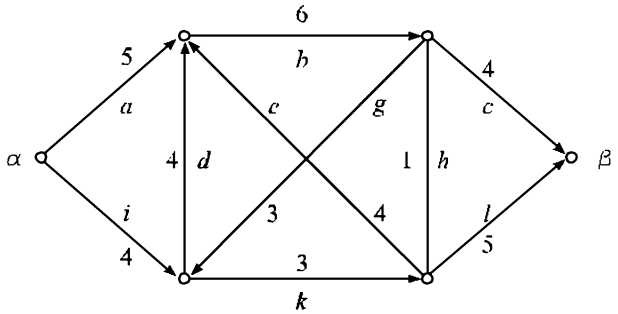


Рис. 34

4. Выделить две вершины в приведённом на рис. 35 графе так, чтобы при указанных пропускных способностях рёбер можно было бы создать поток наибольшей величины через получен-

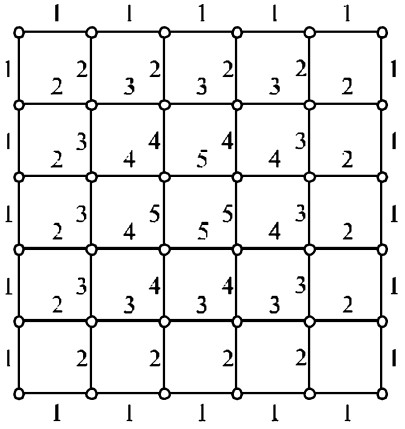


Рис. 35

ную сеть. Сколько попарно неизоморфных сетей, допускающих потоки максимальной величины, можно получить, выбирая разными способами вышеуказанную пару вершин (полюсов) сети?

Единичный n-мерный куб представляет собой граф с 2^n вершинами, занумерованными двоичными наборами, т.е. числами $\widetilde{\alpha} = (\alpha_1, \ldots, \alpha_n)$ длины n, в котором рёбрами соединены те и только те пары вершин, номера которых различаются ровно в одном разряде.

5. Указать максимальное (по числу рёбер) паросочетание для единичного n-мерного куба. Сколько рёбер содержит такое паросочетание?

6. На единичном n-мерном кубе произвольным образом выбраны две вершины. Найти величину максимального потока в полученной двухполюсной сети при условии, что пропускная способность каждого ребра равна единице.

Если вершине v единичного n-мерного куба приписан набор $\widetilde{\alpha} = (\alpha_1, \ldots, \alpha_n)$, то номером этой вершины будет число $N(v) = \sum\limits_{i=1}^{n} \alpha_i 2^{n-i}$, т.е. число, двоичной записью которого является набор $\widetilde{\alpha}$.

7. Найти минимальное остовное дерево и его вес для единичного n-мерного куба, если вес ребра, соединяющего вершины v и v', равен:

а) $\min(N(v), N(v'))$;

б) $\max(N(v), N(v'))$.

Если вершине v единичного n-мерного куба приписан набор $\widetilde{\alpha} = (\alpha_1, \ldots, \alpha_n)$, то весом данной вершины считаем число $w(v) = \sum_{i=1}^{n} \alpha_i$.

8. Найти вес минимального остовного дерева для единичного n-мерного куба, если вес ребра, соединяющего вершины v и v', равен:

а) $\min(w(v), w(v'))$;

б) $\max(w(v), w(v'))$.

ОТВЕТЫ, УКАЗАНИЯ, РЕШЕНИЯ

К главе I

1. а) $\binom{n}{k}, k \leqslant n$. б) H_n^k. в) $H_n^{k-2n}, k \geqslant 2n$.

2. в) Воспользовавшись равенствами $\binom{n-k-1}{0} = \binom{n-k}{0}$ и $\binom{m}{h} = \binom{m-1}{h} + \binom{m-1}{h-1}$, получаем

$$\sum_{i=0}^{k} \binom{n-i-1}{k-i} =$$

$$= \binom{n-1}{k} + \binom{n-2}{k-1} + \ldots + \binom{n-k+1}{2} + \binom{n-k}{1} + \binom{n-k-1}{0} =$$

$$= \binom{n-1}{k} + \binom{n-2}{k-1} + \ldots + \binom{n-k+1}{2} + \binom{n-k}{1} + \binom{n-k}{0} =$$

$$= \binom{n-1}{k} + \binom{n-2}{k-1} + \ldots + \binom{n-k+1}{2} + \binom{n-k+1}{1} =$$

$$= \binom{n-1}{k} + \binom{n-2}{k-1} + \ldots + \binom{n-k+2}{2} = \ldots =$$

$$= \binom{n-1}{k} + \binom{n-1}{k-1} = \binom{n}{k}.$$

г) *Указание.* Провести индукцию по m. (При $m = k$ равенство очевидно. Пусть оно справедливо при $m = k, k+1, \ldots, h$. Для $m = h+1$ получаем

$$\sum_{n=k}^{h+1} \binom{n}{k} = \sum_{n=k}^{h} \binom{n}{k} + \binom{h+1}{k} = \left\{ \begin{array}{l} \text{по предположению} \\ \text{индукции} \end{array} \right\} =$$

$$= \binom{h+1}{k+1} + \binom{h+1}{k} = \{\text{с учётом равенства б)}\} = \binom{h+2}{k+1}.)$$

д) *Указание.* Продифференцировать тождество $(1+t)^n = \sum_{k=0}^{n} \binom{n}{k} t^k$ по t и положить $t=1$.

е) *Указание.* Проинтегрировать тождество $(1+t)^n = \sum_{k=0}^{n} \binom{n}{k} t^k$ по t от 0 до 1.

ж) *Указание.* Проинтегрировать тождество $(1-t)^n = \sum_{k=0}^{n} (-1)^k \binom{n}{k} t^k$ по t от 0 до 1.

з) *Указание.* Сравнить коэффициенты при t^k в левой и правой частях тождества $(1+t)^n(1+t)^m = (1+t)^{n+m}$.

и) *Указание.* Провести индукцию по n с использованием равенств б) и ж). (При $n=1$ равенство очевидно. Пусть оно выполняется при $n = 1, 2, \ldots, h$. Для $n = h+1$ получаем

$$\sum_{k=1}^{h+1} \frac{(-1)^{k-1}}{k} \binom{h+1}{k} = \sum_{k=1}^{h} \frac{(-1)^{k-1}}{k} \binom{h+1}{k} + \frac{(-1)^k}{h+1} =$$

$$= \{\text{используя б)}\} = \sum_{k=1}^{h} \frac{(-1)^{k-1}}{k} \left(\binom{h}{k} + \binom{h}{k-1} \right) + \frac{(-1)^k}{h+1} =$$

$$= \sum_{k=1}^{h} \frac{(-1)^{k-1}}{k} \binom{h}{k} + \sum_{k=0}^{h} \frac{(-1)^k}{k+1} \binom{h}{k} =$$

$$= \{\text{используя предположение индукции и равенство ж)}\} =$$

$$= 1 + \frac{1}{2} + \ldots + \frac{1}{h} + \frac{1}{h+1}.)$$

3. б) Видно из отношения $\binom{n-i}{k-i} \Big/ \binom{n-(i+1)}{k-(i+1)}$, которое равно $\dfrac{n-i}{k-i}$.

4. $\binom{n}{\lfloor n/2 \rfloor}$. (Рассмотреть соотношение $\binom{n}{k} \Big/ \binom{n}{k-1} = \dfrac{n-k+1}{k}$; оно больше 1 при $k \leqslant \lfloor n/2 \rfloor$ и меньше 1 при $k > \lceil n/2 \rceil$.)

5. Минимальное значение суммы достигается при $n_1 = n_2 = \ldots = n_s = \dfrac{n}{s}$ и равно $s\binom{n/s}{k}$ (если бы это было не так, то нашлись бы два биномиальных коэффициента $\binom{n_i}{k}$ и $\binom{n_j}{k}$ такие, что $n_i > \dfrac{n}{s}$, $n_j < \dfrac{n}{s}$, $n_i - n_j > 1$, и тогда, заменив n_i на $n_i - 1$, а n_j на $n_j + 1$, можно было бы, как нетрудно проверить, получить сумму, меньшую, чем исходная).

6. Пусть $n_1, n_2, \ldots, n_k$ — разбиение n на части; можно считать, что $n_1 \geqslant n_2 \geqslant \ldots \geqslant n_k \geqslant 1$ (разбиения, отличающиеся лишь порядком следования слагаемых, не различаются). Рассмотрим n точек в k строках так, чтобы i-я строка содержала n_i точек; получим диаграмму (граф Феррера) разбиения $n = n_1 + \ldots + n_k$. Каждому разбиению числа n на k частей поставим в соответствие разбиение этого числа на части, которое получается «транспонированием» графа Феррера исходного разбиения; при этом строки становятся столбцами, а столбцы — строками (пример приведён на рис. 36). Легко заметить, что такое соответствие задаёт биекцию между разбиениями n на k частей и разбиениями на части, наибольшая из которых равна k.

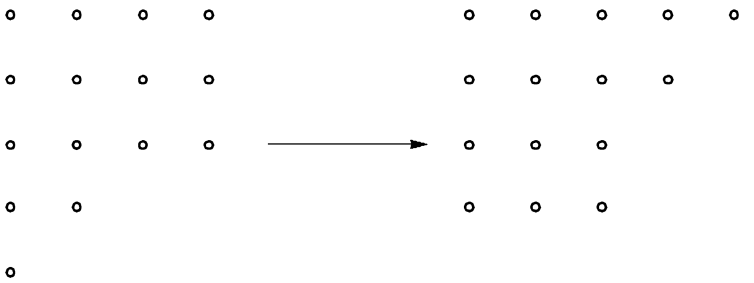


Рис. 36

7. Приведённые равенства следуют из определений характеристической функции и соответствующих операций над множествами.

8. Убедимся, что каждый предмет, обладающий r свойствами, будет учтён в приведённой формуле ровно один раз, а все другие — ни разу. Действительно, элементы, обладающие s свойствами, при $s < r$, очевидно, не учитываются. Элемент, обладающий заданными $s = r + t$ свойствами, будет учитываться во внутренней сумме $\binom{r+t}{r+k}$ раз. Выделяя подсумму, отвечаю-

щую такому одному элементу, используя результат задачи 2. а) и учитывая, что $s = r + t \leqslant n$, т. е. $n - r \geqslant t$, получаем

$$\sum_{k=0}^{n-r}(-1)^k\binom{k+r}{r}\binom{r+t}{r+k} = \sum_{k=0}^{n-r}(-1)^k\binom{r+t}{r}\binom{t}{k} =$$

$$= \binom{r+t}{r}\sum_{k=0}^{t}(-1)^k\binom{t}{k} = \begin{cases} 1 & \text{при } t = 0, \\ 0 & \text{при } t > 0. \end{cases}$$

Таким образом, элементы, обладающие в точности m свойствами, учитываются в формуле ровно по одному разу, а остальные не учитываются.

9. а) $\dfrac{2}{1-t}$.

б) $A(t) = \sum\limits_{n=0}^{\infty} a_n t^n = \sum\limits_{n=0}^{\infty}\binom{n}{k}t^n = \sum\limits_{n=k}^{\infty}\binom{n}{k}t^n = t^k\sum\limits_{n=0}^{\infty}\binom{k+n}{n}t^n =$

$$= t^k\left(1 + (k+1)t + \ldots + \frac{(k+n)\cdot\ldots\cdot(k+1)}{n!}t^n + \ldots\right) =$$

$$= \frac{t^k}{(1-t)^{k+1}}.$$

в) $A(t) = \sum\limits_{n=0}^{\infty}\binom{n+k-1}{k}t^n = \sum\limits_{n=1}^{\infty}\binom{n+k-1}{k}t^n =$

$$= t\sum_{n=1}^{\infty}\binom{n+k-1}{k}t^{n-1} = t\sum_{n=0}^{\infty}\binom{k+n}{k}t^n =$$

$$= t\sum_{n=0}^{\infty}\binom{k+n}{n}t^n = \{\text{см. б)}\} = \frac{t}{(1-t)^{k+1}}.$$

10. а), б) Сравнить коэффициенты при x^k.

11. а) $a_n = 6\cdot 2^n + 4\cdot 3^n$. б) $a_n = -\dfrac{17}{3} + \dfrac{(-1)^n}{3} + \dfrac{47}{3}\cdot 2^n - \dfrac{(-2)^n}{3}$.

12. $F_n = \dfrac{1}{\sqrt{5}}\left(\left(\dfrac{1+\sqrt{5}}{2}\right)^n - \left(\dfrac{1-\sqrt{5}}{2}\right)^n\right)$.

К главе II

1. Пусть $s_1, \ldots, s_n$ — степени вершин графа Γ с n вершинами; очевидно, $s_1, \ldots, s_n \in \{0, 1, \ldots, n-1\}$, поскольку каждой вершине инцидентны не более чем $n-1$ рёбер. Если степени всех вершин различны, то $\{s_1, \ldots, s_n\} = \{0, 1, \ldots, n-1\}$, а это означает, что в Γ одновременно имеются изолированная вершина и вершина степени $n-1$, соединённая рёбрами со всеми остальными вершинами; получаем противоречие.

2. а) Пусть V — множество вершин графа G, $N(v)$ — подмножество вершин из V, смежных с v, и $N'(v) = \{v\} \bigcup N(v)$. Из условия $\sigma(G) \geqslant \dfrac{n-1}{2}$ следует $\left|N'(v)\right| \geqslant \dfrac{n+1}{2}$, откуда, в свою очередь, вытекает неравенство $\left|V \backslash N'(v)\right| \leqslant \dfrac{n-1}{2}$. Из последнего неравенства получаем, что каждая вершина из $V \backslash N'(v)$ смежна с некоторой вершиной из $N'(v)$, а G — связный граф.

б) Очевидно для графа из двух изолированных вершин. Нетрудно также заметить, что указанную замену нельзя сделать для графа G, состоящего из двух связных компонент, каждая из которых представляет собой полный граф с m вершинами (полный граф — это граф, в котором любые две вершины смежны).

3. Пусть в графе G степень каждой вершины больше единицы. Отметим в G какую-нибудь вершину v и пройдём по инцидентному v ребру e в какую-нибудь смежную с v вершину v', которую также отметим. По условию вершине v' инцидентно хотя бы одно какое-то ребро e', отличное от e. По ребру e' из v' пройдём в вершину v'', смежную с v', и т. д. После некоторого очередного шага мы вернёмся (в силу конечности графа G) в одну из ранее пройденных и отмеченных вершин; пройденные вершины и ребра дадут цикл.

4. *Указание.* Воспользовавшись наличием в дереве концевых вершин, провести индукцию по числу рёбер.

5. 11.

6. Применим индукцию по числу вершин. Имеется граф с тремя вершинами и двумя рёбрами, у которого две вершины степени 1 и одна вершина степени 2. Пусть G — n-вершинный связный граф без петель и кратных рёбер, содержащий $n-1$ вершин $v_1, \ldots, v_{n-1}$ с попарно различными степенями. Поскольку степень каждой вершины не меньше единицы (в силу связности графа) и не больше $n-1$, то можно считать, что степени вершин $v_1, \ldots, v_{n-1}$ равны соответственно $1, \ldots, n-1$. Послед-

няя вершина v_n имеет некоторую степень i, $i \in \{1, \dots, n-1\}$. Добавим к G вершину v_{n+1}, соединив её рёбрами с вершинами $v_{i+1}, \dots, v_{n-1}, v_n$. Получим требуемый связный граф, в котором вершины $v_1, \dots, v_i$, v_n, $v_{i+1}, \dots, v_{n-1}$ имеют соответственно степени $1, \dots, i, i+1, i+2, \dots, n$.

7. Пусть в графе G только две вершины v и v' с нечётными степенями. Отметим какое-нибудь ребро e_1, инцидентное вершине v, и перейдём по этому ребру в некоторую смежную с v вершину v_1. Отметим, что число неотмеченных рёбер, инцидентных вершине v, окажется после этого чётным. Если v_1 совпадает с v', то искомая цепь найдена. Если v_1 отличается от v', то число ещё не отмеченных ребёр, инцидентных вершине v_1, окажется нечётным; пусть e_2 — одно из этих рёбер. Ребро e_2 отметим и перейдём по нему в некоторую вершину v_2, и т. д. После некоторого очередного перехода попадём в вершину v'; отмеченные рёбра (их число не превосходит общего числа рёбер в G) составляют искомый путь.

8. Пусть в n-вершинном дереве нет вершин степени 2, а n_i — число вершин степени i ($i = 1, 3, 4, \dots, n-1$; число рёбер в дереве на единицу меньше числа вершин). Любое ребро дерева увеличивает степень каждой из двух инцидентных ему вершин на единицу, и поэтому выполняется равенство $n_1 + \sum\limits_{i=3}^{n-1} i n_i = 2(n-1)$, из которого вытекает $n_1 + 3\sum\limits_{i=3}^{n-1} n_i \leqslant 2(n-1)$. Отсюда, учитывая, что $\sum\limits_{i=3}^{n-1} n_i = n - n_1$, получаем $n_1 + 3(n - n_1) \leqslant 2(n-1)$ и $n_1 \geqslant \dfrac{n}{2} + 1$.

9. Пусть $N(\mathcal{D})$ означает общее число вершин корневого дерева $\mathcal{D}$, а $N'(\mathcal{D})$ — число висячих вершин этого дерева, отличных от корня. Проведём индукцию по числу рёбер в рассматриваемых деревьях. Для корневого дерева $\mathcal{D}$ с двумя рёбрами (не имеющего вершин степени 2, отличных от корня), очевидно, $N(\mathcal{D}) < 2N'(\mathcal{D})$. Пусть $N(\mathcal{D}) \leqslant N'(\mathcal{D})$ для всех рассматриваемых деревьев, содержащих $2, \dots, n$ рёбер; докажем это неравенство для корневого дерева $\mathcal{D}$ с $n+1$ рёбрами. Рассмотрим два случая: 1) корню v^* дерева $\mathcal{D}$ инцидентны некоторые рёбра $(v^*, v_1), \dots, (v^*, v_k)$, $k \geqslant 2$; 2) корню v^* инцидентно ровно одно ребро (v^*, v_1).

1) Если все рёбра $(v^*, v_1), \dots, (v^*, v_k)$ — висячие, то, очевидно, $N(\mathcal{D}) < 2N'(\mathcal{D})$. Пусть теперь к рёбрам $(v^*, v_1), \dots, (v^*, v_i)$

«подвешены» соответственно непустые поддеревья $\mathcal{D}_1, \ldots, \mathcal{D}_i$, $1 \leqslant i \leqslant k$; поскольку $\mathcal{D}$ не имеет вершин степени 2, отличных от v^*, то каждое из поддеревьев $\mathcal{D}_1, \ldots, \mathcal{D}_i$ имеет не менее двух рёбер и не менее двух висячих вершин. Ребро (v^*, v_j) вместе с «подвешенным» к нему деревом $\mathcal{D}_j$ обозначим через $\mathcal{D}_j^*$, $j = 1, \ldots, i$. Согласно предположению индукции $N(\mathcal{D}_j^*) \leqslant 2N'(\mathcal{D}_j^*)$, $j = 1, \ldots, i$ и $\sum\limits_{j=1}^{i} N(\mathcal{D}_j^*) \leqslant 2\sum\limits_{j=1}^{i} N'(\mathcal{D}_j^*)$; добавив к левой части неравенства число $k - i$, а к правой — число $2(k-i)$, получаем $\sum\limits_{j=1}^{i} N(\mathcal{D}_j^*) + k - i \leqslant 2N'(\mathcal{D})$. Поскольку $i \geqslant 1$, то корень v^* в $\sum\limits_{j=1}^{i} N(\mathcal{D}_j^*)$ хотя бы один раз учитывается и потому $N(\mathcal{D}) \leqslant \sum\limits_{j=1}^{i} N(\mathcal{D}_j^*) + k - i \leqslant 2N'(\mathcal{D})$, т. е. требуемое неравенство выполняется.

2) К ребру (v^*, v_0) в вершине v_0 «подвешены» некоторые рёбра $(v_0, v_1), \ldots, (v_0, v_k)$, $k \geqslant 2$. Если все эти рёбра — висячие, то требуемое неравенство, очевидно, выполняется. Пусть к рёбрам $(v_0, v_1), \ldots, (v_0, v_i)$ в вершинах $v_1, \ldots, v_i$ подвешены соответственно поддеревья $\mathcal{D}_1, \ldots, \mathcal{D}_i$, которые вместе с рёбрами $(v_0, v_1), \ldots, (v_0, v_i)$ образуют соответственно деревья $\mathcal{D}_1^*, \ldots, \mathcal{D}_i^*$, $i \in \{1, \ldots, k\}$; рёбра $(v_0, v_{i+1}), \ldots, (v_0, v_k)$ — висячие. Каждое из деревьев $\mathcal{D}_j$ содержит не менее двух рёбер, среди которых не менее двух висячих, $1 \leqslant j \leqslant i$. Считая v_0 корнем каждого из деревьев $\mathcal{D}_1^*, \ldots, \mathcal{D}_i^*$ и используя предположение индукции, получаем неравенства $N(\mathcal{D}_j^*) \leqslant 2N'(\mathcal{D}_j^*)$, $j = 1, \ldots, i$, и $\sum\limits_{j=1}^{i} N(\mathcal{D}_j^*) \leqslant 2\sum\limits_{j=1}^{i} N'(\mathcal{D}_j^*)$. Добавив к левой и правой частям неравенства $2(k-i)$, получим $\sum\limits_{j=1}^{i} N(\mathcal{D}_j^*) + 2(k-j) \leqslant 2N'(\mathcal{D})$. В левой части неравенства имеем сумму, в которой вершина v_0 учитывается i раз, каждая из вершин $v_{i+1}, \ldots, v_k$ учитывается дважды, корень v^* не учитывается ни разу, а все остальные вершины дерева $\mathcal{D}$ учитываются по одному разу; но $k \geqslant 2$ и потому рассматриваемая сумма не меньше числа вершин в дереве $\mathcal{D}$, т. е. требуемое неравенство выполняется и в этом случае.

10. Пусть в связном графе G имеются какие-то две простые цепи Z_1 и Z_2 максимальной длины l. Предположим, что у этих цепей нет общей вершины. Возьмём две произвольные вершины a и b, принадлежащие соответственно цепям Z_1 и Z_2, и соединим эти вершины какой-нибудь простой цепью Z_3. В Z_3 выделим отрезок Z, соединяющий вершину c цепи Z_1 с вершиной d цепи Z_2 и не содержащий иных вершин цепей Z_1, Z_2 (может оказаться, что c совпадает с a, d совпадает с b). Поскольку Z_1 и Z_2 не имеют общих вершин, то длина l' отрезка Z больше нуля. Вершина c разбивает Z_1 на две части, одна из которых, скажем Z_1', имеет длину, не меньшую чем $\frac{l}{2}$; точно так же d разбивает Z_2 на две части, одна из которых, скажем Z_2', имеет длину, также не меньшую чем $\frac{l}{2}$. Из Z_1', Z, Z_2' составим цепь, длина которой окажется больше l, что противоречит исходному предположению о максимальной длине цепей Z_1 и Z_2.

Две простые цепи максимальной длины могут и не иметь общего ребра (рис. 37).

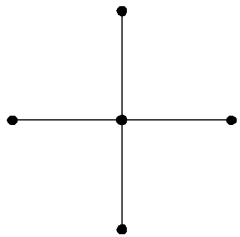


Рис. 37

11. Индукцией по числу вершин докажем, что каждый из рассматриваемых графов допускает геометрическую реализацию, в которой никакие три вершины не лежат на одной прямой и никакие четыре вершины не лежат в одной плоскости. Для графов, содержащих не более трёх вершин, это очевидно. Допустим, что утверждение справедливо для графов, содержащих не более m вершин; докажем его для графа G с $(m+1)$ вершинами. В G выделим подграф G', содержащий какие-нибудь m вершин $v_1, \ldots, v_m$ и все рёбра, инцидентные только этим вершинам. Воспользовавшись предположением индукции, построим для G' требуемую геометрическую реализацию Γ'. Все вершины в Γ' сгруппируем всевозможными способами в пары и в тройки; единственным образом через каждую пару проведём прямую, а через каждую тройку — плоскость. Требуемую геометрическую

реализацию для G получаем, добавив к Γ' ещё одну вершину v_{m+1}, лежащую вне проведённых прямых и плоскостей, и соединив v_{m+1} отрезками прямых со всеми вершинами, соседними с v_{m+1} (в графе G). Действительно, рёбра в Γ' не пересекаются по предположению индукции. Допустим, что некоторое ребро (v_{m+1}, v_i) пересекается с ребром (v_j, v_k) (вершины v_i, v_j, v_k, v_{m+1} попарно различны). В таком случае через данные рёбра можно было бы провести плоскость P, которая совпала бы с плоскостью, ранее проведённой через v_i, v_j, v_k, а это противоречит выбору v_{m+1}.

12. Пусть M — $(n \times m)$-матрица инциденций связного графа G с n вершинами и m рёбрами. В каждом столбце матрицы M ровно две единицы, поэтому сумма по модулю 2 всех строк даёт нулевую строку, что свидетельствует о линейной зависимости строк матрицы M; следовательно, ранг матрицы M не превосходит $n-1$. Возьмём теперь какое-нибудь остовное дерево $\mathcal{D}$ графа G. Выделим в $\mathcal{D}$ какую-нибудь висячую вершину v_1, инцидентную висячему ребру e_1; удалив v_1 и e_1, получим дерево $\mathcal{D}_1$ (отличный от v_1 конец ребра e_1 остаётся в $\mathcal{D}_1$). В $\mathcal{D}_1$ снова выделим висячую вершину v_2, инцидентную висячему ребру e_2; удалив v_2 и e_2, получим дерево $\mathcal{D}_2$. Продолжая этот процесс, получим упорядоченные последовательности вершин $v_1, v_2, \ldots, v_{n-1}, v_n$ (вершина v_n остаётся после удаления v_{n-1} и e_{n-1}), рёбер $e_1, e_2, \ldots, e_{n-1}$ и деревьев $\mathcal{D}_1, \mathcal{D}_2, \ldots, \mathcal{D}_{n-1}$. В исходной матрице M выделим $(n \times (n-1))$-подматрицу M', содержащую столбцы, соответствующие рёбрам $e_1, e_2, \ldots, e_{n-1}$. В M' поменяем местами (при необходимости) столбцы так, чтобы i-й (слева) столбец отвечал ребру e_i, $i = 1, \ldots, n-1$; поменяем местами и строки в M' так, чтобы i-я (сверху) строка отвечала вершине v_i, $i = 1, \ldots, n$; в итоге получим матрицу M''. Поскольку e_1 — висячее ребро в $\mathcal{D}$, то в первой клетке верхней строки матрицы M'' будет единица, а во всех остальных клетках — нули. После удаления из M'' верхней строки и крайнего левого столбца получим подматрицу M_1'', отвечающую дереву $\mathcal{D}_1$; поскольку e_2 — висячее ребро в $\mathcal{D}_1$, то в первой клетке верхней строки матрицы M_1'' будет единица, а во всех остальных клетках — нули. Рассуждая аналогичным образом и дальше, приходим к выводу, что в i-й сверху и j-й слева клетке матрицы M'' стоит 1, если $i = j$, либо 0, если $i > j$ $(i, j \in \{1, \ldots, n-1\})$. Но это означает, что ранг матрицы M'' равен $n-1$; отсюда следует, что ранг исходной матрицы не меньше $n-1$.

13. Докажем переходы 1) $\Rightarrow$ 2) $\Rightarrow$ 3) $\Rightarrow$ 1). Выйдя из произвольной вершины v_0, обойдём все вершины и все рёбра графа по циклу и вернёмся в v_0. При этом всякий раз, приходя в какую-то очередную вершину v по некоторому ребру e, увеличиваем степень этой вершины на единицу; выходя из v по какому-то ребру e', также увеличиваем степень v на единицу. В итоге по завершении обхода степени всех вершин будут найдены, и это будут чётные числа, т. е. 1) $\Rightarrow$ 2).

2) $\Rightarrow$ 3). Начав с произвольной вершины v графа G, будем перемещаться по рёбрам и вершинам до тех пор, пока не вернёмся (в силу конечности графа) в какую-нибудь ранее пройденную вершину v'; такое перемещение возможно, поскольку с учётом чётности степеней вершин в каждую промежуточную (отличную от v') вершину можно зайти по одному ребру, а выйти по другому. В итоге указанной процедуры в G выделится некоторый цикл Z_1; все рёбра данного цикла удалим из G. Заметим, что если v' совпадает с v, то в оставшемся графе G_1 степени всех вершин окажутся чётными и мы можем повторить вышеуказанную процедуру. Если же v' отличается от v, то степени вершин v и v' будут нечётными, а степени всех остальных вершин — чётными; в этом случае очередную процедуру перемещения начинаем с вершины v' и продолжаем до выделения следующего цикла Z_2, который закончится либо в исходной вершине v, либо в некоторой вершине v'', отличной от v. Указанную процедуру повторяем до тех пор, пока множество всех рёбер исходного графа не окажется разбитым на простые циклы.

3) $\Rightarrow$ 1). Доказательство проведём индукцией по числу простых циклов. При наличии одного простого цикла имеем, очевидно, эйлеров граф. Предположим, что графы, содержащие $1, 2, \ldots, k$ простых циклов, являются эйлеровыми. Пусть G — произвольный граф, множество рёбер которого разбито на $k+1$ простых циклов $Z_1, \ldots, Z_{k+1}$. В силу связности графа G мы можем все циклы упорядочить в виде некоторой последовательности $Z_{i_1}, Z_{i_2}, \ldots, Z_{i_{k+1}}$ так, что Z_{i_2} имеет хотя бы одну общую вершину с Z_{i_1}, Z_{i_3} имеет хотя бы одну общую вершину с Z_{i_1} или с Z_{i_2}, и т. д. В итоге получим, что циклы $Z_{i_1}, \ldots, Z_{i_k}$ образуют некоторую связную компоненту G', а последний цикл $Z_{i_{k+1}}$ имеет некоторую общую с G' вершину v. Используя предположение индукции, построим цикл Z^*, начинающийся и заканчивающийся в v и содержащий $Z_{i_1}, \ldots, Z_{i_k}$; добавляя к Z^* очевидным образом простой цикл $Z_{i_{k+1}}$, получаем цикл, содержащий все вершины и все рёбра исходного графа G.

14. а) 6 графов (рис. 38). б) 10 графов (рис. 39). в) 10 графов (рис. 40).

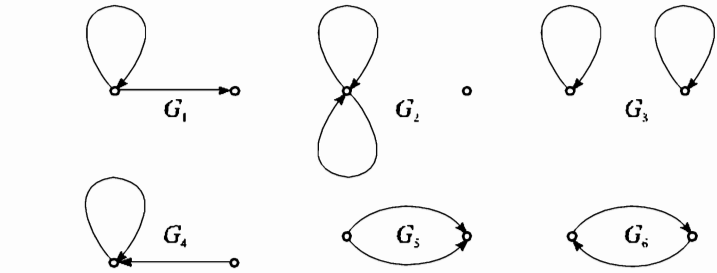


Рис. 38

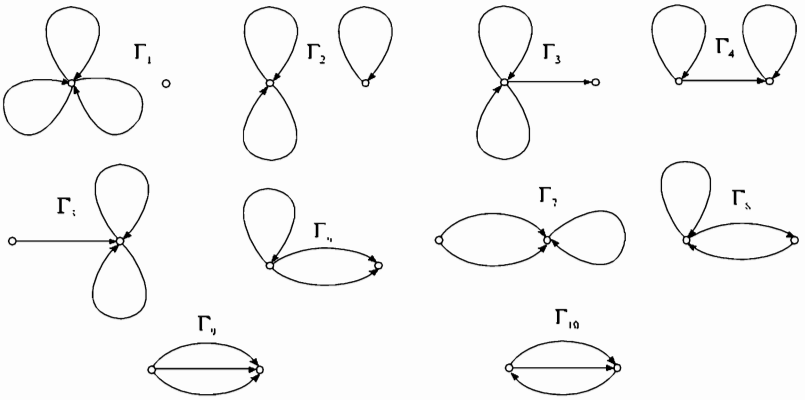


Рис. 39

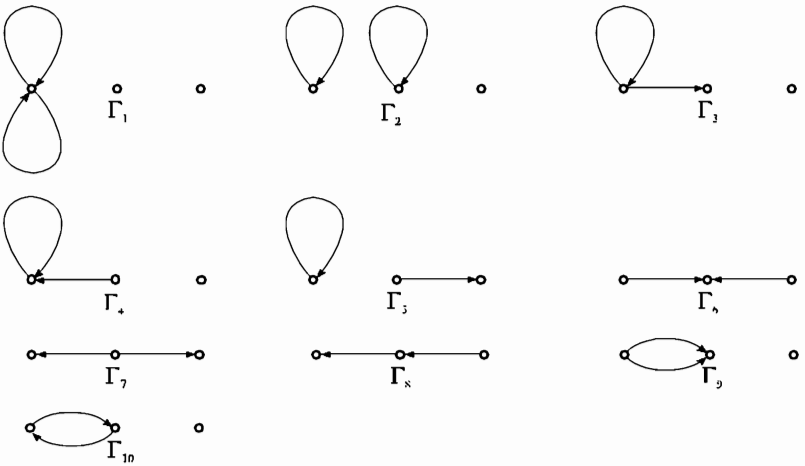


Рис. 40

15. Выйдя из произвольной вершины графа, будем перемещаться по исходящим из вершин дугам, пока не попадём в какую-нибудь ранее пройденную вершину. В силу конечности графа такое событие обязательно наступит и будет свидетельствовать о наличии в графе ориентированного цикла.

К главе III

1. Подставляя единицы вместо двух или более переменных (и, возможно, нули вместо ещё каких-то переменных), получаем константу 1. Подставляя нули вместо любых $n-1$ переменных (и, возможно, ещё какую-нибудь константу вместо последней переменной), получим константу 0. При подстановке единицы вместо какой-то одной переменной и нулей вместо ещё каких-то k переменных, $0 \leqslant k \leqslant n-2$, получаем дизъюнкцию оставшихся $n-k-1$ переменных. При этом можно получить C_n^{n-1} дизъюнкций $n-1$ переменных $x_{i_1} \vee \ldots \vee x_{i_{n-1}}$, $1 \leqslant i_1 < \ldots < i_{n-1} \leqslant n$, C_n^{n-2} дизъюнкций $n-2$ переменных $x_{i_1} \vee \ldots \vee x_{i_{n-2}}$, $1 \leqslant i_1 < \ldots < i_{n-2} \leqslant n$, ..., C_n^1 дизъюнкций (ранга 1) x_i, $1 \leqslant i \leqslant n$, а всего (с учётом равенства $\sum\limits_{i=0}^{n} C_n^i = 2^n$) $2^n - 2$ различных дизъюнкций. Введём обозначение $f_m^2(y_1, \ldots, y_m) = \bigvee\limits_{1 \leqslant i < j \leqslant m} y_i y_j$; при подстановке в $\bigvee\limits_{1 \leqslant i < j \leqslant n} x_i x_j$ нулей вместо некоторых k переменных, $1 \leqslant k \leqslant n-2$, получаем функцию f_{n-k}^2 от оставшихся переменных, а всего подстановками нулей получим $C_n^1 + C_n^2 + \ldots + C_n^{n-2} = 2^n - n - 2$ различных функций. В итоге при подстановке всеми возможными способами констант в $\bigvee\limits_{1 \leqslant i < j \leqslant n} x_i x_j$ получим $2^{n+1} - n - 2$ различных булевых функций (подфункций исходной функции $f_n^2(\widetilde{x})$).

2. а) Очевидно, $f(0, \ldots, 0) = 0$. Рассмотрим теперь произвольный набор $\widetilde{\sigma} = (\sigma_1, \ldots, \sigma_n)$, содержащий k единиц в i_1-м, ..., i_k-м разрядах, $0 < k \leqslant n$. На этом наборе в полиноме Жегалкина для $f(\widetilde{x})$ обращаются в единицу C_k^1 слагаемых $x_{i_1}, \ldots, x_{i_k}$, C_k^2 слагаемых, каждое из которых представляет конъюнкцию двух переменных из $\{x_{i_1}, \ldots, x_{i_k}\}$, ..., C_k^k, т.е. одно слагаемое $x_{i_1} x_{i_2} \ldots x_{i_k}$. Поскольку $\sum\limits_{i=1}^{k} C_k^i = 2^k - 1$, а $2^k - 1 = 1 (\mathrm{mod}\, 2)$, то

$f(\widetilde{\sigma}) = 1$. В итоге получаем, что $f(\widetilde{x})$ обращается в единицу на $2^n - 1$ наборах.

б) Легко заметить, что искомое число не зависит от последовательности $i_1, i_2, \ldots, i_n$ (формально это легко доказать индукцией по n), поэтому без ограничения общности можем рассматривать функцию $f_n(\widetilde{x}) = x_1 \oplus x_1x_2 \oplus \ldots \oplus x_1x_2 \ldots x_n$; через $a(n)$ обозначим число наборов, на которых функция $f_n(\widetilde{x})$ обращается в единицу. Имеем $f_n(\widetilde{x}) = x_1 \oplus x_1(x_2 \oplus x_2x_3 \oplus \ldots \oplus x_2x_3 \ldots x_n) = x_1 \oplus x_1 f_{n-1}(x_2, \ldots, x_n)$. Из последней формулы видно, что $f_n(\widetilde{x})$ обращается в единицу тогда и только тогда, когда x_1 обращается в единицу, а $f_{n-1}(x_2, \ldots, x_n)$ — в нуль; следовательно, $a(n) = 2^{n-1} - a(n-1)$. Последовательно используя полученное рекуррентное соотношение и учитывая, что $a(1) = 1$, получаем $a(n) = 2^{n-1} - 2^{n-2} + 2^{n-3} - \ldots + (-1)^{n-1}$. Воспользовавшись формулой для суммы членов геометрической прогрессии, получаем окончательно $a_n = \dfrac{2^n + (-1)^{n-1}}{3}$.

3. Пусть Φ — формула над A; сложностью формулы Φ будем считать число функциональных символов из A в формуле Φ. Доказательство утверждения проведём индукцией по сложности формул. Для формул сложности 1 утверждение очевидно. Пусть это утверждение справедливо для всех формул сложностей $1, 2, \ldots, k$; докажем его для произвольной формулы Φ сложности $k+1$. По определению Φ представляет собой некоторое выражение $f(\Phi_1, \ldots, \Phi_m)$, где f — символ какой-то функции из A, а каждое из выражений $\Phi_1, \ldots, \Phi_m$ — либо символ переменной, либо формула над A. Пусть x — произвольная переменная из Φ; без ограничения общности можно предположить, что x входит в Φ_1. В силу существенной зависимости функции $f(x_1, \ldots, x_m)$ от всех своих переменных существует булев набор $(\sigma_1, \sigma_2, \ldots, \sigma_m)$, такой, что $f(x, \sigma_2, \ldots, \sigma_m) = x \oplus \sigma_1$. Если среди выражений $\Phi_1, \Phi_2, \ldots, \Phi_m$ имеются формулы, то сложность каждой из них не превосходит k (поскольку внешний символ функции f в формуле Φ не входит ни в одно из выражений $\Phi_1, \ldots, \Phi_m$) и по предположению индукции эта формула реализует функцию, существенно зависящую от всех своих переменных. Следовательно, каждому выражению Φ_i, $i = 1, \ldots, m$, сопоставляется функция φ_i, существенно зависящая от переменных, входящих в φ_i (для случая, когда Φ_i — переменная, это очевидно). Выберем значения переменных из Φ_i, $i = 2, \ldots, m$, так, чтобы значение φ_i оказалось равным σ_i. Значения всех переменных из Φ_1, отличных от x, выберем так, чтобы функция φ_1 обратилась в $x \oplus \delta$, где

δ — какая-нибудь булева константа. В итоге при подстановке выбранных констант вместо всех переменных, отличных от x, реализуемая формулой Ф функция φ окажется равной $x \oplus \delta \oplus \sigma_1$; следовательно, φ существенно зависит от x.

4. Предположим противное. Для булевой функции $f(x_1, \ldots, x_n)$ через f^i_α условимся обозначать подфункцию, получающуюся из f подстановкой константы α вместо x_i; без ограничения общности предположим, что из всех подфункций вида f^i_α наибольшее число существенных переменных имеет подфункция f^1_0. Из исходного предположения следует, что f^1_0 зависит фиктивно от некоторой переменной, например, от x_2, т. е. $f^{1,2}_{0,0} = f^{1,2}_{0,1}$. Поскольку x_1 — существенная переменная, то выполняется хотя бы одно из соотношений $f^{1,2}_{0,0} \neq f^{1,2}_{1,0}$, $f^{1,2}_{0,1} \neq f^{1,2}_{1,1}$. Пусть, например, $f^{1,2}_{0,0} \neq f^{1,2}_{1,0}$. Тогда $f^2_0 = x_1 f^{1,2}_{1,0} \vee \overline{x}_1 f^{1,2}_{0,0}$ существенно зависит от x_1 и от всех существенных переменных подфункции $f^{1,2}_{0,0} = f^1_0$, а это противоречит тому, что f^1_0 имеет наибольшее число существенных переменных.

5. в) Отметим следующее очевидное свойство симметрических функций, вытекающее из их определения: если симметрическая функция $f(x_1, \ldots, x_n)$ обращается в единицу на некотором наборе $\widetilde{\sigma}$, содержащем r единиц, $r \in \{0, 1 \ldots, n\}$, то эта функция обращается в единицу и на любом другом наборе, содержащем r единиц; число r в этом случае назовём рабочим числом функции f. Из этого свойства следует, что произвольную симметрическую функцию $f(x_1, \ldots, x_n)$ можно задать набором всех её рабочих чисел $r_1, r_2, \ldots, r_m$, где $0 \leqslant r_1 < r_2 < \ldots < r_m \leqslant n$, $m \leqslant n+1$; такую функцию условимся обозначать через $S^{r_1, \ldots, r_m}_n(\widetilde{x})$ (нетрудно заметить, например, что $x_1 \& x_2 = S^2_2(\widetilde{x})$, $x_1 \vee x_2 \vee x_3 = S^{1,2,3}_3(\widetilde{x})$, $x_1 x_2 \vee x_1 x_3 \vee x_2 x_3 = S^{2,3}_3(\widetilde{x})$, $x_1 / x_2 = S^{0,1}_2(\widetilde{x})$).

Предположим теперь, что задана произвольная булева функция $f(\widetilde{x})$, $\widetilde{x} = (x_1, \ldots, x_n)$, и $\widetilde{\sigma}_1, \ldots, \widetilde{\sigma}_k$ — все булевы наборы длины n, на которых $f(\widetilde{x})$ обращается в единицу; через $|\widetilde{\sigma}_1|, \ldots, |\widetilde{\sigma}_k|$ обозначим числа, двоичными записями которых являются соответственно наборы $\widetilde{\sigma}_1, \ldots, \widetilde{\sigma}_k$ (т. е. $|\widetilde{\sigma}_i| = \sum_{j=1}^{n} \sigma_{i,j} 2^{n-j}$, $\widetilde{\sigma}_i = (\sigma_{i,1}, \ldots, \sigma_{i,n})$, $i = 1, \ldots, k$). Функции $f(\widetilde{x})$ поставим в соответствие симметрическую функцию $S^{|\widetilde{\sigma}_1|, \ldots, |\widetilde{\sigma}_k|}_{2^n-1}(\widetilde{y})$; теперь нетрудно заметить, что если отождествить первые 2^{n-1} переменных

$y_1, \ldots, y_{2^{n-1}}$ и обозначить их через x_1, затем отождествить следующие 2^{n-2} переменных $y_{2^{n-1}+1}, \ldots, y_{2^{n-1}+2^{n-2}}$ и обозначить их через x_2, ..., наконец, отождествить последние две переменные $y_{2^{n-1}+\ldots+2^2+1}, y_{2^{n-1}+\ldots+2^2+2}$ и обозначить их через x_{n-1}, а последнюю переменную y_{2^n-1} обозначить через x_n, то получим $f(\widetilde{x})$ (очевидно также, что указанное соответствие между множеством булевых функций от n переменных и множеством симметрических функций от $2^n - 1$ переменных является взаимно однозначным).

6. Полином Жегалкина $P(f)$ для $f(x_1, x_2, x_3)$ получается из $c_0 \oplus c_1x_1 \oplus c_2x_2 \oplus c_3x_3 \oplus c_{1,2}x_1x_2 \oplus c_{1,3}x_1x_3 \oplus c_{2,3}x_2x_3 \oplus \oplus c_{1,2,3}x_1x_2x_3$ выбором подходящих значений для коэффициентов $c_0, \ldots, c_{1,2,3}$ (при равном нулю коэффициенте соответствующее слагаемое вычеркивается). Поскольку $f(x_1, x_2, x_3)$ существенно зависит от каждой из переменных x_1, x_2, x_3, то каждая из этих переменных присутствует в $P(f)$. В зависимости от значений c_1, c_2, c_3 рассмотрим следующие три случая: а) $c_1 = c_2 = c_3 = 0$; б) один из коэффициентов, например, c_1 равен единице; в) два коэффициента, например, c_1 и c_2, или все три коэффициента равны единице.

а) Если в $P(f)$ присутствует хотя бы одна конъюнкция ранга 2, например, x_1x_2, то при $x_3 \equiv 0$ получим полином вида $c_0 \oplus x_1x_2$, который задаёт либо x_1x_2 (при $c_0 = 0$), либо $\overline{x_1x_2}$ (при $c_1 = 1$), и утверждение (из задачи) выполняется. При отсутствии в $P(f)$ конъюнкций ранга 2 в $P(f)$ обязательно должна присутствовать конъюнкция $x_1x_2x_3$ и при $x_3 \equiv 1$ опять же получим $c_0 \oplus x_1x_2$.

б) Если в $P(f)$ присутствует слагаемое x_2x_3, то при $x_1 \equiv 0$ получим (аналогично предыдущему) $c_0 \oplus x_2x_3$. Предположим, что x_2x_3 отсутствует в $P(f)$. Если $x_1x_2x_3$ тоже отсутствует, то в таком случае в силу существенной зависимости f от x_1, x_2, x_3 в $P(f)$ должны быть слагаемые x_1x_2 и x_1x_3 и при $x_1 \equiv 1$ утверждение выполняется. Пусть $x_1x_2x_3$ присутствует в $P(f)$. Тогда: при отсутствии x_1x_2 и x_1x_3 положим $x_1 \equiv 1$ и получим $c_0 \oplus 1 \oplus x_2x_3$; при наличии одного из слагаемых x_1x_2, x_1x_3, например, x_1x_2 положим $x_2 \equiv 1$ и получим $c_0 \oplus x_1x_3$; при наличии обоих слагаемых x_1x_2 и x_1x_3 положим $x_1 \equiv 1$ и получим $c_0 \oplus 1 \oplus x_2 \oplus x_3 \oplus x_2x_3 = c_0 \oplus 1 \oplus (x_2 \vee x_3)$; во всех этих случаях утверждение выполняется.

в) В этом случае полагаем $x_1 \equiv 0$ и из полинома $P(f)$ получаем либо $c_0 \oplus x_2 \oplus x_3$, либо $c_0 \oplus x_2 \oplus x_3 \oplus x_2x_3$, т. е. опять

же симметрическую функцию, существенно зависящую от двух переменных.

7. Докажем, что не существует формулы над $\{x \to y\}$, реализующей $x\&y$. Поскольку $x\&y$ имеет только две существенные переменные x и y, то, очевидно, мы можем рассматривать формулы, содержащие только переменные x и y (несущественные переменные в формулах можно заменить, например, на x). Поскольку $x\&y$ обращается в единицу только на одном из четырёх наборов значений переменных, то наше искомое утверждение, очевидно, вытекает из следующего (вспомогательного) утверждения: любая формула над $\{x \to y\}$, содержащая только переменные x и y, обращается в единицу не менее чем на двух наборах значений этих переменных. Последнее утверждение докажем индукцией по сложности формулы. Для формулы $x \to y$ сложности 1 оно, очевидно, справедливо. Пусть оно справедливо для формул сложности $1, 2, \ldots, k$. Пусть $\Phi = A_1 \to A_2$ — формула сложности $k+1$. В этой формуле выражение A_2 — либо переменная, либо формула сложности не более чем k. С учётом предположения индукции функция φ_2, отвечающая A_2, обращается в единицу не менее чем на двух наборах значений переменных x, y; но в таком случае из определения импликации следует, что и функция, реализуемая формулой Φ, обращается в единицу не менее чем на двух наборах.

8. а) Представим булеву функцию $f(x_1, \ldots, x_n)$, отличную от константы 0, в виде совершенной дизъюнктивной нормальной формы Φ. Все слагаемые из Φ разобьём на не более чем 2^{n-1} подсумм $M_{(\sigma_1,\ldots,\sigma_{n-1})}$, $(\sigma_1, \ldots, \sigma_{n-1}) \in E^{n-1}$, помещая в $M_{(\sigma_1,\ldots,\sigma_{n-1})}$ элементарные конъюнкции, имеющие в качестве множителя $x_1^{\sigma_1} \cdot \ldots \cdot x_{n-1}^{\sigma_{n-1}}$ (какие-то подсуммы могут оказаться пустыми). Далее выделим подсуммы, содержащие по два слагаемых; в каждой из выделенных подсумм слагаемые $x_1^{\sigma_1} \cdot \ldots \cdot x_{n-1}^{\sigma_{n-1}} x_n$ и $x_1^{\sigma_1} \cdot \ldots \cdot x_{n-1}^{\sigma_{n-1}} \overline{x_n}$ «склеиваем» в одну конъюнкцию $x_1^{\sigma_1} \cdot \ldots \cdot x_{n-1}^{\sigma_{n-1}}$ ранга $n-1$. В итоге в полученной формуле Φ', эквивалентной Φ, останется не более чем 2^{n-1} слагаемых.

б) Рассмотрим линейную функцию $l_n(\widetilde{x}) = x_1 \oplus \ldots \oplus x_n$. Эта функция обращается в единицу на 2^{n-1} наборах и потому представляющая ее совершенная д. н. ф. содержит 2^{n-1} элементарных конъюнкций ранга n. С другой стороны, произвольная д. н. ф. Φ, представляющая $l_n(\widetilde{x})$, не может содержать конъюнкции ранга, меньшего чем n. Действительно, предположим, что Φ содержит некоторую конъюнкцию K меньшего чем n, ранга, например, $x_1^{\sigma_1} \cdot \ldots \cdot x_k^{\sigma_k}$, $k < n$. Эта конъюнкция, а вместе

с ней и Ф, обращаются в единицу на двух соседних наборах $\widetilde{\sigma} = (\sigma_1, \ldots, \sigma_k, 0, \ldots, 0, 0)$ и $\widetilde{\sigma}' = (\sigma_1, \ldots, \sigma_k, 0, \ldots, 0, 1)$. Но один из наборов $\widetilde{\sigma}$, $\widetilde{\sigma}'$ содержит чётное число единиц, а линейная функция $l_n(\widetilde{x})$ на таком наборе обращается в 0. Получаем противоречие. Следовательно, реализующая функцию $l_n(\widetilde{x})$ д. н. ф. является совершенной и содержит 2^{n-1} элементарных конъюнкций.

9. В случае, когда $f(x_1, \ldots, x_n)$, $n \geqslant 1$, — булева константа, утверждение, очевидно, выполняется. Пусть, далее, $P(f)$ — полином Жегалкина для булевой функции $f(x_1, \ldots, x_n)$, не являющейся константой. Предположим, что $P(f) = K_1 \oplus K_2 \oplus \ldots \oplus K_l$, где K_i — конъюнкция не более чем $n-1$ переменных, $i = 1, \ldots, l$. Рассмотрим последовательность полиномов $P_1 = K_1$, $P_2 = K_1 \oplus K_2$, ..., $P_l = K_1 \oplus \ldots \oplus K_l$. Очевидно, K_1 обращается в единицу на чётном числе наборов значений переменных $x_1, \ldots, x_n$. Предположим, что и каждый из полиномов $P_1, P_2, \ldots, P_i$ обращается в единицу на чётном числе булевых наборов. Поскольку ранг конъюнкции K_{i+1} не превосходит $n-1$, то эта конъюнкция обращается в единицу на чётном числе каких-то наборов $\widetilde{\sigma}_1, \ldots, \widetilde{\sigma}_{2m}$, $m \geqslant 1$. На каких-то r наборах, например, $\widetilde{\sigma}_1, \ldots, \widetilde{\sigma}_r$, полином P_i обращается в единицу, а на остальных наборах $\widetilde{\sigma}_{r+1}, \ldots, \widetilde{\sigma}_{2m}$ — в нуль. После прибавления к P_i конъюнкции K_{i+1} получим полином P_{i+1}, который на наборах $\widetilde{\sigma}_1, \ldots, \widetilde{\sigma}_r$ будет обращаться в нуль, на наборах $\widetilde{\sigma}_{r+1}, \ldots, \widetilde{\sigma}_{2m}$ — в единицу, а на всех остальных наборах значения P_{i+1} совпадут со значениями P_i; в таком случае если полином P_i обращается в единицу на чётном числе наборов, то и полином P_{i+1} будет обращаться в единицу на чётном числе наборов. Таким образом, используя индукцию, мы доказали, что если конъюнкция $x_1 \cdot \ldots \cdot x_n$ не входит в $P(f)$, то f обращается в единицу на чётном числе наборов.

Предположим теперь, что $P(f) = K_1 \oplus \ldots \oplus K_{l-1} \oplus K_l$, $K_l = x_1 \cdot \ldots \cdot x_n$. Как было показано выше, полином $P_{l-1} = K_1 \oplus \ldots \oplus K_{l-1}$ обращается в единицу на чётном числе каких-то наборов $\widetilde{\sigma}_1, \ldots, \widetilde{\sigma}_{2m}$. Конъюнкция $x_1 \cdot \ldots \cdot x_n$ обращается в единицу на единственном наборе $\widetilde{1}$. Если набор $\widetilde{1}$ присутствует среди $\widetilde{\sigma}_1, \ldots, \widetilde{\sigma}_{2m}$, то полином $P(f)$ будет обращаться в единицу на $(2m-1)$ наборах из $\{\widetilde{\sigma}_1, \ldots, \widetilde{\sigma}_{2m}\} \backslash \{\widetilde{1}\}$; если же среди $\widetilde{\sigma}_1, \ldots, \widetilde{\sigma}_{2m}$ нет набора $\widetilde{1}$, то $P(f)$ будет обращаться в единицу на $(2m+1)$ наборах из $\{\widetilde{\sigma}_1, \ldots, \widetilde{\sigma}_{2m}\} \cup \{\widetilde{1}\}$.

10. г) Доказываемое равенство следует из включений $[B] \subseteq [[B]]$, $[B] \supseteq [[B]]$. Первое из них следует (как и в задаче а)) из

определения замыкания. Индукцией по сложности формул над $[B]$ докажем второе включение, т. е. докажем, что для каждой формулы над $[B]$ (реализующей функцию из $[[B]]$) существует эквивалентная ей формула над B (реализующая функцию из $[B]$). Формула Φ сложности 1 над $[B]$ представляет собой выражение $f(x_{i_1}, \ldots, x_{i_m})$, где f — функция из $[B]$, которая реализуется некоторой формулой Ψ над B, причём формулы Φ и Ψ эквивалентны. Пусть наше утверждение справедливо для всех формул над $[B]$ сложности $1, \ldots, k$; докажем его для произвольной формулы Φ над $[B]$ сложности $k+1$. Формулы Φ представляет собой выражение $f(A_1, \ldots, A_m)$, где f — некоторая функция из $[B]$, а каждое из выражений $A_1, \ldots, A_m$ является либо переменной, либо формулой над $[B]$. Пусть, например, $A_1, \ldots, A_n$ — это формулы $\Phi_1, \ldots, \Phi_n$ над $[B]$, а остальные выражения $A_{n+1}, \ldots, A_m$ — это просто переменные $x_{n+1}, \ldots, x_m$. Сложность каждой из формул $\Phi_1, \ldots, \Phi_n$ не превосходит k, и по предположению индукции для этих формул существуют эквивалентные им формулы $\Psi_1, \ldots, \Psi_n$ над B.

Реализуем функцию $f(x_1, \ldots, x_m)$ формулой над B и получим равенство

$$f(x_1, \ldots, x_m) = \Psi[x_1, \ldots, x_m], \tag{1}$$

в левой части которого — формула сложности 1 над $[B]$, а в правой части — какая-то формула над B, содержащая символы переменных $x_1, \ldots, x_m$. Вместо переменных $x_1, \ldots, x_n$ в левую часть равенства (1) подставим соответственно символы $A_1, \ldots, A_n$, а в правую часть — соответственно символы $\Psi_1, \ldots, \Psi_n$, затем каждый из символов $A_1, \ldots, A_n$ заменим на обозначаемую этим символом формулу над $[B]$, а каждый из символов $\Psi_1, \ldots, \Psi_n$ — на соответствующую формулу над B. В итоге получим равенство

$$f(A_1, \ldots, A_n, x_{n+1}, \ldots, x_m) = \Psi^*,$$

в правой части которого — некоторая формула Ψ^* над B, эквивалентная исходной формуле Φ.

11. Ни один из классов T_0, T_1, S, L, M не содержит, например, функцию x/y; это означает, что все эти классы отличны от P_2. Замкнутость каждого из рассматриваемых классов можно доказать индукцией по сложности формул над этим классом. Проделаем это для S и M.

Пусть $f(x_{i_1}, \ldots, x_{i_m})$ — формула сложности 1 над S; очевидно, она реализует функцию из S. Пусть любая формула над S сложности не более k реализует самодвойственную функцию. Произвольную формулу Φ над S сложности $k+1$, содержащую

переменные $x_1, \ldots, x_m$, без ограничения общности можно представить в виде

$$\Phi[x_1, \ldots, x_m] = f(A_1, \ldots, A_n, x_{n+1}, \ldots, x_m), \qquad (2)$$

где $A_1, \ldots, A_n$ — подформулы сложности, не превосходящей k, реализующие в соответствии с предположением индукции некоторые самодвойственные функции $\varphi_1, \ldots, \varphi_n$. Пусть $\widetilde{x} = (x_1, \ldots, x_m)$, $\overline{\widetilde{x}} = (\overline{x}_1, \ldots, \overline{x}_m)$; через $\widetilde{x}_1, \ldots, \widetilde{x}_n$ и $\overline{\widetilde{x}}_1, \ldots, \overline{\widetilde{x}}_n$ обозначим наборы переменных и наборы отрицаний переменных соответственно для функций $\varphi_1, \ldots, \varphi_n$. Заменяя в (2) переменные их отрицаниями и «навешивая» отрицания над Φ и f, получаем

$$\overline{\Phi}[\overline{\widetilde{x}}] = \overline{f}(\varphi_1(\overline{\widetilde{x}}_1), \ldots, \varphi_n(\overline{\widetilde{x}}_n), \overline{x}_{n+1}, \ldots, \overline{x}_m).$$

Из последнего равенства «навешиванием» двойных отрицаний над $\varphi_1, \ldots, \varphi_n$ получаем

$$\overline{\Phi}[\overline{\widetilde{x}}] = \overline{f}(\overline{\varphi}_1(\overline{\widetilde{x}}_1), \ldots, \overline{\varphi}_n(\overline{\widetilde{x}}_n), \overline{x}_{n+1}, \ldots, \overline{x}_m);$$

отсюда с учётом самодвойственности функций $\varphi_1, \ldots, \varphi_n$ следует

$$\overline{\Phi}[\overline{\widetilde{x}}] = \overline{f}(\overline{\varphi}_1(\widetilde{x}_1), \ldots, \overline{\varphi}_n(\widetilde{x}_n), \overline{x}_{n+1}, \ldots, \overline{x}_m).$$

Учитывая самодвойственность функции f, из последнего соотношения выводим

$$\overline{\Phi}[\overline{\widetilde{x}}] = f(\varphi_1(\widetilde{x}_1), \ldots, \varphi_n(\widetilde{x}_n), x_{n+1}, \ldots, x_m). \qquad (3)$$

Сравнивая (2) и (3), получаем

$$\Phi[\widetilde{x}] = \overline{\Phi}[\overline{\widetilde{x}}],$$

т. е. реализуемая формулой Φ функция является самодвойственной.

В доказательстве замкнутости класса M индукцией по сложности формул над M основание индукции, как и выше, очевидно. При индуктивном переходе формулу Φ над M сложности $k+1$ опять же представляем в виде (2), имея в виду в данном случае, что $A_1, \ldots, A_n$ — формулы над M, реализующие по индуктивному предположению монотонные булевы функции $\varphi_1, \ldots, \varphi_n$. Пусть $\widetilde{\alpha}$ и $\widetilde{\beta}$ — произвольная пара наборов значений переменных $x_1, \ldots, x_m$, удовлетворяющая условию $\widetilde{\alpha} \leqslant \widetilde{\beta}$, а $\varphi(\widetilde{x})$ — функция, реализуемая формулой $\Phi[\widetilde{x}]$. Пара наборов $\widetilde{\alpha}$ и $\widetilde{\beta}$ определяет соответствующие пары наборов $\widetilde{\alpha}_1$ и $\widetilde{\beta}_1$, ..., $\widetilde{\alpha}_n$ и $\widetilde{\beta}_n$ значений переменных функций $\varphi_1, \ldots, \varphi_n$. Из условия $\widetilde{\alpha} \leqslant \widetilde{\beta}$ следуют соотношения $\widetilde{\alpha}_1 \leqslant \widetilde{\beta}_1$, ..., $\widetilde{\alpha}_n \leqslant \widetilde{\beta}_n$, а из монотонности функций

$\varphi_1, \ldots, \varphi_n$ получаем $\varphi_1(\widetilde{\alpha}_1) \leqslant \varphi_1(\widetilde{\beta}_1)$, ..., $\varphi_n(\widetilde{\alpha}_n) \leqslant \varphi_n(\widetilde{\beta}_n)$; учитывая ещё и монотонность функции f, а также соотношение (2), получаем

$$f(\varphi_1(\widetilde{\alpha_1}), \ldots, \varphi_n(\widetilde{\alpha_n}), \alpha_{n+1}, \ldots, \alpha_m) \leqslant \\ \leqslant f(\varphi_1(\widetilde{\beta}_1), \ldots, \varphi_n(\widetilde{\beta_n}), \beta_{n+1}, \ldots, \beta_m),$$

т. е. $\varphi(\widetilde{\alpha}) \leqslant \varphi(\widetilde{\beta})$.

12. $\left|T_0^n\right| = \left|T_1^n\right| = 2^{2^n-1}$, $|S^n| = 2^{2^{n-1}}$, $|L^n| = 2^{n+1}$. *Указание.* При подсчёте $\left|T_0^n\right|$, $\left|T_1^n\right|$, $|S^n|$ воспользоваться табличным представлением булевых функций, а при подсчёте $|L^n|$ — представлением в виде полиномов Жегалкина.

13. а) $2^{2^n} - 2^{2^n-2} + 2^{n-1}$. б) 2^{n-1}. в) $2^{2^n-1} + 1$.

г) Каждая функция $f(x_1, \ldots, x_n)$ из T_0^n при табличном задании в верхнем разряде столбца значений содержит 0; всего таких функций $2^{2^n} - 1$. Для самодвойственной функции выполняется соотношение

$$f(x_1, \ldots, x_n) = \overline{f}(\overline{x}_1, \ldots, \overline{x}_n), \tag{4}$$

из которого следует, что если в таблице для f значения на наборах $(0, \sigma_2, \ldots, \sigma_n)$ заданы (произвольным образом), то значения f на наборах $(1, \sigma_2, \ldots, \sigma_n)$ определяются уже из соотношения (4); отсюда получаем $|S^n| = 2^{2^{n-1}}$. Множество $T_0^n \cup S^n$ получается добавлением к T_0^n тех функций из S^n, которые не сохраняют 0, т. е. содержат 1 в верхнем разряде столбца значений f (в таблице для f), а таких функций, как теперь уже нетрудно заметить, всего $2^{2^{n}-1}$ штук; следовательно, $\left|T_0^n \cup S^n\right| = 2^{2^n-1} + 2^{2^{n}-1}$.

Множество L^n разобьём на два подмножества: $L^n = A^n \cup B^n$, где $A^n = \left(T_0^n \cup S^n\right) \cap L^n$, а $B^n = L^n \setminus \left(T_0^n \cup S^n\right)$ (в B^n входят все те линейные функции $f(x_1, \ldots, x_n)$, каждая из которых не сохраняет 0 и не является самодвойственной). Функции из B^n представимы, как легко заметить, полиномами Жегалкина вида $1 \oplus x_{i_1} \oplus \ldots \oplus x_{i_k}$, где $x_{i_1}, \ldots, x_{i_k} \in \{x_1, \ldots, x_n\}$, а k — нечётное число; таких полиномов $C_n^1 + C_n^3 + \ldots + C_n^{n-1} = 2^{n-1}$ штук при чётном n или опять же $C_n^1 + C_n^3 + \ldots + C_n^n = 2^{n-1}$ штук при нечётном n. Учитывая, что $|L^n| = 2^{n+1}$, получаем далее $|A^n| = |L^n| - |B^n| = 3 \cdot 2^{n-1}$. Таким образом, $\left(T_0^n \cup S^n\right) \setminus L^n = \left(T_0^n \cup S^n\right) \setminus \left(\left(T_0^n \cup S^n\right) \cap L^n\right)$, а $\left|\left(T_0^n \cup S^n\right) \setminus L^n\right| = \left|T_0^n \cup S^n\right| - |A^n| = 2^{2^n-1} + 2^{2^{n-1}-1} - 3 \cdot 2^{n-1}$.

д) В $A^n = \left(T_0^n \cup T_1^n \cup S^n\right) \cap L^n$ не входят, очевидно, все те функции из L^n, каждая из которых не сохраняет нуль,

не сохраняет единицу и не является самодвойственной. Пусть $P = c_0 \oplus c_1 x_1 \oplus \ldots \oplus c_n x_n$ — полином Жегалкина, представляющий булеву функцию f из $L^n \backslash A^n$. Поскольку f не сохраняет нуль, то $c_0 = 1$; поскольку f не сохраняет единицу, то в полиноме может присутствовать только нечётное число переменных; поскольку f не является самодвойственной функцией, то в P может присутствовать только чётное число переменных. Перечисленным условиям удовлетворяет единственный полином $P = 1$. Значит, $|A^n| = |L^n| - 1 = 2^{n+1} - 1$.

е) Множество $S^n \backslash (T_0^n \backslash L^n)$ можно получить, выполнив последовательно следующие две операции: 1) из S^n удалить все функции, сохраняющие нуль; 2) к множеству S_0^n, полученному после выполнения предыдущей операции, добавить обратно те самодвойственные функции из S^n, каждая из которых сохраняет нуль и одновременно является линейной. В S^n ровно $2^{2^{n-1}-1}$ функций сохраняют нуль. Полином для самодвойственной линейной функции $f(x_1, \ldots, x_n)$, сохраняющей нуль, представляет собой сумму $x_{i_1} \oplus \ldots \oplus x_{i_k}$ нечётного числа переменных; таких полиномов 2^{n-1}. В итоге имеем $\left|S^n \backslash (T_0^n \backslash L^n)\right| = 2^{2^{n-1}} - 2^{2^{n-1}-1} + 2^{n-1} = 2^{2^{n-1}-1} + 2^{n-1}$.

ж) Множество T_0^n содержит 2^{2^n-1} функций. Добавим к ним 2^{2^n-2} функций из T_1^n, сохраняющих единицу, но не сохраняющих нуль. К полученному множеству $T_0^n \cup T_1^n$ добавим $2^{2^{n-1}-1}$ самодвойственных функций из S^n, не сохраняющих константы, и получим $T_0^n \cup T_1^n \cup S^n$. Убедимся, что $L^n \subseteq T_0^n \cup T_1^n \cup S^n$. Действительно, возьмём произвольную функцию f из L^n и полином Жегалкина $c_0 \oplus x_{i_1} \oplus \ldots \oplus x_{i_k}$ для неё. Если $f \notin T_0^n$, то $c_0 = 1$; если $f \notin T_1^n$, а $c_0 = 1$, то k — нечётное число; если $f \notin S^n$, то k — чётное число. Перечисленные условия одновременно выполняться не могут, и потому $L^n \subseteq T_0^n \cup T_1^n \cup S^n$. В итоге получаем $\left|T_0^n \cup T_1^n \cup S^n \cup L^n\right| = 2^{2^n-1} + 2^{2^n-2} + 2^{2^{n-1}-1} = 3 \times \times 2^{2^n-2} + 2^{2^{n-1}-1}$.

з) Среди 2^{n+1} линейных функций из L^n ровно 2^n самодвойственных (каждая из них представима полиномом Жегалкина $c_0 \oplus x_{i_1} \oplus \ldots \oplus x_{i_k}$ с нечётным числом переменных и произвольным $c_0 \in \{0, 1\}$). Каждая из оставшихся в $L^n \backslash S^n$ функций содержит в полиноме Жегалкина чётное число переменных. При $k = 0$ две функции — константы 0 и 1 — являются монотонными. Пусть $f \in (L^n \backslash S^n) \backslash \{0, 1\}$; реализующий f полином Жегалкина содержит по крайней мере две переменные, например, x_1 и x_2 в качестве слагаемых. При этом

$(c_0 \oplus 1, 0, 0, \ldots, 0) < (c_0 \oplus 1, 1, 0, \ldots, 0)$, а $f(c_0 \oplus 1, 0, 0, \ldots, 0) = 1 > 0 = f(c_0 \oplus 1, 1, 0, \ldots, 0)$ и функция f не может быть монотонной. В итоге имеем $(L^n \backslash S^n) \backslash M^n = 2^n - 2$.

14. Если $f(x_1, \ldots, x_n) \notin S$, то найдутся два противоположных набора $\widetilde{\sigma} = (\sigma_1, \ldots, \sigma_n)$ и $\overline{\widetilde{\sigma}} = (\overline{\sigma}_1, \ldots, \overline{\sigma}_n)$, на которых f принимает одно и то же значение α. Если в $f(x_1, \ldots, x_n)$ вместо x_i подставим $x_i^{\sigma_i}$, $i = 1, \ldots, n$, то получим функцию $\varphi(x) = f(x^{\sigma_1}, \ldots, x^{\sigma_n})$, для которой выполняются соотношения $\varphi(0) = f(0^{\sigma_1}, \ldots, 0^{\sigma_n}) = f(\overline{\sigma_1}, \ldots, \overline{\sigma_n}) = \alpha = f(\sigma_1, \ldots, \sigma_n) = f(1^{\sigma_1}, \ldots, 1^{\sigma_n}) = \varphi(1)$, а это означает, что $\varphi(x) \equiv \alpha$.

15. Если $f(x_1, \ldots, x_n) \notin M$, то существуют два набора $\widetilde{\sigma}$ и $\widetilde{\sigma}'$, такие, что $f(\widetilde{\sigma}) > f(\widetilde{\sigma}')$, а $\widetilde{\sigma} < \widetilde{\sigma}'$. Рассмотрим последовательность наборов $\widetilde{\sigma}_1, \ldots, \widetilde{\sigma}_k$, в которой каждый из последующих наборов получается из предыдущего заменой нуля в каком-то одном разряде, причём $\widetilde{\sigma}_1 = \widetilde{\sigma}$, а $\widetilde{\sigma}_k = \widetilde{\sigma}'$. Очевидно, в указанной последовательности найдутся два соседних набора $\widetilde{\sigma}_i$ и $\widetilde{\sigma}_j$, различающиеся ровно в одном, например, в первом разряде и удовлетворяющие условиям: $\widetilde{\sigma}_i = (0, \sigma_2, \ldots, \sigma_n) < \widetilde{\sigma}_j = (1, \sigma_2, \ldots, \sigma_n)$, $f(\widetilde{\sigma}_i) > f(\widetilde{\sigma}_j)$. Подставляя в $f(x_1, x_2, \ldots, x_n)$ вместо x_1 переменную x, а вместо остальных переменных $x_2, \ldots, x_n$ соответственно константы $\sigma_2, \ldots, \sigma_n$, получим функцию $\varphi(x) = f(x, \sigma_2, \ldots, \sigma_n)$, удовлетворяющую соотношениям $\varphi(0) = f(0, \sigma_2, \ldots, \sigma_n) > f(1, \sigma_2, \ldots, \sigma_n) = \varphi(1)$; этим условиям удовлетворяет единственная элементарная функция φ от одной переменной — инверсия $\overline{x}$.

16. Пусть $f(x_1, \ldots, x_n) \notin L$, а P — полином Жегалкина, представляющий f. Пусть A — множество всех тех конъюнкций из P, каждая из которых содержит не менее двух переменных; поскольку $f \notin L$, то множество A непусто. Из всех конъюнкций множества A выделим конъюнкцию $K = x_{i_1} \cdot \ldots \cdot x_{i_k}$, $2 \leqslant k \leqslant n$, содержащую наименьшее число переменных. Подставим в P единицу вместо каждой из переменных $x_{i_3}, \ldots, x_{i_k}$ и нули вместо всех остальных переменных, кроме x_{i_1}, x_{i_2}. Все конъюнкции из A, кроме K, окажутся равными нулю, и полином P превратится (после очевидных упрощений) в формулу $x_{i_1}x_{i_2} \oplus \alpha x_{i_1} \oplus \beta x_{i_2} \oplus \gamma$, где $\alpha, \beta, \gamma \in \{0, 1\}$; подставляя в эту формулу $x \oplus \beta$ вместо x_{i_1} и $y \oplus \alpha$ вместо x_{i_2}, получим $xy \oplus \varepsilon$, где ε — некоторая булева константа (равная $\alpha\beta \oplus \gamma$).

17. *Необходимость* следует из того, что классы T_0, T_1, S, L, M замкнуты и отличны от P_2 (см. задачу 11). Допустим, например, что заданное множество B содержится в T_0, т. е. $B \subseteq T_0$. Тогда с учетом свойств замыкания (см. задачу

10.д)) и замкнутости T_0 получаем $[B] \subseteq [T_0] = T_0$. Но $T_0 \neq P_2$, следовательно, B также не обладает полнотой в P_2.

Достаточность. Пусть B не содержится целиком ни в одном из классов T_0, T_1, S, L, M. В таком случае, очевидно, в B можно выделить некоторую подсистему $B' = (f_1, \dots, f_5)$, такую, что $f_1 \notin T_0$, $f_2 \notin T_1$, $f_3 \notin S$, $f_4 \notin L$, $f_5 \notin M$ (некоторые из функций $f_1, \dots, f_5$ могут совпадать). Покажем, что формулами над B' можно реализовать функции $\overline{x}$, $x\&y$.

Вначале формулами над B' реализуем константы. Возьмём функцию f_1. Если $f_1(1, \dots, 1) = 1$, то $\varphi(x) = f_1(x, \dots, x)$ есть константа 1, ибо $\varphi(0) = f_1(0, \dots, 0) = 1$, $\varphi(1) = f_1(1, \dots, 1) = 1$; при наличии константы 1 вторая константа 0 реализуется формулой $f_2(1, \dots, 1)$, поскольку из условия $f_2 \notin T_2$ вытекает $f_2(1, \dots, 1) = 0$ (при окончательном построении формулы над B' единицы в выражении $f_2(1, \dots, 1)$ заменяются на $f_1(x, \dots, x)$; эти замены и аналогичные им ниже мы не выполняем, чтобы не загромождать выкладки). Пусть теперь $f_1(1, \dots, 1) = 0$; тогда $\varphi(x) = f_1(x, \dots, x)$ есть $\overline{x}$, ибо $\varphi(0) = f_1(0, \dots, 0) = 1$, $\varphi(1) = f_1(1, \dots, 1) = 0$. При наличии $\overline{x}$ из несамодвойственной функции f_3 получим константу (см. задачу 14), после чего, опять же используя отрицание, получим и вторую константу. Итак, формулами над B можно реализовать обе булевы константы.

При наличии констант и немонотонной функции f_5 строим формулу над B', реализующую $\overline{x}$ (см. задачу 15).

При наличии констант 0, 1 и функций $\overline{x}$, f_4 (и соответствующих формул над B') строим формулу над B', реализующую $x\&y$ (см. задачу 16; заметим, что при наличии инверсии функция $x\&y$ получается из $\overline{x\&y}$ очевидным образом).

Таким образом, установлено, что $\overline{x}$ и $x\&y$ можно реализовать формулами над B' (а тем самым и над B). Далее остаётся воспользоваться известным фактом полноты системы $\{\overline{x}, x\&y\}$ и теоремой о полноте двух систем.

18. Функция f сохраняет константы; иначе если, скажем, $f(\widetilde{0}) = 1$, то в силу монотонности f обращалась бы в 1 и на всех остальных наборах значений переменных, т. е. была бы константой 1, а это невозможно, т. к. f существенно зависит не менее, чем от двух переменных. Отсюда получаем, что $\overline{f} \notin T_0$ и $\overline{f} \notin T_1$. По тем же причинам f не может быть и линейной функцией (см. задачу 13.з)); в таком случае, очевидно, и $\overline{f} \notin L$. Поскольку $\overline{f}$ не сохраняет константы, то $\overline{f} \notin M$. Константа 0 — несамодвойственная функция. В соответствии с критерием полноты получаем, что система $\{0, \overline{f}\}$ полна в P_2.

19. а) Пусть полином Жегалкина $P = K_1 \oplus \ldots \oplus K_l \oplus \sigma$, где конъюнкции $K_1, \ldots, K_l$ содержат хотя бы по одной переменной, а σ — булева константа, представляет некоторую функцию из T_1; легко заметить, что l чётно при $\sigma = 1$ и нечётно при $\sigma = 0$. При $l = 0$ константу 1 выражаем через функцию $\varphi(x, y, z) = xy \sim z$ в виде $\varphi(x, x, x)$ (или $xx \sim x$).

Имея $\varphi(x, y, z)$ и константу 1, можем получить конъюнкцию xy в виде $\varphi(x, y, 1)$. Имея конъюнкцию xy, можем реализовать и полином $P_1 = x_{i_1} \cdot \ldots \cdot x_{i_r}$ для произвольного r, $r \geqslant 2$.

Пусть $P_2 = K_1 \oplus K_2 \oplus 1$. Имея K_1 и K_2 (или, точнее, формулы над $\{\varphi(x, y, z)\}$, реализующие K_1 и K_2) возьмём формулу $\varphi(K_1, K_1, K_2)$, реализующую $K_1 \oplus K_2 \oplus 1$.

Пусть $P_3 = K_1 \oplus K_2 \oplus K_3$. Имея K_1 и $K_2 \oplus K_3 \oplus 1$, функцию $K_1 \oplus K_2 \oplus K_3$ реализуем формулой $\varphi(K_1, K_1, K_2 \oplus K_3 \oplus 1)$.

Проводя аналогичным образом индукцию по l и далее, убеждаемся, что любую функцию из T_1 можно реализовать формулой над $\{\varphi(x, y, z)\}$; следовательно, в данном случае $\{xy \sim z\}$ является базисом в T_1.

б) Поскольку $\{xy \vee z\} \subseteq T_1$, то (по свойству замыкания) $[\{xy \vee z\}] \subseteq [T_1] = T_1$, т. е. любая формула над $\{xy \vee z\}$ реализует функцию, сохраняющую 1. Но константа 0 сохраняет нуль и потому содержится в T_0, но не сохраняет единицу и, следовательно, не может быть реализована формулой над $\{xy \vee z\}$; значит, $\{xy \vee z\}$ не является базисом в T_0.

в) Линейные функции 0, 1, x реализуются соответственно формулами $x \oplus x$, $x \sim x$, $x \sim 1$; любая другая линейная функция может быть реализована формулой с использованием $x \oplus y$ и, быть может, константы 1. Таким образом, $[\{x \sim y, x \oplus y\}] = L$. Эквивалентность $x \sim y$ сохраняет единицу и не сохраняет нуль; сумма $x \oplus y$, наоборот, сохраняет нуль и не сохраняет единицу. Следовательно, $x \sim y$ нельзя реализовать формулой над $\{x \oplus y\}$, а $x \oplus y$ нельзя реализовать формулой над $\{x \sim y\}$, т. е. любое собственное подмножество множества B не обладает полнотой в L. В итоге получаем: множество $\{x \sim y, x \oplus y\}$ является базисом в L.

г) При $k \leqslant 1$ формулами над B, очевидно, нельзя реализовать функции, существенно зависящие от двух или более переменных. Если k нечётно,то функция $x_1 \oplus \ldots \oplus x_k$ сохраняет единицу, $B \subseteq T_1$ и, скажем, константу 0 нельзя реализовать формулой над B; следовательно, в данном случае B не может быть базисом в L. Если же k чётно, отождествлением переменных в $x_1 \oplus \ldots \oplus x_k$ получаем константу 0; затем, подставляя константу 0 вместо $x_3, \ldots, x_k$, получаем $x_1 \oplus x_2$; при наличии $x_1 \oplus x_2$ и констан-

ты 1 уже нетрудно реализовать любую функцию из L. Любое собственное подмножество B' множества B уже не обладает полнотой в L (например, $B' = \{x_1 \oplus \ldots \oplus x_k\} \subseteq T_0$, и в этом случае формулой над B' нельзя реализовать константу 1). В итоге получаем, что $\{x_1 \oplus \ldots \oplus x_k, 1\}$ является базисом в L при чётном k, не меньшем двух.

д) Полином любой функции из $T_0 \cap T_1$ содержит нечётное число слагаемых и не содержит 1 в качестве свободного члена. Такой полином можно построить, используя xy и $x \oplus y \oplus z$. В свою очередь, функцию $x \oplus y \oplus z$ можно выразить через $\varphi(x, y, z) = xy \vee x\overline{z} \vee y\overline{z}$ формулой $\varphi(\varphi(y, z, x), \varphi(x, z, y), z)$ (в этом можно убедиться непосредственной проверкой, используя представление $\varphi(x, y, z)$ полиномом $x \oplus y \oplus xy \oplus xz \oplus yz$). Следовательно, $[B] = A$.

С другой стороны, нетрудно убедиться, что $xy \in M$, $xy \notin S$, а $\varphi(x, y, z) \notin M$, $\varphi(x, y, z) \in S$, поэтому $\varphi(x, y, z)$ нельзя выразить через xy, а xy нельзя выразить через $\varphi(x, y, z)$. Следовательно, любая собственная подсистема системы B не обладает полнотой в A.

В итоге получаем, что B является базисом в A.

21. а) Функция f не является шефферовой ни при каких n, поскольку $f \in T_1$ при чётных n и $f \in S$ при нечётных n.

б) При нечётном n функция f не является шефферовой, поскольку $f \in T_1$. При чётных n функция является шефферовой, поскольку отождествлением переменных $x_2, \ldots, x_n$ из функции $f(x_1, \ldots, x_n)$ получаем штрих Шеффера $x_1 x_2 \oplus 1$.

в) Если $n = 4k$ или $n = 4k + 1$, где $k = 0, 1, 2, \ldots$, то полином $1 \oplus \sum\limits_{1 \leqslant i < j \leqslant n} x_i x_j$ либо вовсе не содержит, либо содержит чётное число слагаемых $x_i x_j$ и $f \in T_1$, откуда следует, что f не является шефферовой функцией. Если $n = 4k + 3$, то в этом случае

$$\overline{f}(\overline{x}_1, \ldots, \overline{x}_n) = \sum_{1 \leqslant i < j \leqslant n} (x_i \oplus 1)(x_j \oplus 1) =$$

$$= \sum_{1 \leqslant i < j \leqslant n} x_i x_j \oplus \sum_{i=1}^{n} x_i (\underbrace{1 \oplus \ldots \oplus 1}_{4k+2 \text{ единиц}}) \oplus (\underbrace{1 \oplus \ldots \oplus 1}_{4k+3 \text{ единиц}}) =$$

$$= 1 \oplus \sum_{1 \leqslant i < j \leqslant n} x_i x_j = f(x_1, \ldots, x_n),$$

т. е. $f \in S$ и опять же f не является шефферовой функцией.

Рассмотрим случай $n = 4k + 2$. В этом случае имеем

$$f(x_1, \dots, x_{4k+2}) = 1 \oplus \sum_{1 \leqslant i < j \leqslant 4k+2} x_i x_j =$$
$$= 1 \oplus x_1(x_2 \oplus x_3 \oplus \dots \oplus x_{4k+2}) \oplus \sum_{2 \leqslant i < j \leqslant 4k+2} x_i x_j.$$

В последнем выражении в скобках присутствует нечётное число слагаемых (переменных), а в последней подсумме присутствует чётное число C_{4k+1}^2 слагаемых $x_i x_j$. Поэтому после отождествления переменных $x_2, x_3, \dots, x_{4k+2}$ получаем функцию $f(x_1, x_2, x_2, \dots, x_2) = 1 \oplus x_1 x_2$, которая является шефферовой. Отсюда получаем, что исходная функция является шефферовой при $n = 4k + 2$ $(k = 0, 1, 2, \dots)$.

г) Ни при каких n. д) При чётных n.

22. Вычеркнув из P_2^n все функции, сохраняющие константы, получим множество $A = P_2^n \setminus (T_0^n \cup T_1^n)$. Если $f \in A$, то $f(\widetilde{0}) = 1$, $f(\widetilde{1}) = 0$, поэтому в A нет монотонных функций и $|A| = 2^{2^n-2}$. Самодвойственную функцию f из A можно получить, положив $f(\widetilde{0}) = 1$, задавая значения f на $(2^{n-1} - 1)$ наборах $(0, \sigma_2, \dots, \sigma_n)$ произвольным образом и полагая далее $f(1, \alpha_1, \dots, \alpha_n) = \overline{f}(0, \overline{\alpha}_2, \dots, \overline{\alpha}_n)$; существует всего $2^{2^{n-1}-1}$ таких функций. Вычеркнув из A все самодвойственные функции, получим $B = A \setminus S^n$ и $|B| = 2^{2^n-2} - 2^{2^{n-1}-1}$. Рассмотрим произвольную линейную функцию $f = c_0 \oplus x_{i_1} \oplus \dots \oplus x_{i_k}$, где $x_{i_1}, \dots, x_{i_k} \in \{x_1, \dots, x_n\}$. Если $f \notin T_0^n$, то $c_0 = 1$; далее, если ещё и $f \notin T_1^n$, то k — чётное; наконец, если ещё и $f \notin S$, то k — нечётное. Поскольку получили взаимно исключающие условия, то это означает, что линейные функции в B отсутствуют, т. е. все оказавшиеся в B функции шефферовы и их всего $2^{2^n-2} - 2^{2^{n-1}-1}$ штук.

К главе IV

1. а) $S(x, x, x)$. б) $P(x, x, x) \& \overline{S}(x, x, x)$. в) $(\exists y) S(y, y, x)$.

г) $\overline{P}(x, x, x) \& \overline{\big((\exists y)(\exists z)\,(\overline{P}(y, y, y) \& \overline{P}(z, z, z) \& P(y, z, x))\big)}$.

д) $(\exists z)\,(S(x, z, y) \& S(z, z, z))$. е) $(\exists z)\,\big(\overline{S}(z, z, z) \& S(x, z, y)\big)$.

ж) $(\exists z) P(x, z, y)$.

2. а) Формула $(\forall y)\,\big(Q(x, x) \& \overline{Q}(x, y)\big)$ ложна в любой модели (т. е. принимает значение 0 на всех наборах значений

своих свободных переменных; в рассматриваемой формуле свободной является переменная x), поскольку при $y = x$ получаем $Q(x,x)\&\overline{Q}(x,x) = 0$. Следовательно, в любой интерпретации $(\forall x)(\forall y)\left(Q(x,x)\&\overline{Q}(x,y)\right) = 0$.

б) Формула принимает значение 1, например, в модели, для которой $M = \{a\}$ (т. е. предметная область содержит единственный элемент), а $R(a,a,a) = 1$.

в) Формула Ф будет истинной в любой модели, в которой $R(y)$ — истинная формула.

3. а) Формула Ф принимает значение 0, например, в модели, для которой $M = \{a,b\}$ и $P(a) = 0$, $P(b) = 1$.

б) Формула Ф принимает значения 0 в модели с предметной областью M из одного элемента.

в) Формула Ф обращается в нуль в модели с предметной областью $M = \{a,b\}$ и предикатом $P(x,y)$, обращающимся в 1 на наборах (a,a), (b,b) и в нуль на наборах (a,b), (b,a).

4. а) Формула Ф обращается в 0 в модели, представленной в решении задачи 3.в), и не является общезначимой.

б) Пусть имеем произвольную модель $\mathcal{D}$ с предметной областью M и определённым на M предикатом $P(x,y)$. Если в этой модели выполнено равенство $(\exists x)(\exists y)P(x,y) = 0$, то и формула Ф с учётом правил переименования связанных переменных и выноса кванторов за скобки оказывается в этой модели истинной.

Предположим, что $(\exists x)(\forall y)P(x,y) = 1$ в модели $\mathcal{D}$. Пусть T — таблица, задающая предикат $P(x,y)$, и строкам этой таблицы отвечают значения переменной x, а столбцам — значения переменной y. Легко заметить, что в рассматриваемом случае в T найдётся по крайней мере одна строка, заполненная одними единицами. Но тогда в данной модели при любом значении y формула $(\exists x)P(x,y)$ принимает значение 1 и потому $(\forall y)(\exists x)P(x,y) = 1$, т. е. исходная формула Ф обращается в единицу; значит, Ф общезначима.

5. а) $(\forall x)(\exists y)\left(\overline{P}(x) \vee Q(x,y)\right)$.

б) Для заданной формулы Ф эквивалентную ей нормальную формулу можем получить, например, следующими преобразованиями:

$$(\exists x)P(x) \vee (\forall y)\left(P(y) \to \overline{(\exists x)Q(x,y)}\right) =$$

$$= (\exists x)P(x) \vee (\forall y)\left(\overline{P}(y) \vee \overline{(\exists z)Q(z,y)}\right) =$$

$$= (\exists x)P(x) \vee (\forall y)\left(\overline{P}(y) \vee (\forall z)\overline{Q(z,y)}\right) =$$

$$= (\exists x)P(x) \vee (\forall y)(\forall z)\left(\overline{P}(y) \vee \overline{Q}(z,y)\right) =$$
$$= (\exists x)(\forall y)(\forall z)\left(P(x) \vee \overline{P}(y) \vee \overline{Q}(z,y)\right).$$

в) $(\forall x)(P(x) \to Q(x,x)) \to ((\forall y)P(y) \sim (\exists y)Q(y,z)) =$
$$= \overline{(\forall x)(\overline{P}(x) \vee Q(x,x))} \vee ((\forall y)P(y))\&((\exists y)Q(y,z)) \vee$$
$$\vee \overline{(\forall y)P(y)}\&\overline{(\exists y)(Q(y,z))} =$$
$$= (\exists x)(P(x)\&\overline{Q}(x,x)) \vee ((\forall y)P(y))\&((\exists u)Q(u,z)) \vee$$
$$\vee \overline{(\forall v)P(v)}\&\overline{(\exists t)Q(t,z)} =$$
$$= (\exists x)(P(x)\&\overline{Q}(x,x)) \vee (\forall y)(\exists u)(P(y)\&Q(u,z)) \vee$$
$$\vee (\exists v)(\forall t)(\overline{P}(v)\&\overline{Q}(t,z)) =$$
$$= (\exists x)(\forall y)(\exists u)(\exists v)(\forall t)(P(x)\overline{Q}(x,x) \vee P(y)Q(u,z) \vee \overline{P}(v)\overline{Q}(t,z)).$$

г) $(\exists x)(\forall y)P(x,y) \to (\exists x)(\forall z)Q(x,z) =$
$$= \overline{(\exists x)(\forall y)P(x,y)} \vee (\exists x)(\forall z)Q(x,z) =$$
$$= (\forall x)\left(\overline{(\forall y)P(x,y)}\right) \vee (\exists u)(\forall z)Q(u,z) =$$
$$= (\forall x)(\exists y)\overline{P}(x,y) \vee (\exists u)(\forall z)Q(u,z) =$$
$$= (\forall x)(\exists y)(\exists u)(\forall z)\left(\overline{P}(x,y) \vee Q(u,z)\right).$$

6. а) Формула $(\forall x)(\exists y)P(x,y)$ истинна в модели $\mathcal{D}_1$ (табл. 20) (и не является истинной в модели $\mathcal{D}_2$, табл. 21).

Таблица 20

x \ y	a	b
a	0	1
b	1	0

Таблица 21

x \ y	a	b
a	1	1
b	0	0

б) Формула $P(x) \to (\forall y)P(y)$ истинна на одноэлементном множестве (и не является истинной на множестве из большего числа элементов).

в) Формула $P(x) \to (\exists y)P(y)$ общезначима.

7. а) Зададим предикаты $S(x,y,z)$ и $P(x,y,z)$ из задачи 1 на множестве $M = \{0,1,2,\ldots\}\backslash\{1\}$; в этой модели формулы $S(x,x,x)$ и $P(x,x,x)$ эквивалентны.

б) Формулы $(\exists x)(\exists y)(\exists z)P(x,y,z)$ и $(\forall x)(\forall y)(\forall z)P(x,y,z)$ эквивалентны на множестве из одного элемента.

в) Формулы $(P(x) \to Q(y))\&(Q(y) \to P(x))$ и $P(x) \sim Q(y)$ эквивалентны.

8. а) Обе формулы F_1 и F_2 истинны в (произвольной) модели $\mathcal{D}$ в том и только в том случае, когда предикат $P(x, y)$ принимает значение 1 на любом наборе значений переменных; следовательно, эти формулы эквивалентны.

б) Формулы F_1 и F_2 эквивалентны, поскольку они обе одновременно ложны в модели $\mathcal{D}$ в том и только в том случае, когда предикат $P(x, y)$ обращается в 0 на любом наборе значений переменных.

9. Возьмём модель $\mathcal{D}$ с множеством $M = \{a, b\}$ и предикатом $P(x, y)$, определённым на M следующим образом: $P(a, a) = P(b, a) = 1$, $P(a, b) = P(b, b) = 0$. В этой модели первая формула ложна, а вторая истинна; следовательно, эти формулы неэквивалентны.

10. Пусть на множестве $M = \{a_1, \ldots, a_n\}$ определён предикат $P(x)$. Тогда $(\forall x)P(x) = \mathop{\&}\limits_{i=1}^{n} P(a_i)$, а $(\exists x)P(x) = \mathop{\vee}\limits_{i=1}^{n} P(a_i)$.

11. а) Предикаты $P(x)$ и $Q(x)$ можно определить на множестве $M = \{a_1, \ldots, a_n\}$, $n \geqslant 2$, например, так: $P(a_1) = Q(a_2) = \ldots = Q(a_n) = 1$, $Q(a_1) = P(a_2) = \ldots = P(a_n) = 0$.

в) Предикат $P(x, y)$ определим на множестве $M = \{a_1, \ldots, a_n\}$, $n \geqslant 2$, так:

$$P(a_i, a_j) = \begin{cases} 1 \text{ при } i = j, \\ 0 \text{ при } i \neq j. \end{cases}$$

К главе V

1. б) Верхняя оценка устанавливается конструктивно — нетрудно построить схемы, содержащие по четыре элемента и реализующие функции $x \oplus y$, $x \sim y$ (одна из таких схем, реализующая $x \oplus y$, представлена на рис. 10). Докажем нижнюю оценку $\min\left(L_{\{\&,\vee,-\}}(x \oplus y), L_{\{\&,\vee,-\}}(x \sim y)\right) \geqslant 4$. Пусть S — минимальная (т. е. содержащая наименьшее возможное число элементов) схема, реализующая линейную функцию от двух переменных. Очевидно, $L(S) > 1$. Пусть E_1 — выходной элемент схемы S. Предположим, что на вход элемента E_1 подаётся переменная, например, x. Обозначим через $\chi(E)$ «забивающую» для E константу, положив

$$\chi(E) = \begin{cases} 0, \text{ если } E \text{ — конъюнктор,} \\ 1, \text{ если } E \text{ — дизъюнктор или инвертор.} \end{cases}$$

При подаче на вход схемы, отвечающий переменной x, забивающей для E_1 константы $\chi(E_1)$ на выходе схемы получим также

константу, а должна быть функция $\chi(E_1) \oplus y \oplus \alpha$, существенно зависящая от y. Получаем противоречие, исключающее наше предположение о подаче переменной на вход выходного элемента E_1.

Предположим, что входы элемента E_1 соединены лишь с выходом элемента E_2. В этом случае элемент E_2 реализует, как нетрудно заметить, линейную функцию $x \oplus y \oplus \alpha$ (если E_1 — конъюнктор или дизъюнктор) или $x \oplus y \oplus \alpha \oplus 1$ (если E_1 — инвертор). Удалив из S элемент E_1, получим схему S' меньшей сложности с выходным элементом E_2, реализующую линейную функцию от двух переменных, а это противоречит предположению о том, что исходная схема S имеет наименьшую сложность среди всех схем, реализующих линейные функции от двух переменных.

Остаётся рассмотреть случай, когда E_1 имеет два входа (т. е. не является инвертором) и один из этих входов соединён с выходом элемента E_2, а второй — с выходом третьего элемента E_3. Предположим, что иных элементов в S нет. Пусть, далее, хотя бы один из элементов E_2, E_3, например, E_2 — инвертор. В этом случае вход элемента E_2 не может соединяться с выходом элемента E_3, поскольку при таком соединении на выходе элемента E_1, как нетрудно заметить, оказалась бы реализованной булева константа. Допустим, что вход инвертора E_2 соединён с некоторым входом схемы S, например, отвечающим переменной x. В этом случае при подаче вместо x константы $\overline{\chi}(E_1)$ на выходе элемента E_2 и затем на выходе элемента E_1 (т. е. на выходе схемы) опять же получаем константу $\chi(E_1)$. Предположим, что среди элементов E_1, E_2, E_3 нет инверторов. Но тогда на выходе схемы получим функцию, реализуемую формулой над множеством $\{x\&y, x \vee y\}$, т. е. монотонную функцию (в силу замкнутости класса монотонных функций — см. задачу 11 к главе III), тогда как $x \oplus y \oplus \alpha$ — немонотонная функция.

Таким образом, схема S должна содержать более трёх элементов, и требуемая нижняя оценка выполняется.

2. а), б) *Указание.* Верхние оценки получаются конструктивно, т. е. в базисе $\{\&, \oplus, 1\}$ строятся схемы, «моделирующие» формулы $x \oplus y \oplus 1$, $(x\&y) \oplus 1$, $(x\&y) \oplus x \oplus y$, содержащие по три элемента и реализующие функции $x \sim y$, x/y, $x \vee y$. Для обоснования требуемых нижних оценок можно просто воспользоваться перебором всевозможных схем из двух элементов и убедиться, что ни одна из функций $x \sim y$, x/y, $x \vee y$ не может быть реализована схемой из двух элементов. В ряде случаев оказывается воз-

можным существенно сократить перебор, используя те или иные свойства рассматриваемых схем и реализуемых функций. Скажем, функции $x \sim y$, x/y не сохраняют 0, а базисные функции $x\&y$, $x \oplus y$ сохраняют 0, поэтому можно утверждать (используя результат задачи 11 к главе III), что схемы в рассматриваемом базисе, реализующие функции $x \sim y$, x/y, должны содержать элемент «1» (реализующий константу 1); далее почти очевидно, что, добавляя к элементу «1» ещё один элемент («&» или «$\oplus$»), нельзя получить схему, реализующую $x \sim y$ или x/y.

в) Представленная на рис. 41 схема содержит 4 элемента и реализует $x \to y$; отсюда следует верхняя оценка $L_{\{\&,\oplus,1\}}(x \to y) \leqslant 4$. Нижнюю оценку $L_{\{\&,\oplus,1\}}(x \to y) \geqslant 4$ докажем от противного. Предположим, что существует минимальная схема S, содержащая не более трёх элементов и реализующая $x \to y$. Учитывая результат задачи 11 к главе III, замечаем: $x \to y \notin L$, а $x \oplus y$, $1 \in L$ и потому S содержит элемент «&» (конъюнктор); $x\&y$, $1 \in M$, а $x \to y \notin M$ и потому S содержит элемент «$\oplus$»; $x\&y$, $x \oplus y \in T_0$, а $x \to y \notin T_0$, и потому S содержит элемент «1».

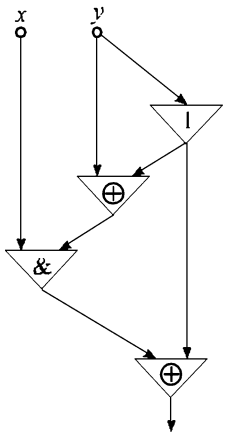


Рис. 41

Элемент «1», очевидно, не может быть выходным элементом в схеме S. Предположим, что выходной элемент E_1 в S — конъюнктор. Переменная x подаваться на вход элемента E_1 не может, поскольку при $x \equiv 0$ на выходе схемы получим 0, а должно быть 1. Если на вход элемента E_1 подаётся переменная y, то снова получаем противоречие: при $y \equiv 0$ на выходе схемы получим 0, а должна быть реализована функция $\overline{x}$. Если вход элемента E_1, например, правый, соединён с выходом элемента «1», то в этом случае левый вход элемента E_1 должен быть соединён с выходом элемента E, которому приписана функция $x \to y$; но в этом случае элемент E_1 можно удалить и получить схему с выходным элементом E, реализующую $x \to y$, а это противоречит предположению о минимальности S. Предположение о том, что оба входа элемента E_1 соединены с выходом элемента «$\oplus$», приводит снова к противоречию, поскольку в этом случае элемент E_1 опять можно будет удалить и схема S окажется неминимальной.

Предположим теперь, что выходной элемент E_1 — это элемент «$\oplus$». В этом случае можем считать, что вход одного из

элементов «&», «1» с выходом другого в схеме S не соединяется. Действительно, если вход элемента «&» соединён с выходом элемента «1», то элемент «&» можно будет удалить из схемы S, а если вход элемента «1» соединён с выходом элемента «&», то его можно отсоединить от выхода элемента «&» и соединить с любым входом схемы. Вместе с тем в силу минимальности схемы S выходы элементов «&» и «1» должны соединяться со входами других элементов (иначе эти элементы, очевидно, можно было бы удалить из S); значит, выходы элементов «&», «1» соединены в S со входами элемента E_1. Кроме того, входы элемента «&» должны быть соединены со входами схемы, причём с разными (при подаче одной и той же переменной на оба входа элемента «&» этот элемент реализует данную переменную и его, очевидно, можно будет удалить из схемы). Но в таком случае на выходе схемы S будет реализована функция $x\&y$, а не $x \to y$.

В итоге получаем, что схемы S, реализующей $x \to y$ и содержащей не более трёх элементов, не существует.

3. в) Верхние оценки $L_{\{/\}}(\overline{x}\,\overline{y}) \leqslant 4$ и $L_{\{/\}}(x \oplus y) \leqslant 4$ выполняются для схем, представленных на рис. 42.

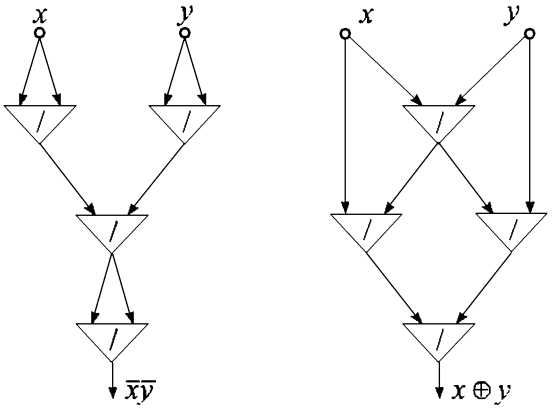


Рис. 42

Докажем оценку $L_{\{/\}}(\overline{x}\,\overline{y}) \geqslant 4$ от противного. Предположим, что существует минимальная схема S с выходным элементом E_1, реализующая функцию $\overline{x}\,\overline{y}$ и содержащая не более трёх элементов. Предположим, что на вход элемента E_1 подаётся какая-нибудь переменная, например, x. В таком случае при $x \equiv 0$ на выходе схемы будет реализована константа 1, а должна быть функция $\overline{y}$; получаем противоречие. Значит, на входы элемента E_1 переменные не подаются. Если бы S содержала всего два

элемента E_1 и E_2, то оба входа элемента E_1 соединялись бы с выходом элемента E_2, а на входы элемента E_2 подавались бы обе переменные x и y (в силу существенной зависимости выходной функции $\overline{x}\,\overline{y}$ схемы S от обеих переменных). Но в таком случае на выходе схемы получили бы xy, а не $\overline{x}\,\overline{y}$; следовательно, двумя элементами ограничиться нельзя и схема S содержит три элемента E_1, E_2, E_3.

Введём монотонную нумерацию вершин схемы S, в соответствии с которой элементы E_1, E_2, E_3 получат некоторые номера n_1, n_2, n_3; без ограничения общности можем считать, что $n_1 > n_2 > n_3$. В силу минимальности схемы S выходы элементов E_2, E_3 должны соединяться со входами других элементов, причём выход элемента E_2 должен соединяться хотя бы с одним входом элемента E_1. Возможны два случая: оба входа элемента E_1 соединены с выходом элемента E_2; один вход элемента E_1 соединён с выходом элемента E_2, а другой — с выходом элемента E_3.

В первом случае выход элемента E_3 должен быть соединён хотя бы с одним входом элемента E_2. Если выход элемента E_3 соединён с обоими входами элемента E_2, то тогда на входы E_3 должны подаваться обе переменные x, y и на выходе схемы получим x/y, а не $\overline{x}\,\overline{y}$. Предположим, что выход E_3 соединён с одним входом элемента E_2, а на другой вход E_2 подаётся переменная, скажем, x. Тогда либо y подаётся на оба входа E_3, либо на входы E_3 подаются обе переменные x, y; в обоих случаях на выходе схемы получим $x\overline{y}$, но не $\overline{x}\,\overline{y}$. Таким образом, условия первого случая приводят к противоречию.

Рассмотрим второй случай. Если на оба входа элемента E_3 подаётся одна и та же переменная, например, x, то при $x \equiv 1$ на выходе схемы получим константу 1, а должно быть $\overline{y}$. Если на входы E_3 подаются обе переменные x, y, то при $x = y = 1$ на выходе схемы вместо требуемого нуля окажется единица. Снова получаем противоречие, которое исключает этот случай, а вместе с ним и исходное предположение.

Докажем неравенство $L_{\{/\}}(x \oplus y) \geqslant 4$. Предположим, что существует минимальная схема S, реализующая линейную функцию $x \oplus y \oplus \alpha$, $\alpha \in \{0, 1\}$, и содержащая не более трёх элементов. Пусть E_1 — выходной элемент схемы S. Если на вход элемента E_1 подаётся переменная, скажем, x, то при $x \equiv 0$ на выходе схемы получим константу 1, а должна быть реализована функция $y \oplus \alpha$; значит, на входы элемента E_1 переменные не подаются.

Предположим, что какая-нибудь переменная, например, x подаётся на входы лишь одного какого-то элемента E. Если x

подаётся на оба входа элемента E, то на выходе этого элемента реализуется $\overline{x}$. В этом случае элемент E удалим, а соединённые в S с выходом элемента E входы других элементов соединим со входом схемы, отвечающим переменной x. Полученная схема реализует $x \oplus y \oplus \alpha \oplus 1$, т. е. линейную функцию от переменных x, y, и оказывается проще исходной схемы, что противоречит предположению о минимальности S.

Если x подаётся на один вход v элемента E, то на другой вход v' этого элемента должна подаваться константа 1 — иначе можно будет подобрать такое значение (константу) для y, при котором значение на выходе элемента E, а значит, и на выходе всей схемы S не будет зависеть от x, а это противоречит предположению о том, что S реализует линейную функцию от x, y. При подаче на входы v, v' переменной x и константы 1 на выходе элемента E будет реализована инверсия $\overline{x}$, а такой вариант уже рассмотрен выше (он приводит к противоречию). Следовательно, каждая из переменных x, y подаётся на входы не менее, чем двух элементов. Но в схеме S, помимо выходного элемента E_1, могут быть ещё только два элемента E_2, E_3, причём на входы этих элементов должны подаваться переменные x, y, а значит, и на выходах элементов E_2, E_3 будет реализована одна и та же функция x/y, а это в минимальной схеме невозможно.

Полученное в итоге противоречие опровергает наше исходное предположение о том, что S реализует линейную функцию от x, y и содержит не более трёх элементов.

4. Дизъюнкцию $x_1 \vee \ldots \vee x_n$ можно реализовать схемой из $2n$ элементов в базисе $\{\&, {}^-\}$, моделируя представление дизъюнкции в виде $x_1 \vee \ldots \vee x_n = \overline{\overline{x}_1 \& \ldots \& \overline{x}_n}$; в этой схеме вначале с использованием n инверторов реализуем инверсии всех переменных, которые затем перемножаем с использованием $(n-1)$ конъюнкторов, и полученный результат «инвертируем» с использованием ещё одного инвертора. Аналогичным образом конъюнкцию $x_1 \& \ldots \& x_n$ реализуем схемой из $2n$ элементов в базисе $\{\vee, {}^-\}$, моделируя представление $x_1 \& \ldots \& x_n = \overline{\overline{x}_1 \vee \ldots \vee \overline{x}_n}$. Далее покажем, что эти схемы минимальны.

Докажем вначале неравенство $L_{\{\&, {}^-\}}(x \vee y) \geqslant 4$. Предположим, что существует минимальная схема S с выходным элементом E_1, реализующая $x \vee y$ и содержащая не более трёх элементов. Предположим, что E_1 — инвертор. В этом случае вход элемента E_1 должен соединяться с выходом конъюнктора E_2, поскольку переменная на вход инвертора E_1 подаваться, очевидно, не может, а при соединении входа E_1 с выходом инвертора

схема S оказалась бы неминимальной. Если на вход конъюнктора E_2 подаётся переменная, скажем, x, то при $x \equiv 0$ на выходе схемы получим 1, а должно быть y. При соединении входов E_2 с выходом одного и того же элемента E_3 схема окажется неминимальной. Следовательно, тремя элементами в S в данном случае ограничиться нельзя.

Допустим, что элемент E_1 — конъюнктор. Переменные на входы E_1 в данном случае, как нетрудно заметить, подаваться не могут. В силу минимальности схемы оба входа элемента E_1 не могут соединяться с выходом одного и того же элемента. Следовательно, в S имеются элементы E_2, E_3, входы которых соединены со входами E_1. Среди элементов E_2, E_3 хотя бы один, например, E_2 удовлетворяет условию: входы этого элемента не соединяются с выходами другого. Пусть на вход E_2 подаётся переменная, скажем, x. Если E_2 — конъюнктор, то при $x \equiv 0$ на выходе схемы получим 0, а не y (как требуется); если же E_2 — инвертор, то при $x = y = 1$ на выходе схемы вместо требуемой единицы окажется 0. Значит, переменные на входы E_2 подаваться не могут, а это означает, что в схеме должны быть отличные от E_1, E_3 элементы, выходы которых соединены со входами E_2.

Получаем противоречие, исключающее наше исходное предположение о том, что $L_{\{\&,-\}}(x \vee y) \leqslant 3$.

Нижнюю оценку $L_{\{\&,-\}}(x_1 \vee \ldots \vee x_n) \geqslant 2n$ при $n \geqslant 3$ нетрудно теперь получить индукцией по n, если воспользоваться неравенством $L_{\{\&,-\}}(x \vee y) \geqslant 4$ (в качестве основания индукции) и следующим утверждением: из любой минимальной схемы S, реализующей $x_1 \vee \ldots \vee x_n$, $n \geqslant 3$, можно удалить не менее двух элементов и получить (возможно, после изменения соединений оставшихся элементов) схему S', реализующую $x_1 \vee \ldots \vee x_{n-1}$.

Докажем последнее утверждение. Отметим следующее почти очевидное свойство схем в рассматриваемом базисе: если в схеме S, реализующей отличную от константы функцию f, имеется элемент E, реализующий булеву константу, то этот элемент E можно удалить из S и получить схему, реализующую ту же функцию f. Формально это свойство можно обосновать, например, так. Добавим в S две новые вершины v_0, v_1, в которых реализуются соответственно константы 0 и 1 (это могут быть либо дополнительные входы схемы, на которые вместо переменных подаются константы, либо специальные отдельные схемы S_0, S_1, реализующие в вершинах v_0, v_1 булевы константы). Если в S имеется конъюнктор E и на вход его подаётся константа 0, то

удалим E, а входы других элементов, соединённые с выходом E, соединим с дополнительным входом v_0. Если на какой-то вход конъюнктора E подаётся константа 1, а второй вход этого конъюнктора соединён в S с вершиной v, то после удаления E входы других элементов схемы, соединённые с выходом E, соединим с v. Через конечное число шагов (в силу конечности схемы) вместо S получаем реализующую прежнюю функцию f схему S', в которой не останется элементов, на входы которых подаются булевы константы; после этого можно отбросить и дополнительные входы v_0, v_1.

Выделим теперь в схеме S элемент E, на вход которого подаётся x_n. Элемент E, очевидно, не может выть выходным элементом схемы S, и его выход в силу минимальности S должен соединяться со входом ещё какого-то хотя бы одного элемента E'. При $x_n \equiv 0$ на входы элементов E, E' будут поданы константы, и эти (а возможно, и ещё какие-то) элементы по вышеуказанному свойству схем можно удалить из S и получить схему S', реализующую $x_1 \vee \ldots \vee x_{n-1}$.

Нижняя оценка $L_{\{\vee,-\}}(x_1 \& \ldots \& x_n) \geqslant 2n$ доказывается аналогичным образом.

5. а) Пусть S — минимальная схема в базисе $\{\&, \vee, {}^-\}$, реализующая булеву функцию $f(x_1, \ldots, x_n)$ и содержащая $L_{\{\&,\vee,-\}}(f(x_1, \ldots, x_n))$ элементов. Пусть v — произвольная вершина схемы S, отвечающая входу схемы или двухвходовому элементу. Если в схеме S нет инвертора, вход которого соединён с вершиной v, то добавим такой инвертор. После добавления всех необходимых инверторов получим схему S', которая по-прежнему реализует $f(\widetilde{x})$ и сложность которой не превосходит $2L(S) + n$, т. е. $2L_{\{\&,\vee,\ \}}(f(\widetilde{x})) + n$.

Пусть $E^\vee$ — какой-то дизъюнктор в S, входы которого соединены с вершинами u, v, выход — с вершинами $v_1, \ldots, v_k$, причём вершина v_1 — инвертор, «подвешенный» в S' к E. В соответствии с тождеством $x \vee y = \overline{\overline{x}\&\overline{y}}$ дизъюнктор $E^\vee$ заменим на конъюнктор $E^\&$, причём входы $E^\&$ соединим с вершинами, связанными с u, v, выход $E^\&$ соединим с теми вершинами, которые в схеме S' соединялись с v_1 (т. е. с выходом инвертора, подвешенного в S' к $E^\vee$), а вершину v_1 (выход инвертора) соединим с вершинами $v_2, \ldots, v_k$; пример указанной замены показан на рис. 43. Нетрудно заметить, что при такой замене дизъюнкторов на конъюнкторы в получающейся схеме S'' контуры не могут появиться и в вершинах схемы S'' будут реализованы все те функции, которые

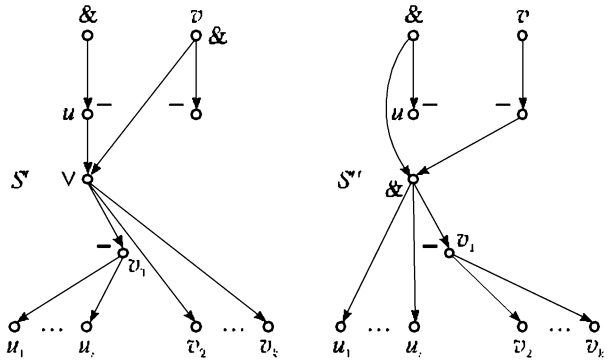


Рис. 43

реализованы в вершинах схемы S' (в том числе и функция $f(\widetilde{x})$), а $L(S'') = L(S') \leqslant 2L_{\{\&,\vee,-\}}(f(\widetilde{x})) + n$.

Оценка

$$L_{\{\vee,\ \}}(f(x_1, \ldots, x_n)) \leqslant L_{\{\&,\vee,\ \}}(f(x_1, \ldots, x_n)) + n$$

устанавливается аналогичным образом.

7. в) Используя $n-2$ элементов, реализуем $f_1 = x_2 \oplus x_3 \oplus \ldots \oplus x_n$. Добавим в схему ещё один элемент и получим $f = x_1 \oplus f_1 = x_1 \oplus x_2 \oplus \ldots \oplus x_n$. Каждую из последующих функций f_i, $i = 2, 3, \ldots, n$, реализуем в соответствии с представлением $f_i = x_i \oplus f$; при этом для каждой из реализуемых функций f_i в схему добавляем по одному элементу. В итоге получим схему из $2n-2$ элементов, реализующую заданную систему функций; отсюда вытекает верхняя оценка $L_{\{\oplus\}}(N_n) \leqslant 2n-2$ $(n \geqslant 4)$.

Докажем нижнюю оценку $L_{\{\oplus\}}(N_n) \geqslant 2n-2$. Пусть S — произвольная минимальная схема из элементов «$\oplus$», реализующая систему N_n. Зафиксируем для S какую-нибудь монотонную нумерацию вершин, при которой выходные элементы $E_1, \ldots, E_n$ схемы получат номера $a_1, \ldots, a_n$; без ограничения общности можем считать, что на выходах этих элементов реализуются соответственно функции $f_1, \ldots, f_n$, причём $a_1 < a_2 < \ldots < a_n$. Легко заметить, что для реализации линейной функции от k переменных требуется не менее $k-1$ элементов (формально это легко доказать индукцией по k). Поэтому подсхема S' исходной схемы S, содержащая элемент E_1 и все остальные элементы из S с номерами, меньшими чем a_1, реализует f_1 и содержит не менее $n-2$ элементов. Если S' содержит более, чем $n-2$ элементов, то для S выполняется требуемая оценка $L(S) \geqslant 2n-2$.

Предположим, что S' содержит ровно $n-2$ элементов и входы элемента E_1 соединены с вершинами u, v. Рассмотрим два возможных случая: u и v — элементы; одна из вершин, скажем, u — вход схемы, а вторая — элемент «$\oplus$».

В первом случае в вершине u реализуется линейная функция φ_u от a переменных, а в вершине v — линейная функция φ_v от b переменных, причём $a, b \geqslant 2$ и $a+b=n-1$. При сложении (по модулю 2) любых двух функций, реализованных в вершинах подсхемы S', получается линейная функция от переменных из $\{x_2, \ldots, x_n\}$, т. е. никак не f_2. Учитывая, что f_1 зависит от $n-1$ переменных, а всякая другая функция, реализуемая в S', зависит не более, чем от $n-3$ переменных, нетрудно заметить, что функцию f_2 нельзя получить сложением переменной с какой-нибудь функцией, реализуемой в вершине подсхемы S'. Следовательно, в S имеется элемент с номером a', где $a_1 < a' < a_2$, выход которого соединён со входом элемента E_2, и $L(S) \geqslant 2n-2$.

Рассмотрим второй случай, когда входы элемента E_1 соединены со входом u схемы, отвечающим, скажем, переменной x_n, и с выходом некоторого элемента E. В этом случае в вершине v реализуется $x_2 \oplus \ldots \oplus x_{n-1}$, а во всех остальных вершинах подсхемы S' реализуются линейные функции от переменных из множества $\{x_2, \ldots, x_{n-1}\}$, каждая из которых зависит не более, чем от $n-3$ переменных. Если в схеме S имеется невыходной элемент E с номером, бóльшим a_1, то в этом случае требуемая нижняя оценка выполняется. Предположим, что такого элемента E в S нет. В таком случае сложением двух функций, реализуемых на входах схемы, в вершинах подсхемы S' и на выходе элемента E_1, можно получить только одну функцию из N_n, а именно, $f_n = x_1 \oplus x_2 \oplus \ldots \oplus x_{n-1}$. Но далее, как нетрудно заметить, уже ни одной функции f_i из $\{f_2, \ldots, f_{n-1}\}$ нельзя получить сложением двух функций φ, φ', каждая из которых является либо переменной, либо одной из функций, реализуемых в вершинах подсхемы S' и на выходе элемента E_2. Полученное противоречие исключает наше последнее предположение.

8. б) Пусть $f(x_1, \ldots, x_n)$ — произвольная функция из $Q_{n,k}$. Представим f полиномом Жегалкина P и разобьём слагаемые из P на три подсуммы P_1, P_2, P_3. В P_1 соберём конъюнкции первого ранга x_i, в P_2 — конъюнкции второго ранга $x_i x_j$, в P_3 — конъюнкции третьего ранга $x_i x_j x_k$, $1 \leqslant i < j < k \leqslant n$. Слагаемые из P_3, в свою очередь, разобьём на подсуммы $P_{i,j}$, $i < j$, помещая в $P_{i,j}$ все конъюнкции $x_i x_j x_k$, $k \in \{j+1, \ldots, n\}$, содержащие x_i и x_j (какие-то подсуммы $P_{i,j}$ могут оказаться пусты-

ми). Далее каждую подсумму $P_{i,j}$ представим в виде $x_i x_j P'_{i,j}$, где $P'_{i,j} = x_{k_1} \oplus x_{k_2} \oplus \ldots \oplus x_{k_r}$, $1 \leqslant i < j < k_1 < k_2 < \ldots < k_r$, $r \leqslant n-2$. В соответствии с полученным представлением $f(\widetilde{x})$ построим схему S.

Подсумму P_1, очевидно, можно получить на выходе подсхемы S_1, содержащей не более $n-1$ элементов «$\oplus$». Подсумма P_2 содержит не более $C_n^2 = \dfrac{n(n-1)}{2}$ слагаемых. Эту подсумму получим на выходе подсхемы S_2, содержащей менее чем $n(n-1)$ элементов; в эту подсхему войдут C_n^2 конъюнкторов, реализующих все возможные конъюнкции второго ранга, и не более чем $(C_n^2 - 1)$ элементов «$\oplus$», с помощью которых получается собственно подсумма P_2.

Множество переменных $\{x_1, \ldots, x_n\}$ разобьём на подмножества $\Gamma_1, \Gamma_2, \ldots, \Gamma_m$, каждое из которых, кроме, быть может, Γ_m, содержит l переменных, где $l = \lfloor \log n \rfloor$, $m = \left\lceil \dfrac{n}{l} \right\rceil$. Используя не более чем $(l-1) \cdot 2^l$ элементов «$\oplus$», построим подсхему $S_{3,1}$, реализующую всевозможные подсуммы переменных из множества Γ_1 вида $x_{i_1} \oplus x_{i_2} \oplus \ldots \oplus x_{i_a}$, где $1 \leqslant i_1 < i_2 < \ldots < i_a \leqslant l$. Аналогичным образом далее построим подсхемы $S_{3,2}, \ldots, S_{3,m}$ для реализации всевозможных подсумм переменных соответственно из подмножеств $\Gamma_2, \ldots, \Gamma_m$. Во всех подсхемах $S_{3,1}, \ldots, S_{3,m}$ окажется менее чем $ml \cdot 2^l$ элементов. В соответствии с разбиением переменных $x_1, \ldots, x_n$ на подмножества $\Gamma_1, \ldots, \Gamma_m$ разобьём и $P'_{i,j}$ на подсуммы (их будет не более m), уже полученные на выходах подсхем $S_{3,1}, \ldots, S_{3,m}$. В соответствии с этим разбиением построим подсхему $S'_{i,j}$, содержащую не более $m-1$ элементов и реализующую $P'_{i,j}$; входы подсхемы $S'_{i,j}$ соединяются с выходами подсхем $S_{3,1}, \ldots, S_{3,m}$. Для построения подсхем $S'_{i,j}$ потребуется не более чем $(m-1)C_n^2$ элементов.

Последнюю подсхему S_4 строим из элементов «$\oplus$» для сложения выходных функций подсхем $S_1, S_2, S'_{i,j}$ $(1 \leqslant i < j \leqslant n$; $S'_{i,j}$ — непустая подсхема); эта подсхема содержит не более, чем $C_n^2 + 1$ элементов и реализует $f(\widetilde{x})$.

Используя верхние оценки сложности всех подсхем, для сложности построенной схемы S получаем оценку

$$L(S) \leqslant L(S_1) + L(S_2) + \sum_{1 \leqslant i < j \leqslant n} L(S'_{i,j}) \leqslant$$

$$\leqslant n - 1 + C_n^2 - 1 + (m-1)C_n^2 + C_n^2 + 1 < \frac{n(n-1)}{2}(m+1) + n,$$

из которой с учётом выбранных значений параметров m, l получаем

$$L(S) \lesssim \frac{n^3}{2\log n}.$$

в) Верхняя оценка $L_{\{\&,\oplus,1\}}(f_n) \lesssim \dfrac{n^2}{\log n}$ — результат задачи 8.а) (получается аналогично верхней оценке из задачи 8.б)). Нижняя оценка $\max\limits_{f_n \in Q_{n,2}} L_{\{\&,\oplus,1\}}(f_n) \gtrsim \dfrac{n^2}{\log n}$ получается с использованием мощностных соображений. Полиномы Жегалкина, представляющие функции из $Q_{n,2}$, могут содержать в качестве слагаемых константу 1, конъюнкции первого ранга x_i, $i \in \{1, \ldots, n\}$, и конъюнкции второго ранга $x_i x_j$ $(1 \leqslant i < j \leqslant n)$. Всего таких слагаемых $C_n^2 + n + 1$ штук; следовательно, различных полиномов, а значит и функций в $Q_{n,2}$, имеется $2^{C_n^2+n+1}$ штук. Как аналог леммы 6 в данном случае, справедливо следующее утверждение: если для некоторой функции $k(n)$ при $n \to \infty$ выполнено условие

$$\frac{N(k(n))}{2^{C_n^2+n+1}} \to 0, \tag{1}$$

то, начиная с некоторого n, имеет место оценка $L(Q_{n,2}) > k(n)$, причём доля тех функций $f(x_1, \ldots, x_n)$ из $Q_{n,2}$, для которых $L(f) < k(n)$, стремится к 0 с ростом n. Используя лемму 7, непосредственной проверкой убеждаемся, что условие (1) выполняется для $k(n) = \dfrac{n^2}{5\log n}$ и требуемая нижняя оценка для самых сложных функций из $Q_{n,2}$ выполняется.

9. Реализуем все булевы функции от n переменных схемой S в заданном базисе; в силу полноты базиса это можно сделать. Для схемы S зафиксируем какую-нибудь монотонную нумерацию вершин, при которой входы схемы получат номера $1, \ldots, n$ (это, как нетрудно заметить, можно сделать). Просмотрим все функциональные элементы в порядке возрастания их номеров и лишние элементы удалим из схемы. Пусть E — функциональный элемент с номером i. Пусть на выходе этого элемента реализуется некоторая функция φ, причём этот выход соединён со входами $v_1, \ldots, v_a$ других элементов схемы S. Если ни в одной вершине схемы S с номером меньшим, чем i, функция φ не реализуется, то элемент E оставляем в схеме и переходим к следующему элементу с номером $i + 1$. Предположим, что функция φ уже реализована в некоторой вершине v схемы с номером j, где $j < i$.

В таком случае элемент E оказывается лишним и удаляется из схемы; входы $v_1, \ldots, v_a$ (других элементов) после удаления E соединим с вершиной v. После удаления лишнего элемента снова получим схему, реализующую то же множество функций, что и до удаления. В силу конечности исходной схемы через конечное число шагов будет получена реализующая все булевы функции от n переменных схема S^*, во всех вершинах которой реализуются различные функции. Булевых функций от n переменных имеется всего 2^{2^n} штук, следовательно, и вершин в схеме S^* должно быть 2^{2^n} штук. Входами схемы будут ровно n вершин; остальные $2^{2^n} - n$ вершин будут функциональными элементами.

Поскольку в любой схеме S, реализующей все булевы функции от n переменных, ограничиться числом вершин, меньшим чем 2^{2^n}, очевидно, нельзя, а только n вершин могут отвечать входам схемы, то в S должно быть не менее чем $2^{2^n} - n$ функциональных элементов; отсюда следует, что полученная после удаления лишних элементов схема S^* является минимальной.

10. Пусть наборы $\widetilde{x} = (x_n, x_{n-1}, \ldots, x_1)$ и $\widetilde{y} = (y_n, y_{n-1}, \ldots, y_1)$ изображают два n-разрядных двоичных числа (младшие разряды, как обычно, справа). Схема, изображённая на рис. 44, как нетрудно заметить, содержит $5n - 3$ элементов и реализует сумму чисел $\widetilde{x}$ и $\widetilde{y}$; на этом рисунке $p_1, p_2, \ldots, p_{n-1}, p_n$ обозначают соответственно переносы во 2-й, в 3-й,..., в n-й, в $(n+1)$-й разряды суммы.

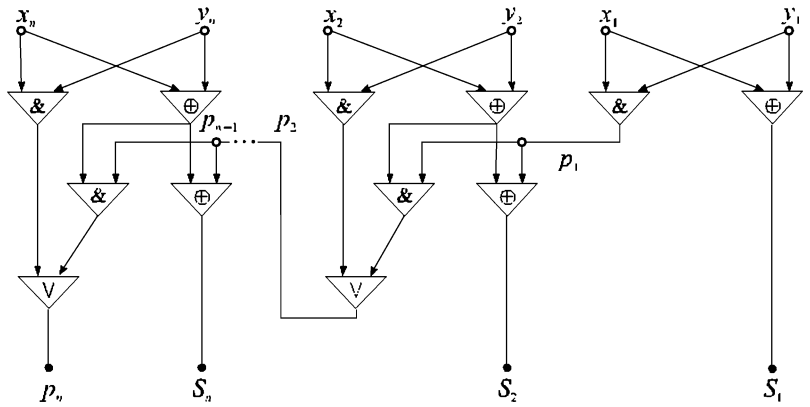


Рис. 44

11. Предположим, что $n = 2^m$, и укажем способ построения схемы R_n, вычисляющей число единиц в наборе $\widetilde{x} = (x_1, \ldots, x_n)$. Из результатов задач 10 и 1.б) следует, что сумматор S_l для

сложения двух l-разрядных двоичных чисел $\widetilde{x}$ и $\widetilde{y}$ можно построить из $11l-6$ элементов «&», «$\vee$», «$^-$»; подходящая схема получается из схемы, изображённой на рис. 44, заменой каждого элемента «$\oplus$» блоком из четырёх элементов «&», «$\vee$», «$^-$», реализующим «$\oplus$». Разобьём теперь разряды входного набора $\widetilde{x}$ на пары (x_1, x_2), (x_3, x_4),..., (x_{n-1}, x_n) — всего 2^{m-1} пар. Для сложения одноразрядных чисел в каждой паре возьмём 2^{m-1} сумматоров S_1 с общей сложностью $2^{m-1}\cdot 5$. На выходах сумматоров S_1 получим 2^{m-1} двухразрядных чисел. Разобьём теперь эти числа на 2^{m-2} пар и в каждой паре сложим двухразрядные числа с использованием 2^{m-2} сумматоров S_2 с общей сложностью $2^{m-2}\cdot(11\cdot 2-6)$. Полученные на выходах сумматоров S_2 трёхзначные числа разобьём на пары, и т. д. На i-м шаге сложим 2^{m-i+1} штук i-разрядных чисел с использованием 2^{m-i} сумматоров S_i с общей сложностью $2^{m-i}(11i-6)$. После m-го шага будет построена схема R_n с $m+1$ выходами, на которых будет реализован набор требуемых функций $f_1(\widetilde{x}), \ldots, f_{m+1}(\widetilde{x})$.

Для сложности схемы R_n получаем оценку

$$L(R_n)\leqslant \sum_{i=1}^{m} 2^{m-i}(11i-6) < 11\cdot 2^m \sum_{i=1}^{m}\frac{i}{2^i} < 11\cdot 2^m \sum_{i=1}^{\infty}\frac{i}{2^i} \leqslant c\cdot 2^m,$$

ибо ряд $\sum\limits_{i=1}^{\infty}\dfrac{i}{2^i}$ сходится (c — некоторая константа).

Рассмотрим теперь общий случай, когда n — произвольное: $2^{m-1} < n \leqslant 2^m$. Нетрудно заметить, что в этом случае можно воспользоваться построенной выше схемой R_{2^m}, подавая на первые n входов переменные $x_1, \ldots, x_n$, а на остальные $(2^m - n)$ входов константу 0, которую можно получить с использованием двух элементов. В итоге получаем, что сложность реализации функций $f_1, \ldots, f_{\lceil \log(n+1)\rceil}$ не превосходит $2cn+1$, т. е. $L_{\{\&,\vee,\ ^-\}}(C_n) \lesssim n$.

12. Из определения симметрической функции следует, что значение такой функции $f(x_1, \ldots, x_n)$ на произвольном наборе $\widetilde{\sigma} = (\sigma_1, \ldots, \sigma_n)$ значений переменных однозначно определяется числом единиц в наборе $\widetilde{\sigma}$. С учетом этого обстоятельства можно воспользоваться следующей схемой вычисления значений функции f. Вначале $(n, \lceil \log(n+1)\rceil)$-оператор C_n (см. предыдущую задачу) по набору $\widetilde{\sigma}$ длины n вычисляет набор $\widetilde{\beta} = (\beta_1, \ldots, \beta_m)$, $m = \lceil \log(n+1)\rceil$, являющийся двоичной записью числа единиц в $\widetilde{\sigma}$. Затем $(m, 1)$-оператор S_m по набору $\widetilde{\beta}$ выдаёт значение

функции f на наборе $\widetilde{\sigma}$; оператор S_m, представляющий собой некоторую булеву функцию φ от m переменных, при необходимости можно доопределить произвольным образом на наборах $\widetilde{\beta}$ таких, что $|\widetilde{\beta}| > n$. Из результата предыдущей задачи следует, что оператор C_n можно реализовать со сложностью, по порядку не превосходящей n, а из теоремы 17 с учётом значения параметра m получаем, что булеву функцию φ можно реализовать со сложностью, по порядку не превосходящей $\dfrac{n}{\log n}$. В итоге получаем, что сложность реализации симметрической булевой функции от n переменных по порядку не превосходит n.

13. Пусть $Б_1 = \{f_1, \ldots, f_k\}$, а $Б_2 = \{\varphi_1, \ldots, \varphi_l\}$. Функции $f_1, \ldots, f_k$ из $Б_1$ реализуем соответственно схемами $S_{1,1}, \ldots, S_{1,k}$ в базисе $Б_2$ и положим $c_1 = \max\{L(S_{1,1}), \ldots, L(S_{1,k})\}$; функции $\varphi_1, \ldots, \varphi_l$ из $Б_2$ реализуем соответственно схемами $S_{2,1}, \ldots, S_{2,l}$ в базисе $Б_1$ и положим $c_2 = \max\{L(S_{2,1}), \ldots, L(S_{2,k})\}$. Всякой схеме S в базисе $Б_1$, реализующей булеву функцию f, очевидно, можно сопоставить схему S' в базисе $Б_2$, также реализующую f и такую, что $L(S') \leqslant \dfrac{1}{c_1} L(S)$ — достаточно лишь элементы «f_1», ..., «f_k» из S заменить соответственно подсхемами $S_{1,1}, \ldots, S_{1,k}$; отсюда следует неравенство $L_{Б_2}(f) \lesssim L_{Б_1}(f)$. Аналогичным образом выводится неравенство $L_{Б_1}(f) \lesssim L_{Б_2}(f)$.

14. Множество $K_n \cup \mathcal{D}_n$ содержит, очевидно, 2^{n+1} попарно различных функций, отличных от $x_1, \ldots, x_n$. Поэтому функции из $K_n \cup \mathcal{D}_n$ могут быть реализованы (в любом базисе) только на выходах различных элементов; отсюда вытекают нижние оценки $L_{\{\&,-\}}(K_n \cup \mathcal{D}_n) \geqslant 2^{n+1}$ и $L_{\{\vee,-\}}(K_n \cup \mathcal{D}_n) \geqslant 2^{n+1}$.

Верхние оценки получаем конструктивно. Построим схему в базисе $\{\&, {}^-\}$, содержащую асимптотически не более чем 2^{n+1} элементов и реализующую систему всех элементарных конъюнкций и дизъюнкций ранга n. Разобьём множество $\{x_1, \ldots, x_n\}$ на два подмножества $\{x_1, \ldots, x_m\}$ и $\{x_{m+1}, \ldots, x_n\}$, где $m = \left\lceil \dfrac{n}{2} \right\rceil$. Как и при доказательстве теоремы 15, реализуем все элементарные конъюнкции $x_1^{\sigma_1} \cdot \ldots \cdot x_m^{\sigma_m}$ схемой S_1 с использованием m инверторов и $2^{m+1} - 4$ конъюнкторов, а все элементарные конъюнкции $x_{m+1}^{\sigma_{m+1}} \cdot \ldots \cdot x_n^{\sigma_n}$ схемой S_2 с использованием $n - m$ инверторов и $2^{n-m+1} - 4$ конъюнкторов. К схемам S_1, S_2 добавим ещё 2^n конъюнкторов, на выходах которых получим

требуемые конъюнкции; конъюнкция $x_1^{\sigma_1} \cdot \ldots \cdot x_m^{\sigma_m} \cdot x_{m+1}^{\sigma_{m+1}} \cdot \ldots \times$ $\times\, x_n^{\sigma_n}$ получается при этом на выходе конъюнктора, входы которого соединяются с выходами подсхем S_1 и S_2, отвечающими конъюнкциям $x_1^{\sigma_1} \cdot \ldots \cdot x_m^{\sigma_m}$ и $x_{m+1}^{\sigma_{m+1}} \cdot \ldots \cdot x_n^{\sigma_n}$. К полученной схеме добавим 2^n инверторов, на выходах которых получим требуемые дизъюнкции; дизъюнкция $x_1^{\sigma_1} \vee \ldots \vee x_m^{\sigma_m} \vee x_{m+1}^{\sigma_{m+1}} \vee \ldots \vee x_n^{\sigma_n}$ получается на выходе инвертора, вход которого соединяется с выходом конъюнктора, реализующего конъюнкцию $x_1^{\overline{\sigma}_1} \cdot \ldots \cdot x_m^{\overline{\sigma}_m} \times$ $\times\, x_{m+1}^{\overline{\sigma}_{m+1}} \cdot \ldots \cdot x_n^{\overline{\sigma}_n}$. В итоге получим схему, которая реализует систему функций $K_n \cup \mathcal{D}_n$ и сложность которой асимптотически не превосходит 2^{n+1}.

Асимптотическая верхняя оценка $L_{\{\vee,-\}}(K_n \cup \mathcal{D}_n) \lesssim 2^{n+1}$ получается аналогичным образом. При этом нужно только вначале с помощь соответствующих подсхем S_1 и S_2 реализовать всевозможные дизъюнкции $x_1^{\sigma_1} \vee \ldots \vee x_m^{\sigma_m}$ и $x_{m+1}^{\sigma_{m+1}} \vee \ldots \vee x_n^{\sigma_n}$, затем добавить 2^n дизъюнкторов и реализовать всевозможные дизъюнкции ранга n; наконец, добавить 2^n инверторов и реализовать всевозможные конъюнкции ранга n.

К главе VI

1. а) Первый и пятый столбцы матрицы M различаются только во второй строке, следовательно, любой тест T эту строку содержит. Подматрица M', получающаяся из M вычёркиванием второй строки, состоит из двух — в данном случае одинаковых — подматриц M_1 и M_0; M_1 содержит 1-й–4-й столбцы (в этих столбцах во второй строке исходной матрицы стоят единицы), а M_0 содержит 5-й–8-й столбцы (в этих столбцах во второй строке исходной матрицы стоят нули). Тест T для M, очевидно, должен содержать строки, составляющие тест для M_1 (и одновременно тест для M_0). Если T содержит 1-ю строку, то в этом случае T содержит также 3-ю или 4-ю строку (только в них различаются 1-й и 2-й столбцы) и 5-ю или 6-ю строку (только в этих строках различаются 3-й и 4-й столбцы). Всего в T окажется 4 строки. Если первая строка из M в T отсутствует, то в этом случае в минимальный (т. е. содержащий наименьшее число строк) тест должны войти, как нетрудно заметить, какие-то три строки из последних четырёх и всего в T окажется опять же 4 строки.

б) Пусть A_i, $i = 1, \ldots, 12$, — i-я строка, а B_j, $j = 1, \ldots, 11$, — j-й столбец матрицы M. Пусть T — тест для M. В зависимости от вхождения в T первой и последней строк рассмотрим четыре

случая: 1) $A_1, A_{12} \in T$; 2) $A_1 \in T$, $A_{12} \notin T$; 3) $A_1 \notin T$, $A_{12} \in T$; 4) $A_1, A_{12} \notin T$.

1) Для того чтобы различать столбцы B_2–B_6, в T должны присутствовать четыре из пяти строк A_2–A_6, а чтобы различать столбцы B_7–B_{11}, в T должны присутствовать четыре из пяти строк A_7–A_{11}. Всего в T окажется 10 строк, и их будет достаточно, чтобы составить тест.

2) В T должны быть строки A_2–A_6 (чтобы различать столбцы B_1–B_6) и четыре из пяти строк A_7–A_{12} (чтобы различать столбцы B_7–B_{11}). Всего в T окажется 10 строк.

3) В T должны присутствовать четыре из пяти строк A_2–A_6 и пять строк A_7–A_{11} (чтобы различать столбцы B_2–B_{11} между собой и отличать их от B_1). В T войдут 10 строк.

4) В T должны войти строки A_2–A_{11} (чтобы различать столбцы B_2–B_{11} между собой и отличать их от B_1).

В итоге получаем, что длина минимального теста таблицы M равна 10.

2. а) n.

б) Любые $n-1$ строк составят минимальный тест. Действительно, любые два столбца из M различаются между собой по крайней мере в двух строках. С другой стороны, для любых двух строк найдутся два столбца с одним и тем же числом единиц, которые различаются между собой только в данных строках. Значит, минимальный тест матрицы M имеет длину $n-1$.

3.

$$\{(000), (001), (011), (101), (110)\},$$
$$\{(000), (001), (011), (101), (111)\},$$
$$\{(001), (010), (011), (101), (110)\},$$
$$\{(001), (010), (011), (101), (111)\},$$
$$\{(001), (011), (100), (101), (110)\},$$
$$\{(001), (011), (100), (101), (111)\}.$$

4. а) Представим линейную функцию в виде с. д. н. ф.:

$$x_1 \oplus \ldots \oplus x_n = \bigvee_{\substack{(\sigma_1,\ldots,\sigma_n) \\ \sigma_1 \oplus \ldots \oplus \sigma_n = 1}} x_1^{\sigma_1} \ldots x_n^{\sigma_n}. \qquad (1)$$

В соответствии с этим представлением построим схему S, которая содержит блок A (рис. 45,*а*) из $(n-1)$ инверторов, реализующий инверсии переменных $x_2, \ldots, x_n$, блок B из 2^{n-1} цепочек $Z_{\tilde{\sigma}}$, реализующих конъюнкции из правой части представления (1), и блок C, представляющий собой цепочку из $2^{n-1}-1$

дизъюнкторов для выполнения внешней операции (дизъюнкции) из (1). Цепочка $Z_{\widetilde{\sigma}}$ реализует конъюнкцию $K_{\widetilde{\sigma}}(\widetilde{x}) = x_1^{\sigma_1} x_2^{\sigma_2} \dots x_n^{\sigma_n}$ и устроена, как показано на рис. 45,*б* в случае $\sigma_1 = 1$ или на рис. 45,*в* в случае $\sigma_1 = 0$. Схема S в исправном состоянии, очевидно, реализует $x_1 \oplus \dots \oplus x_n$.

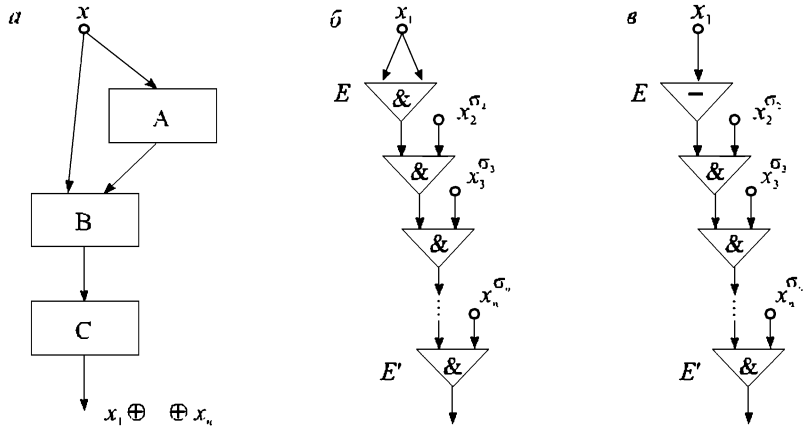


Рис. 45

Пусть $\widetilde{\sigma}$ — произвольный набор, удовлетворяющий условию $\sigma_1 \oplus \dots \dots \oplus \sigma_n = 1$, а $Z_{\widetilde{\sigma}}(\widetilde{x})$ — цепочка из S, реализующая $K_{\widetilde{\sigma}}(\widetilde{x})$. Предположим, что в цепочке $Z_{\widetilde{\sigma}}(\widetilde{x})$ нижний элемент E' неисправен и выдаёт константу 0. В этом случае схема S реализует функцию неисправности $g_{\widetilde{\sigma}}(\widetilde{x})$, отличающуюся от $x_1 \oplus \dots \oplus x_n$ только на наборе $\widetilde{\sigma}$ ($g_{\widetilde{\sigma}}(\widetilde{\sigma}) = 0$, а $\sigma_1 \oplus \dots \oplus \sigma_n = 1$); значит, набор $\widetilde{\sigma}$ должен войти в проверяющий тест для S.

Пусть теперь $\widetilde{\sigma}$ — нулевой набор для функции $x_1 \oplus \dots \oplus x_n$, т. е. $\sigma_1 \oplus \dots \oplus \sigma_n = 0$. Предположим, что в цепочке $Z_{\widetilde{\sigma}'}$, где $\widetilde{\sigma}' = (\overline{\sigma}_1, \sigma_2, \sigma_3, \dots, \sigma_n)$, верхний элемент E неисправен и выдаёт константу 1. В этом случае цепочка $Z_{\widetilde{\sigma}'}$ реализует конъюнкцию $(n-1)$-го ранга $x_2^{\sigma_2} \dots x_n^{\sigma_n}$ и на выходе схемы S получим функцию неисправности $g'_{\widetilde{\sigma}}(\widetilde{x})$, отличающуюся от $x_1 \oplus \dots \oplus x_n$ только на наборе $\widetilde{\sigma}$ ($g'_{\widetilde{\sigma}}(\widetilde{\sigma}) = 1$, а $\sigma_1 \oplus \dots \oplus \sigma_n = 0$). Значит, и в этом случае набор $\widetilde{\sigma}$ придётся включить в тест для S. В итоге в проверяющем тесте окажутся все 2^n наборов.

б) Реализуем линейную функцию $f_n^{\oplus} = x_1 \oplus \dots \oplus x_n$ схемой S, изображённой на рис. 46. Возьмем множество T из четырёх наборов $\widetilde{\sigma}_1 = (0, 0, 0, \dots, 0)$, $\widetilde{\sigma}_2 = (1, 0, 0, \dots, 0)$, $\widetilde{\sigma}_3 = (0, 1, 1, \dots, 1)$, $\widetilde{\sigma}_4 = (1, 1, 1, \dots, 1)$. Если мы подадим на входы нашей схемы набор $\widetilde{\sigma}_1$, а затем набор $\widetilde{\sigma}_2$, то при переходе от

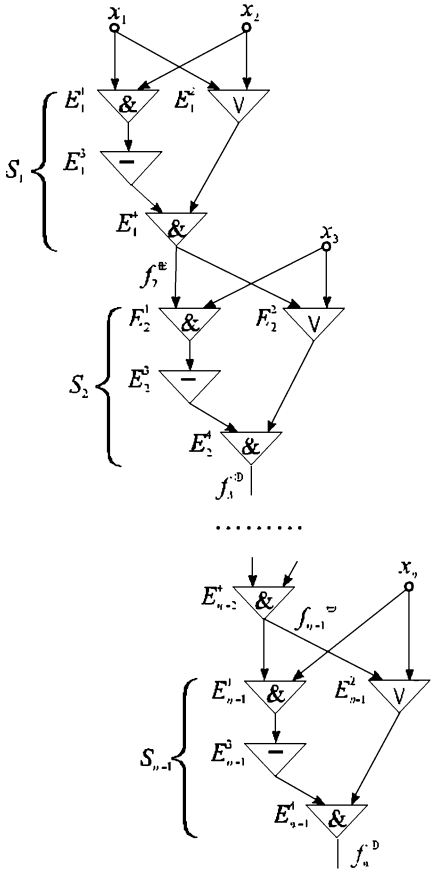


Рис. 46

набора $\widetilde{\sigma}_1$ к набору $\widetilde{\sigma}_2$ значение на выходе исправной схемы S изменится; изменяется значение на выходе исправной схемы S и при переходе от набора $\widetilde{\sigma}_3$ к набору $\widetilde{\sigma}_4$.

Предположим, что в схеме S неисправен хотя бы один дизъюнктор, скажем, дизъюнктор E_i^2, $i \in \{1, \ldots, n-1\}$. В этом случае независимо от состояния элемента E_i^1 (этот элемент может находиться как в исправном, так и в неисправном состоянии) значения на выходах элементов E_i^1, E_i^2 при переходе от $\widetilde{\sigma}_1$ к $\widetilde{\sigma}_2$ не изменяются; это следует из неисправности элемента E_i^2 и равенства нулю переменной x_{i+1}, подаваемой на один из входов элемента E_i^1. Не изменяются при указанном переходе и значения переменных $x_{i+2}, \ldots, x_n$, подаваемых на входы элементов

$E^1_{i+1}, E^2_{i+1}, \dots, E^1_{n-1}, E^2_{n-1}$ (т. е. на входы подсхем $S_{i+1}, \dots, S_{n-1}$). В итоге и значение на выходе всей схемы при переходе от $\widetilde{\sigma}_1$ к $\widetilde{\sigma}_2$ не изменится, т. е. рассматриваемая неисправность обнаруживается на входных наборах $\widetilde{\sigma}_1$, $\widetilde{\sigma}_2$.

А теперь предположим, что в какой-нибудь подсхеме S_i неисправен хотя бы один элемент, отличный от дизъюнктора E^2_i, $i \in \{1, \dots, n-1\}$, этой подсхемы. В этом случае при подаче на входы схемы набора $\widetilde{\sigma}_3$, а затем набора $\widetilde{\sigma}_4$ значение на выходе дизъюнктора E^2_i не изменяется (при любом состоянии этого дизъюнктора), поскольку на один вход дизъюнктора E^2_i подаётся в обоих случаях единица. А так как один из элементов E^1_i, E^3_i, E^4_i неисправен, то отсюда следует, что при переходе от $\widetilde{\sigma}_3$ к $\widetilde{\sigma}_4$ не изменится и значение на выходе E^4_i, т. е. на выходе подсхемы S_i. Значения переменных $x_{i+2}, \dots, x_n$, подаваемых на входы подсхем $S_{i+1}, \dots, S_{n-1}$, на наборах $\widetilde{\sigma}_3$, $\widetilde{\sigma}_4$ не изменяются. В итоге получаем, что при переходе от $\widetilde{\sigma}_3$ к $\widetilde{\sigma}_4$ значение на выходе всей схемы S не изменится, т. е. рассматриваемая в данном случае неисправность будет обнаружена.

Таким образом, получаем, что множество T из четырёх наборов является полным проверяющим тестом для приведенной схемы S.

5. Множество $Q(f)$ содержит, очевидно, не более, чем $2n$ попарно различных функций $\varphi_1, \dots, \varphi_r$, $r \leqslant 2n$. Зададим функции $f, \varphi_1, \dots, \varphi_r$ таблицей M из $(r+1)$ столбцов значений этих функций. По теореме 20 существует тест T этой таблицы, содержащий не более чем r строк. Но тест T таблицы M определяет и единичный диагностический тест функции f, который состоит из множества наборов значений переменных $x_1, \dots, x_n$, отвечающих строкам из T; входит в этот тест не более чем $2n$ наборов.

6. Можно взять, например, функцию

$$f(x_1, x_2, y_0, y_1, y_2, y_3) = \overline{x}_1\overline{x}_2y_0 \vee \overline{x}_1x_2y_1 \vee x_1\overline{x}_2y_2 \vee x_1x_2y_3.$$

7. Положим $m = \lceil \log n \rceil$, $l = n - m$ и возьмём булеву функцию $f(x_1, \dots, x_m, y_0, \dots, y_{l-1}) = \bigvee\limits_{i=0}^{l-1} x_1^{\sigma_{i,1}} \dots x_m^{\sigma_{i,m}} y_m$, где набор $(\sigma_{i,1}, \dots, \sigma_{i,m})$ является двоичной записью числа i. Рассмотрим единичные константные неисправности на входах, соответствующих переменным $y_0, \dots, y_{l-1}$, и получающиеся при этом функции неисправности. При «забивании» переменной y_i ($i \in \{0, 1, \dots, l-1\}$) константой 0 (т. е. при подстановке 0 вместо y_i) получим функцию неисправности $g_{0,i}$; эта функция от-

личается от функции f лишь на наборах из множества $M_{1,i}$, содержащего все те входные наборы, у каждого из которых первые m разрядов (значения переменных $x_1, \ldots, x_m$) представляют двоичную запись i, $(m+i+1)$-й разряд (значение переменной y_i) равен 1, а все остальные разряды (значения переменных $y_0, \ldots, y_{i-1}, y_{i+1}, \ldots, y_{l-1}$) произвольны. Понятно, что искомый тест должен содержать хотя бы один набор из $M_{1,i}$ — в противном случае на наборах из теста нельзя будет отличить функцию $g_{0,i}$ от f.

При забивании переменной y_i константой 1 получаем функцию неисправности $g_{1,i}$; эта функция отличается от функции f лишь на наборах из множества $M_{0,i}$, содержащего все те входные наборы, у каждого из которых первые m разрядов представляют двоичную запись числа i, а $(m+i+1)$-й разряд равен 0, $i \in \{0, 1, \ldots, l-1\}$. Ясно, что в тест необходимо включить хотя бы по одному набору от каждого из множеств $M_{0,0}, \ldots, M_{0,l-1}$.

Если теперь учесть, что все множества $M_{0,0}$, ..., $M_{0,l-1}$, $M_{1,0}, \ldots, M_{1,l-1}$ попарно не пересекаются, то получаем, что любой единичный проверяющий тест для выбранной нами функции f должен содержать не менее чем $2l$ входных наборов.

8. Возьмём булеву функцию $f(\widetilde{x}) = \overline{x}_1\overline{x}_2\overline{x}_3\overline{x}_4\overline{x}_5 \vee x_1x_2x_3x_4x_5$. Обозначим через $g_{\sigma,i}(\widetilde{x})$, где $\sigma \in \{0,1\}$, $i \in \{1, \ldots, 5\}$, функцию, получающуюся из исходной булевой функции $f(\widetilde{x})$ подстановкой константы σ вместо переменной x_i. Исходную функцию и все функции неисправности зададим табл. 22, в которой выделим 12 входных наборов: нулевой набор; подмножество A из пяти наборов, каждый из которых содержит ровно одну единицу; подмножество B из пяти наборов, каждый из которых содержит ровно один нуль; единичный набор.

Пусть теперь T — произвольный диагностический тест для функции f; покажем, что T содержит не меньше чем $2n$ наборов. В зависимости от принадлежности нулевого и единичного наборов значений переменных тесту рассмотрим четыре случая: 1) $\widetilde{0}, \widetilde{1} \notin T$; 2) $\widetilde{0} \in T, \widetilde{1} \notin T$; 3) $\widetilde{0} \notin T$, $\widetilde{1} \in T$; 4) $\widetilde{0}, \widetilde{1} \in T$.

1) $\widetilde{0}, \widetilde{1} \notin T$. Все наборы из подмножеств A и B должны входить в T, ибо в противном случае при отсутствии в T i-го набора (с единицей в i-м разряде) из A функции $f(\widetilde{x})$ и $g_{o,i}(\widetilde{x})$ имели бы одинаковые значения на всех наборах из T, а при отсутствии в T i-го набора (с нулём в i-м разряде) из B были бы неразличимы на всех наборах из T функции $f(\widetilde{x})$ и $g_{1,i}(\widetilde{x})$.

2) $\widetilde{0} \in T, \widetilde{1} \notin T$. Все наборы из A должны входить в T — иначе на наборах из теста функция $f(\widetilde{x})$ была бы не отличима

Таблица 22

	x_1	x_2	x_3	x_4	x_5	f	$g_{0,1}$	$g_{0,2}$	$g_{0,3}$	$g_{0,4}$	$g_{0,5}$	$g_{1,1}$	$g_{1,2}$	$g_{1,3}$	$g_{1,4}$	$g_{1,5}$
	0	0	0	0	0	1	1	1	1	1	1	0	0	0	0	0
	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0
A	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0
	0	0	0	1	0	0	0	0	0	1	0	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	1	0	0	0	0	0	0	1	0	0	0	0
	1	0	1	1	1	0	0	0	0	0	0	0	1	0	0	0
B	1	1	0	1	1	0	0	0	0	0	0	0	0	1	0	0
	1	1	1	0	1	0	0	0	0	0	0	0	0	0	1	0
	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	1
	.....					...	...	...	...	...	...	...	...	...	...	...
	1	1	1	1	1	1	0	0	0	0	0	1	1	1	1	1

по крайней мере от одной из функций $g_{0,1}(\widetilde{x}), \ldots, g_{0,5}(\widetilde{x})$. Из подмножества B должны войти в T не меньше четырёх наборов — иначе на наборах из теста были бы неразличимы какие-нибудь две функции $g_{1,i}(\widetilde{x})$ и $g_{1,j}(\widetilde{x})$ (при отсутствии в тесте i-го и j-го наборов из B, $i \neq j$).

3) $\widetilde{0} \notin T, \widetilde{1} \in T$. Этот случай рассматривается аналогично предыдущему.

4) $\widetilde{0}, \widetilde{1} \in T$. От каждого из подмножеств A, B в тест должно войти не менее чем по четыре набора — иначе по крайней мере какие-нибудь две функции неисправности оказались бы неразличимыми на наборах из теста; при отсутствии i-го и j-го $(i \neq j)$ наборов из A на наборах из теста оказались бы неразличимыми функции $g_{0,i}(\widetilde{x})$ и $g_{0,j}(\widetilde{x})$, а при отсутствии i-го и j-го наборов из B были бы неразличимы функции $g_{1,i}(\widetilde{x})$ и $g_{1,j}(\widetilde{x})$.

Таким образом, во всех случаях тест T содержит не менее десяти наборов.

К главе VII

1. а), б), д) Функция является недетерминированной.

в), г) Функция является ограниченно-детерминированной. В случае г) при нечётном t, большем двух, как нетрудно заметить, выполняется условие $y(t) = \overline{x}(t) \vee x(t) = 1$.

е) Заданная функция, очевидно, является детерминированной. Рассмотрим занумерованное дерево $\mathcal{D}$ с корнем ξ_0, зада-

ющее $f(x)$. Пусть r — произвольное натуральное число. В $\mathcal{D}$ выделим произвольным образом $(r+1)$ вершин $\xi_1, \ldots, \xi_{r+1}$ из $2r$-го яруса, удовлетворяющих следующему условию: в наборе $(\alpha^{(i)}(1), \ldots, \alpha^{(i)}(2r))$ номеров дуг пути, ведущего из ξ_0 в ξ_i, нулей на i больше, чем единиц, $i \in \{1, \ldots, r+1\}$; такие вершины, очевидно, найдутся. Пусть $\mathcal{D}_1, \ldots, \mathcal{D}_{r+1}$ — бесконечные занумерованные поддеревья исходного дерева $\mathcal{D}$ с корнями в вершинах $\xi_1, \ldots, \xi_{r+1}$. Эти поддеревья, как нетрудно заметить, задают детерминированные функции $f_1, \ldots, f_{r+1}$ со следующими свойствами: для любой выходной последовательности $\{\gamma(1), \gamma(2), \ldots\}$ функции f_i начальный кусок этой последовательности длины i содержит одни нули, причём существует выходная последовательность, в которой $(i+1)$-й разряд равен 1. Из этих свойств деревьев $\mathcal{D}_1, \ldots, \mathcal{D}_{r+1}$ следует, что все они попарно неэквивалентны, а поскольку r выбиралось произвольным образом, то в итоге получаем, что исходная функция $f(x)$ не является ограниченно-детерминированной.

2. а) О.-д. функция веса 2.

б) О.-д. функция веса 3.

в) О.-д функция веса 2.

г) Рассмотрим занумерованное дерево $\mathcal{D}_0$ с корнем ξ_0, отвечающее детерминированной функции $f(x)$. В этом дереве все вершины из одного и того же яруса, как нетрудно заметить, эквивалентны — они будут корнями поддеревьев, задающих равные о.-д. функции. По построению в дереве $\mathcal{D}_0$ каждой дуге приписаны два числа (или два номера, или две метки). Первое из этих чисел — «входная» метка — определяет номер выбранной дуги в пучке дуг, исходящих из одной и той же вершины, а второе число (0 или 1) — «выходная» метка — определяет соответствующий разряд в выходной последовательности. Для $i \in \{0, 1, \ldots, n-1\}$ в дереве $\mathcal{D}_i$ выходные метки дуг в первых $n-i$ ярусах — одни лишь нули, а в последующих ярусах — и нули, и единицы; для $i \geqslant n$ во всех ярусах дерева $\mathcal{D}_i$ присутствуют дуги как с нулевыми, так и с единичными выходными метками, причём все вершины из n-го и последующих ярусов $\mathcal{D}_0$ эквивалентны. В итоге получаем, что все вершины дерева $\mathcal{D}_0$ распадаются на $n+1$ классов эквивалентности и вес о.-д. функции $f(x)$ равен $n+1$.

д) Очевидно, функция $f(x)$ — детерминированная. Пусть $\mathcal{D}_0$ — занумерованное дерево, задающее $f(x)$. В дереве $\mathcal{D}_0$ рассмотрим последовательность вершин $\xi_0, \xi_1, \ldots, \xi_r, \ldots$, лежащих на левой ветви: в вершину ξ_r ведёт путь из корня ξ_0,

включающий r дуг с нулевыми номерами (входными метками). Пусть $\mathcal{D}_0, \mathcal{D}_1, \ldots, \mathcal{D}_r$ — бесконечные поддеревья с корнями $\xi_0, \xi_1, \ldots, \xi_r$, определяющие детерминированные функции $f_0(x) = f(x)$, $f_1(x), \ldots, f_r(x)$. Рассмотрим последовательность $\gamma_i = f_i(1, 1, 1 \ldots)$, $i \in \{1, \ldots, r\}$, представляющую собой значение функции $f(x)$ на единичной входной последовательности. В последовательности γ_i в первых i разрядах с чётными номерами $2, 4, \ldots, 2i$ стоят нули, а в последующих разрядах с чётными номерами — единицы. Это означает, что все функции $f_0, f_1, \ldots, f_r$ попарно различны и исходная функция $f(x)$ не является ограниченно-детерминированной.

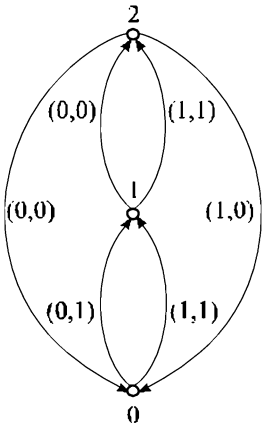


Рис. 47

е) О.-д. функция веса 4.

ж) О.-д. функция веса 12.

3. а) Диаграмма переходов и таблица переходов для заданной о.-д. функции представлены соответственно на рис. 47 и в табл. 23. Функции $y(t)$, $q_1(t)$, $q_2(t)$ в 3-й и в 8-й строках таблицы переходов доопределены так, чтобы $q_1(t)$ совпадала с $q_2(t-1)$ (при этом для реализации $q_1(t)$ вообще не потребовались функциональные элементы), а совершенные д. н. ф. для $y(t)$ и $q_2(t)$ содержали как можно меньше слагаемых.

Таблица 23

$x(t)$	$q_1(t-1)$	$q_2(t-1)$	$y(t)$	$q_1(t)$	$q_2(t)$
0	0	0	1	0	1
0	0	1	0	1	0
0	1	0	0	0	0
0	1	1	0	1	0
1	0	0	1	0	1
1	0	1	1	1	0
1	1	0	0	0	0
1	1	1	0	1	0

Функцию $y(t)$ представим в виде с. д. н. ф. и затем упростим следующим образом:

$$y(t) = \overline{x}(t)\overline{q}_1(t-1))\overline{q}_2(t-1) \vee x(t)\overline{q}_1(t-1)\overline{q}_2(t-1) \vee$$
$$\vee x(t)\overline{q}_1(t-1)q_2(t-1) =$$
$$= \overline{q}_1(t-1)\overline{q}_2(t-1) \vee x(t)\overline{q}_1(t-1) = (x(t) \vee \overline{q}_2(t-1))\overline{q}_1(t-1).$$

Канонические уравнения представим в следующем виде:

$$y(t) = (x(t) \vee \overline{q}_2(t-1))\overline{q}_1(t-1),$$
$$q_1(t) = q_2(t-1),$$
$$q_2(t) = \overline{q}_1(t-1)\overline{q}_2(t-1),$$
$$q_1(0) = q_2(0) = 0.$$

Построенная в соответствии с этими каноническими уравнениями схема автомата представлена на рис. 48.

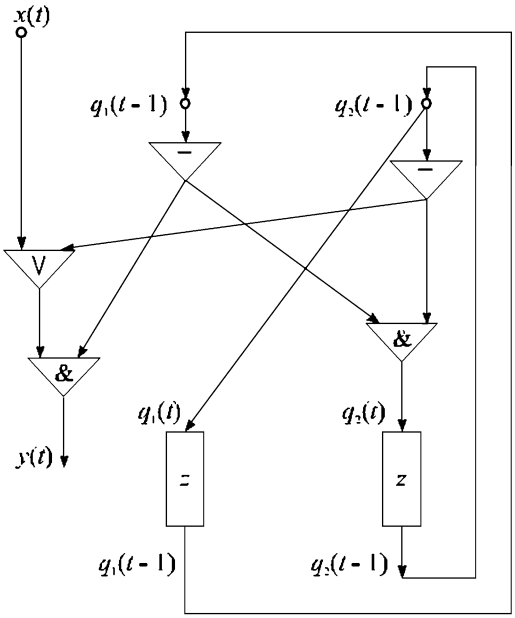


Рис. 48

б) В данном случае использование бесконечного занумерованного дерева для представления заданной о.-д. функции приводит к весьма громоздким вычислениям, поскольку для того, чтобы получить достаточно полное представление о получающемся дереве, придется построить, скорее всего, не менее восьми ярусов этого дерева. Гораздо проще воспользоваться представлением об автомате как о дискретном преобразователе, который в каждый момент времени $t = 1, 2 \ldots$, получая сигнал (0 или 1) на входе, должен в тот же момент выдать нужный сигнал (0 или 1) на выходе. В данной задаче о.-д. функция задана определёнными условиями (соотношениями), в соответствии с которыми по входной последовательности однозначно находится выходная

последовательность. Значит, и автомат должен «помнить» эти условия и правильным образом учитывать их при своей работе. «Память» автомата связывается с его состояниями, а каждое состояние автомата отвечает определённой вершине диаграммы переходов.

В нашем конкретном случае исходные условия, задающие о.-д. функцию, очевидно, можно несколько видоизменить и представить следующим образом:

$$y(t) = \begin{cases} 0, & \text{если} \quad t(\mathrm{mod}\,4) = 1 \quad \text{или} \quad t(\mathrm{mod}\,4) = 3, \\ x(t), & \text{если} \quad t(\mathrm{mod}\,4) = 2, \\ x(t-2), & \text{если} \quad t(\mathrm{mod}\,4) = 0. \end{cases}$$

С учётом такого представления вершины диаграммы переходов разместим в четырёх ярусах с номерами $0, 1, 2, 3$. В любую вершину из i-го яруса или, точнее, в состояние, отвечающее этой вершине, автомат переходит в моменты времени t, удовлетворяющие условию $t(\mathrm{mod}\,4) = i$, и самим этим фактом перехода в вершину из i-го яруса «запоминает», когда это произошло. Раздвоение диаграммы на две части (A и B) при $t(\mathrm{mod}\,4) = 2$ позволяет запомнить сигнал, поданный на вход автомата в момент t при $t(\mathrm{mod}\,4) = 2$; в вершины из A автомат попадает при нулевом сигнале на входе, а в вершины из B — при единичном сигнале на входе. Эти рассуждения позволяют построить диаграмму, приведённую на рис. 49; с учётом распределения вершин по ярусам и по частям A и B легко устанавливаем и вторую метку (выходной сигнал) каждой дуги диаграммы.

Обратим внимание на то обстоятельство, что число вершин в построенной указанным способом диаграмме переходов может оказаться, вообще говоря, больше числа классов эквивалентности для вершин в соответствующем занумерованном бесконечном дереве, т. е. больше веса заданной о.-д. функции. Но тем не менее построенные таким способом диаграмма переходов, а затем с использованием этой диаграммы таблица переходов, канонические уравнения и схема автомата отвечают заданной о.-д. функции.

Таблица переходов, составленная в соответствии с приведённой на рис. 49 диаграммой переходов, представлена табл. 24, в которой функция $q_1(t)$ доопределена в 7-й, 8-й, 15-й и 16-й строках единицами — чтобы её можно было выразить через $q_2(t-1)$, а функции $y(t)$, $q_2(t)$, $q_3(t)$ доопределены нулями — чтобы совершенные д. н. ф., представляющие эти функции, содержали меньше слагаемых. В этой таблице (и далее в канонических

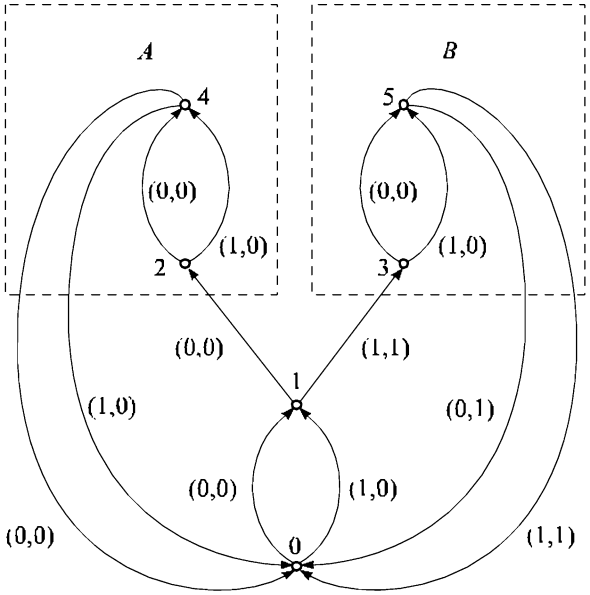


Рис. 49

Таблица 24

x	q_1	q_2	q_3	y	$q_1(t)$	$q_2(t)$	$q_3(t)$
0	0	0	0	0	0	0	1
0	0	0	1	0	0	1	0
0	0	1	0	0	1	0	0
0	0	1	1	0	1	0	1
0	1	0	0	0	0	0	0
0	1	0	1	1	0	0	0
0	1	1	0	0	1	0	0
0	1	1	1	0	1	0	0
1	0	0	0	0	0	0	1
1	0	0	1	1	0	1	1
1	0	1	0	0	1	0	0
1	0	1	1	0	1	0	1
1	1	0	0	0	0	0	0
1	1	0	1	1	0	0	0
1	1	1	0	0	1	0	0
1	1	1	1	0	1	0	0

уравнениях и на схеме) для краткости вместо $x(t)$, $y(t)$, $q_1(t-1)$, $q_2(t-1)$, $q_3(t-1)$ мы пишем соответственно x, y, q_1, q_2, q_3.

При составлении канонических уравнений $q_1(t)$ выражаем через $q_2(t-1)$, а задаваемые табл. 24 функции $y(t)$, $q_2(t)$, $q_3(t)$ представляем сначала в виде совершенных д. н. ф. После упрощения д. н. ф. с использованием подходящих эквивалентностей получим следующие канонические уравнения:

$$\begin{cases} y(t) = (x \vee q_1)\overline{q}_2 q_3, \\ q_1(t) = q_2, \\ q_2(t) = \overline{q}_1 \overline{q}_2 q_3, \\ q_3(t) = x\overline{q}_1\overline{q}_2 \vee \overline{q}_1(\overline{q}_2\overline{q}_3 \vee q_2 q_3), \\ q_1(0) = q_2(0) = q_3(0) = 0. \end{cases}$$

Схема автомата, реализующего заданную о.-д. функцию $y(t)$, изображена на рис. 50. Чтобы не загромождать рисунок некоторые соединения указаны условно; при этом обозначения $x, q_i, \overline{q}_i$ у входов функциональных элементов означают, что эти вхо-

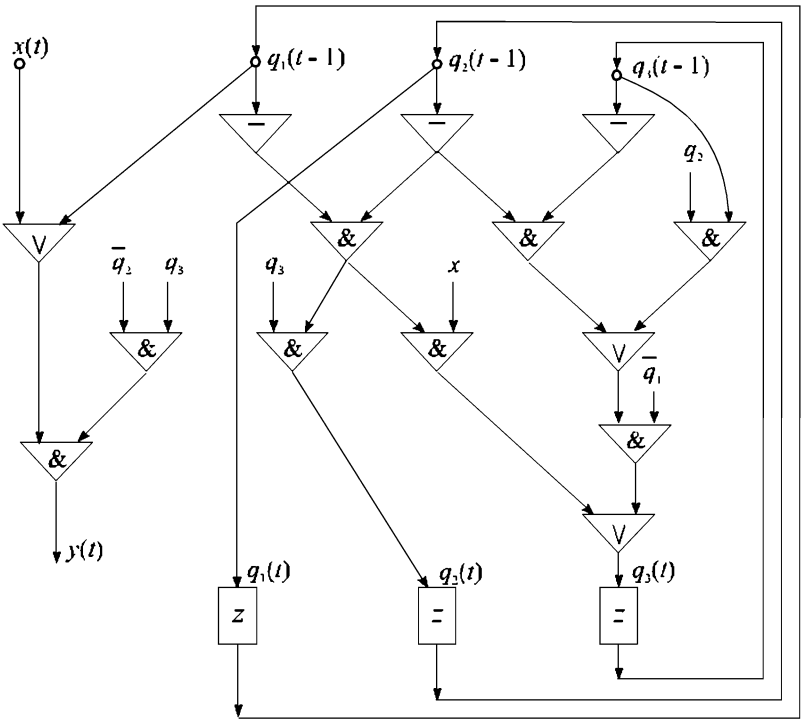


Рис. 50

ды соединены соответственно со входами схемы, отвечающими $x(t), q_i(t-1)$, и с выходами инверторов, на которых реализуются $\overline{q}_i(t-1)$, $i = 1, 2, 3$.

в) Можно воспользоваться диаграммой переходов, представленной на рис. 51. На этой диаграмме в первый момент времени осуществляется переход из состояния 0 в состояние 1; независимо от сигнала, подаваемого на вход автомата в этот момент времени, автомат выдаёт (на выходе) 1. В момент времени 2 автомат переходит из состояния 1 в состояние 2 при наличии нуля на входе или в состояние 3 при наличии единицы на входе; при обоих переходах автомат выдаёт 0. С этого момента времени и во все последующие моменты автомат переходит в состояние 2 при подаче на вход нуля; таким образом, состояние 2 связывается с «запоминанием» автоматом факта подачи на его вход нуля. Соответственно, оказавшись в состоянии 3, автомат «помнит», что на вход была подана единица. Нетрудно заметить, что функционирующий в соответствии с приведённой диаграммой автомат правильным образом «перерабатывает» входную последовательность сигналов в выходную, т. е. реализует именно заданную исходными условиями о.-д. функцию.

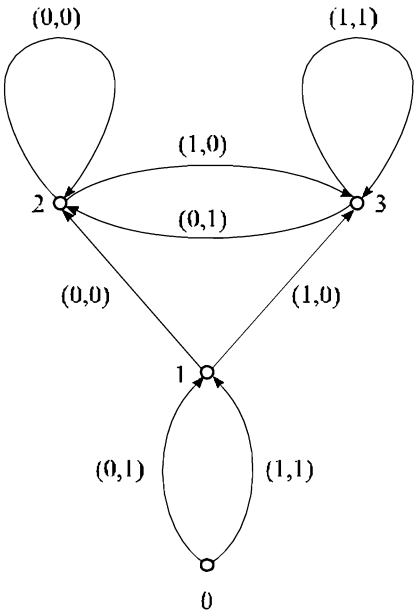


Рис. 51

г) Можно взять диаграмму переходов, изображённую на рис. 52; на этой диаграмме параллельные дуги с одинаковыми вторыми метками склеены (например, вместо двух дуг с метками $(0,0)$ и $(1,0)$, ведущих из вершины 3 в вершину 4, изображена одна дуга с метками $(0,1;0)$). В состояния 3, 5, 7, 9 автомат переходит в чётные моменты времени, а в состояния 4, 6, 8, 10 — в нечётные моменты. Пара вершин 3, 4 отвечает подаче на вход автомата в моменты 1, 2 значений 0, 0; пара 5, 6 отвечает первым двум сигналам 0, 1 и т. д. В соответствии с приведённой диаграммой «память» автомата трактуется достаточно просто и наглядно. Оказавшись, скажем, в состоянии 8, автомат «помнит», что переход в это состояние произошёл в нечётный момент времени и что в моменты 1 и 2 на вход поступили соответственно сигналы 1 и 0. Опять-таки в данном случае мы не исключаем возможности построения диаграммы с меньшим числом вершин, но, как нетрудно заметить, автомат, построенный по данной диаграмме, реализует именно заданную о.-д. функцию (построение таблицы переходов, канонических уравнений и схемы автомата не приводится, поскольку это делается уже весьма просто — почти очевидным образом).

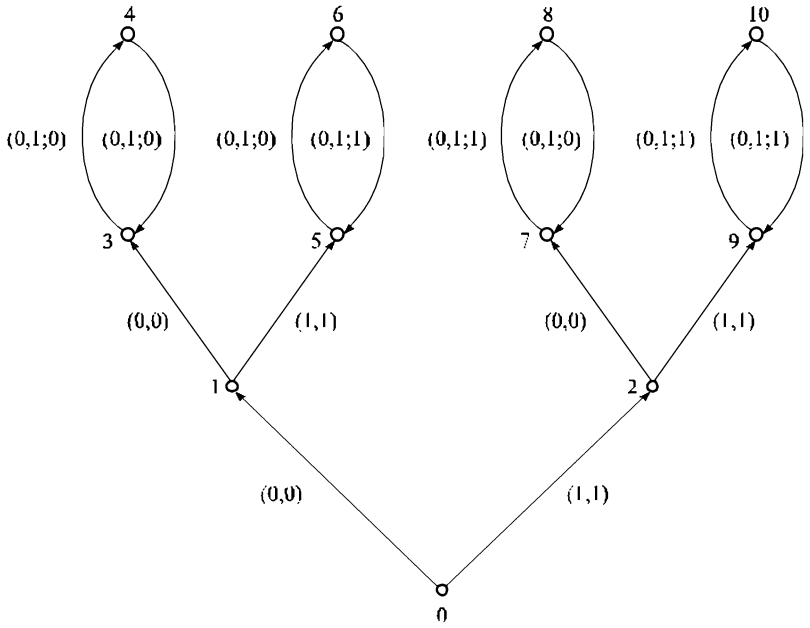


Рис. 52

д) Можно воспользоваться диаграммой переходов, изображённой на рис. 53; на этой диаграмме номер состояния, в которое переходит автомат в момент t, определяется как $\sum_{i=1}^{t} x(i) (\mathrm{mod}\, 3)$.

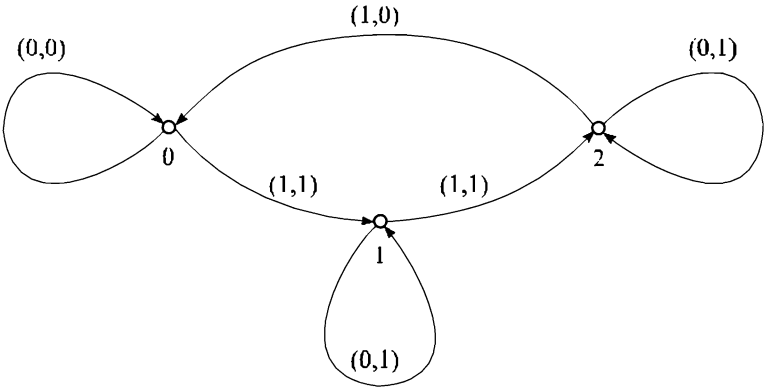


Рис. 53

е) Из соотношений, задающих $y(t)$, нетрудно заметить, что если единичный сигнал впервые поступит на вход автомата в момент t_0, то в моменты $1, 2, \ldots, t_0$ автомат выдаёт 0, а во все последующие моменты он выдаёт 1; именно такой автомат задаёт диаграмма переходов, представленная на рис. 54.

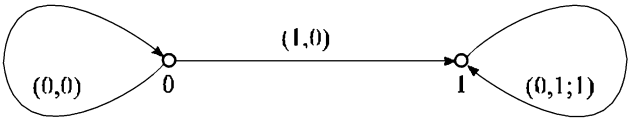


Рис. 54

ж) При реализации заданной о.-д. функции автомат в первый момент времени, очевидно, должен выдать единицу (независимо от сигнала на входе в этот момент времени), а в каждый последующий момент должен выдавать сигнал на выходе с учётом того, какие сигналы во входной и в выходной последовательностях имели место в предшествующий момент времени. Организуем «память» автомата следующим образом: если сигналы на входе и на выходе автомата в момент t составляют одну из пар меток $(0,0), (0,1), (1,0), (1,1)$ (а иных пар и не существует), то автомат переходит соответственно в одно из состояний 4, 1, 2, 3 и тем самым «запоминает» к следующему моменту $t+1$, что́ у него было на входе и на выходе перед этим. Соответствующая

диаграмма переходов изображена на рис. 55; легко заметить, что отвечающий этой диаграмме автомат реализует заданную о.-д. функцию.

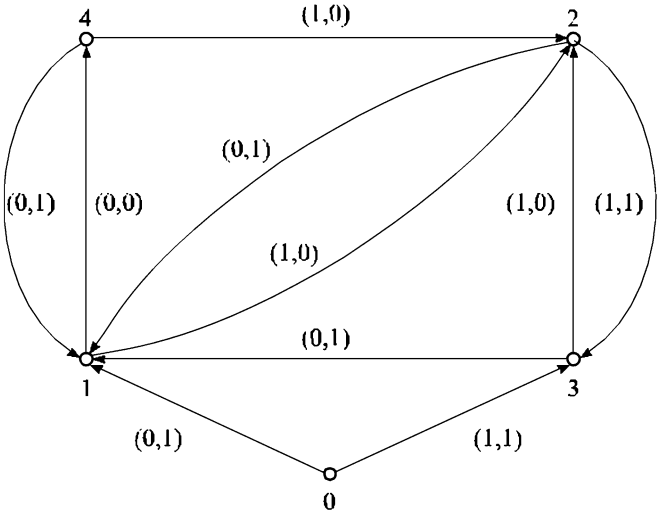


Рис. 55

з) На рис. 56,*а* приведена диаграмма переходов, по которой видно, что определяемый ею автомат в первый момент времени переходит из нулевого состояния в состояние 1, в следующий мо-

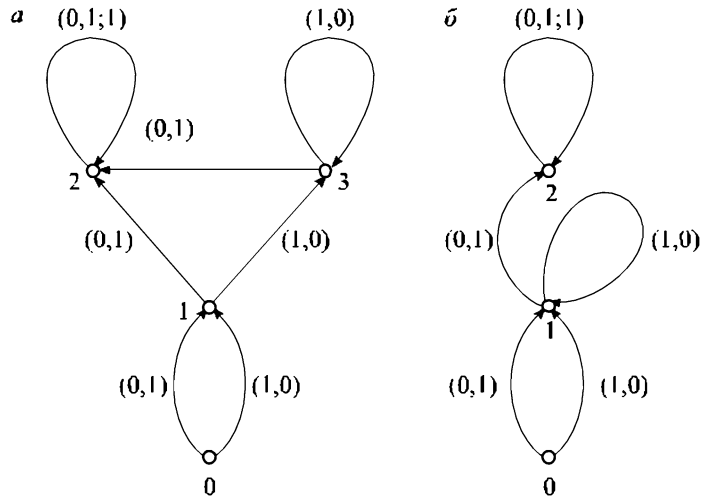


Рис. 56

мент времени переходит в зависимости от выдаваемого сигнала на выходе либо в состояние 2 (запоминая, что на выходе в этот момент выдана единица) либо в состояние 3 (запоминая нулевой сигнал на выходе в данный момент). При дальнейшей работе автомат переходит в состояние 2 при появлении единичного сигнала на выходе или в состояние 3 при появлении нулевого сигнала на выходе (две петли у вершины 2 с метками $(0, 1)$ и $(1, 1)$ склеены в одну с метками $(0, 1; 1)$); нетрудно заметить, что в данном случае реализуется заданная о.-д. функция. Заметим, что если рассмотреть занумерованное дерево для рассматриваемой о.-д. функции (а в данном случае это сделать нетрудно), то выяснится, что вес заданной о.-д. функции равен трём и ей соответствует диаграмма переходов с тремя вершинами, представленная на рис. 56,*б*.

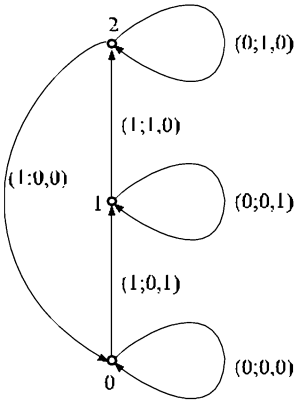


Рис. 57

4. Диаграмма переходов и таблица переходов для заданной пары о.-д. функций изображены на рис. 57 и в табл. 25 соответственно. На диаграмме переходов первая метка у каждой дуги задаёт сигнал на входе, а вторая метка — двухразрядное двоичное число — задаёт сигналы на выходах автомата (эти сигналы отвечают соответствующим разрядам во входной и в выходных последовательностях). В таблице переходов 4-я и 8-я строки доопределены; в левой половине таблицы x, q_1, q_2 зависят от t.

Таблица 25

x	q_1	q_2	$y_1(t)$	$y_2(t)$	$q_1(t)$	$q_2(t)$
0	0	0	0	0	0	0
0	0	1	0	1	0	1
0	1	0	1	0	1	0
0	1	1	1	1	1	1
1	0	0	0	1	0	1
1	0	1	1	0	1	0
1	1	0	0	0	0	0
1	1	1	1	0	1	0

По таблице переходов составляем канонические уравнения:

$$\begin{cases} y_1(t) = \overline{x}(t)q_1(t-1) \vee x(t)q_2(t-1), \\ y_2(t) = \overline{x}(t)q_2(t-1) \vee x(t)\overline{q}_1(t-1)\overline{q}_2(t-1), \\ q_1(t) = \overline{x}(t)q_1(t-1) \vee x(t)q_2(t-1), \\ q_2(t) = \overline{x}(t)q_2(t-1) \vee x(t)\overline{q}_1(t-1)\overline{q}_2(t-1), \\ q_1(0) = q_2(0) = 0. \end{cases}$$

В соответствии с каноническими уравнениями строим схему автомата, представленную на рис. 58.

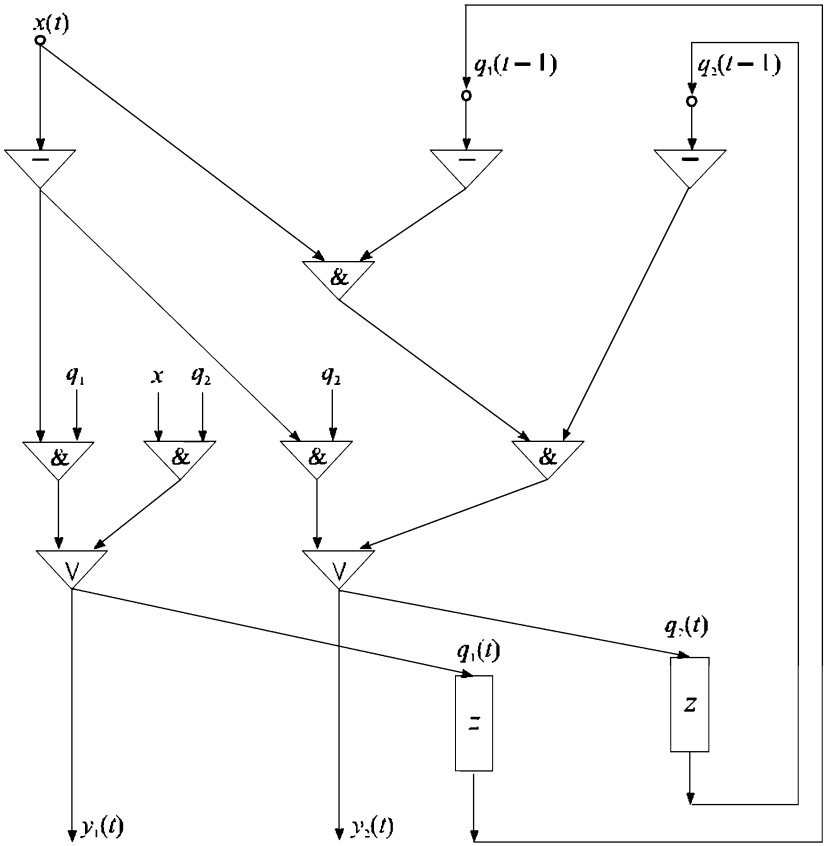


Рис. 58

5. Диаграмма переходов представлена на рис. 59. На этой диаграмме первая метка у каждой дуги (двузначное двоичное число) отвечает сигналам $x_1(t)$ и $x_2(t)$ на выходах; в состоянии 0

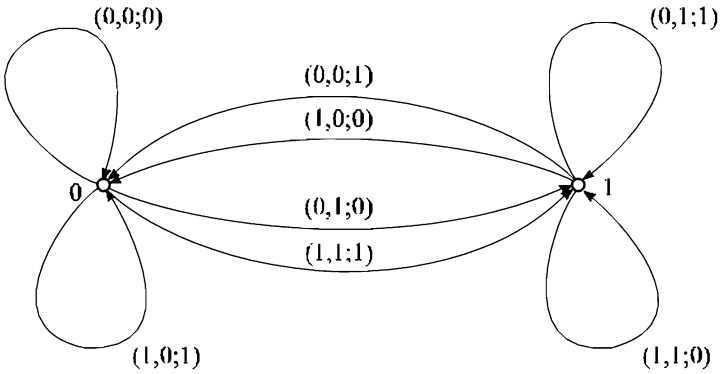


Рис. 59

запоминается нулевое значение сигнала $x_2(t)$ на втором входе, а в состоянии 1 запоминается единичное значение $x_2(t)$.

Схема автомата, реализующего заданную о.-д. функцию, приведена на рис. 60.

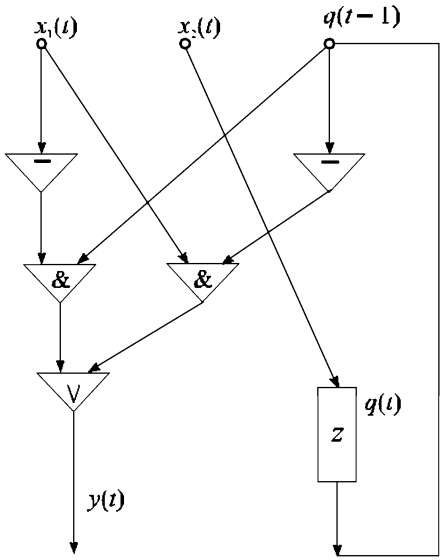


Рис. 60

6. б) В схеме автомата на рис. 28 выделим схему из функциональных элементов и отдельно элементы задержки в соответствии с «каноническим» представлением схемы автомата на

рис. 21. Полученная каноническая схема автомата изображена на рис. 61; по этой схеме легко находим канонические уравнения:

$$\begin{cases} y(t) = \overline{x}(t)q_2(t-1), \\ q_1(t) = \overline{x}(t), \\ q_2(t) = q_1(t-1), \\ q_1(0) = q_2(0) = 0. \end{cases}$$

По каноническим уравнениям построить таблицу переходов и диаграмму переходов уже совсем нетрудно.

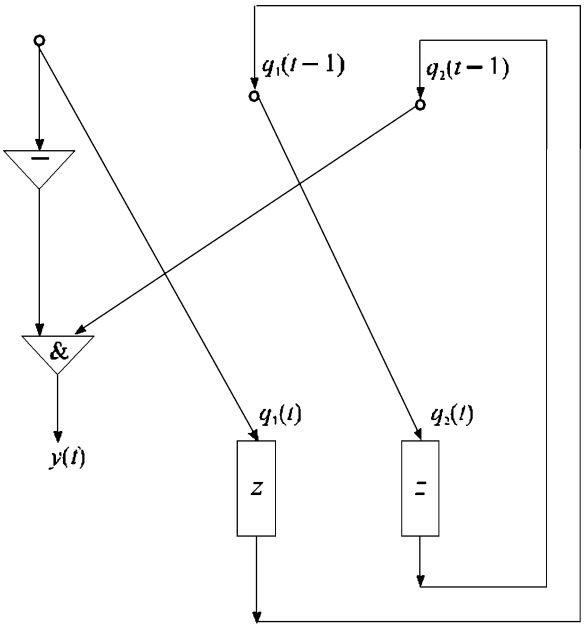


Рис. 61

К главе VIII

2. в) Можно взять, например, машину Тьюринга, задаваемую следующей системой команд (справа приводятся неформальные пояснения на содержательном уровне).

$q_1 1 \to q_1 1Л$, Головка проходит справа налево правый массив M из единиц.

$q_1 0 \to q_2 0Л$,

$q_2 0 \to q_2 0Л$, Головка проходит справа налево «разделительный» массив из единиц.

$q_2 1 \to q_3 1 Л$, Головка достигает крайней правой единицы в левом массиве M_1 единиц. Далее условный переход. Если в состоянии q_3 головка считывает 1, то это означает, что в M_1 более одной единицы; в соответствии с этим условием происходит дальнейшая работа машины. Если в состоянии q_3 головка считывает 0, то это означает, что в M_1 одна единица; в соответствии с этим условием организуется дальнейшая работа машины.

$q_3 1 \to q_4 1 П$,

$q_4 1 \to q_5 0 П$,
$q_5 0 \to q_5 0 П$, В M_1 стирается одна крайняя правая единица, и далее в состоянии q_5 происходит сдвиг по нулям вправо.

$q_5 1 \to q_6 0 П$, Головка достигает крайней левой единицы в M и стирает её.

$q_6 0 \to q_6 0 С$, Если в состоянии q_6 головка считывает 0, то это означает, что в M_1 больше единиц, чем в M; машина «зацикливается».

$q_6 1 \to q_2 1 Л$, Если в состоянии q_6 головка считывает 1, то это означает, что в массивах M и M_1 стёрто по одной единице, итерация завершена и машина переходит к следующей итерации.

$q_3 0 \to q_7 0 П$,

$q_7 1 \to q_8 0 П$,
$q_8 0 \to q_8 0 П$, В состоянии q_7 головка стирает последнюю единицу в левом массиве и в состоянии q_8 по «разделительным» нулям возвращается к правому массиву единиц.

$q_8 1 \to q_9 0 П$, Дойдя до левой единицы в массиве M, головка стирает её, переходит в состояние q_9 и сдвигается вправо.

$q_9 1 \to q_9 1 С$, Если в состоянии q_9 головка считывает 1, то это означает, что в M больше единиц, чем в M_1; машина зацикливается.

$q_9 0 \to q_0 0 С$. Если в состоянии q_9 головка считывает 0, то это означает, что в M и в M_1 единиц поровну; машина останавливается.

3. В зависимости от значений q и q' рассмотрим четыре случая: 1) $(q, q') = (q_1, q_1)$; 2) $(q, q') = (q_0, q_0)$; 3) $(q, q') = (q_1, q_0)$; 4) $(q, q') = (q_0, q_1)$.

1) Машина Тьюринга задаётся системой команд вида

$$q_1 0 \to q_1 a S,$$
$$q_1 1 \to q_1 a' S'.$$

Такая машина, очевидно, не применима ни к какому слову (при любых $a, a' \in \{0, 1\}$ и $S, S' \in \{\text{Л}, \text{П}, \text{С}\}$). Обозначим через K_1 класс эквивалентности, содержащий рассматриваемые в данном случае машины Тьюринга.

2) Имеем систему

$$q_1 0 \to q_0 a S,$$
$$q_1 1 \to q_0 a' S'.$$

Если машина Тьюринга M применима к слову A и результатом её работы оказывается слово B, то будем обозначать это в виде $A(M)B$. Пустое (т. е. не содержащее единиц) слово на ленте условимся обозначать через $\widetilde{0}$, а всякое непустое слово с конечным числом единиц — через $\widetilde{0}\widetilde{a}01^n\widetilde{0}$, где подслово $\widetilde{a}$ начинается с единицы (или отсутствует), слева от $\widetilde{a}$ и справа от массива единиц 1^n — одни лишь нули (предполагается, что только такие слова могут быть записаны на ленте в начальной конфигурации).

Если $(a, a') = (0, 0)$, то рассматриваемой машине M отвечает система команд

$$q_1 0 \to q_0 0 S,$$
$$q_1 1 \to q_0 0 S'.$$

В этом случае, как нетрудно заметить, $\widetilde{0}\widetilde{a}01^n\widetilde{0}(M)\widetilde{0}\widetilde{a}01^{n-1}\widetilde{0}$, $\widetilde{0}(M)\widetilde{0}$; очевидно, M принадлежит некоторому классу эквивалентности K_2, отличному от K_1. Если $(a, a') = (0, 1)$, то $\widetilde{0}\widetilde{a}01^n\widetilde{0}(M)\widetilde{0}\widetilde{a}01^n\widetilde{0}$, $\widetilde{0}(M)\widetilde{0}$; ясно, что $M \in K_3$, где K_3 — новый класс эквивалентности, отличный от K_1 и от K_2. Если $(a, a') = (1, 0)$, то $\widetilde{0}\widetilde{a}01^n\widetilde{0}(M)\widetilde{0}\widetilde{a}01^{n-1}\widetilde{0}$, $\widetilde{0}(M)\widetilde{0}1\widetilde{0}$ и $M \in K_4$ (K_4 — ещё один новый класс эквивалентности). Если $(a, a') = (1, 1)$, то $\widetilde{0}\widetilde{a}01^n\widetilde{0}(M)\widetilde{0}\widetilde{a}01^n\widetilde{0}$, $\widetilde{0}(M)\widetilde{0}1\widetilde{0}$ и $M \in K_5$.

3) Система команд выглядит так:

$$q_1 0 \to q_1 a S,$$
$$q_1 1 \to q_0 a' S'.$$

Если $(a, a') = (0, 0)$, то $\widetilde{0}\widetilde{a}01^n\widetilde{0}(M)\widetilde{0}\widetilde{a}01^{n-1}\widetilde{0}$, машина M неприменима к $\widetilde{0}$ и поэтому $M \in K_6$. Если $(a, a') = (0, 1)$, то $\widetilde{0}\widetilde{a}01^n\widetilde{0}(M)\widetilde{0}\widetilde{a}01^n\widetilde{0}$, M неприменима к $\widetilde{0}$ и $M \in K_7$. Если

$(a, a') = (1, 0)$ и $S \in \{\text{Л}, \text{П}\}$, то $\widetilde{0}\widetilde{a}01^n\widetilde{0}(M)\widetilde{0}\widetilde{a}01^{n-1}\widetilde{0}$, M неприменима к $\widetilde{0}$ и $M \in K_6$, а если $(a, a') = (1, 0)$ и $S = C$, то $\widetilde{0}\widetilde{a}01^n\widetilde{0}(M)\widetilde{0}\widetilde{a}01^{n-1}\widetilde{0}$, $\widetilde{0}(M)\widetilde{0}$ и $M \in K_2$. Если $(a, a') = (1, 1)$ и $S \in \{\text{Л}, \text{П}\}$, то $\widetilde{0}\widetilde{a}01^n\widetilde{0}(M)\widetilde{0}\widetilde{a}01^n\widetilde{0}$, M неприменима к $\widetilde{0}$ и $M \in K_7$, а если $(a, a') = (1, 1)$ и $S = C$, то $\widetilde{0}\widetilde{a}01^n\widetilde{0}(M)\widetilde{0}\widetilde{a}01^n\widetilde{0}$, $\widetilde{0}(M)\widetilde{0}1\widetilde{0}$ и $M \in K_5$.

4) Для M имеем систему команд

$$q_1 0 \to q_0 a S,$$
$$q_1 1 \to q_1 a' S'.$$

Если $(a, a') = (0, 0)$ и $S \in \{\text{П}, \text{С}\}$, то $\widetilde{0}\widetilde{a}01^n\widetilde{0}(M)\widetilde{0}\widetilde{a}01^{n-1}\widetilde{0}$, $\widetilde{0}(M)\widetilde{0}$ и $M \in K_2$, а если $(a, a') = (0, 0)$ и $S' = \text{Л}$, то $\widetilde{0}(M)\widetilde{0}$, $\widetilde{0}\widetilde{a}01^n\widetilde{0}(M)\widetilde{0}\widetilde{a}\widetilde{0}$ и $M \in K_8$ (K_8 — очередной класс эквивалентности). Если $(a, a') = (0, 1)$ и $S \in \{\text{Л}, \text{П}\}$, то $\widetilde{0}(M)\widetilde{0}$, $\widetilde{0}\widetilde{a}01^n\widetilde{0}(M)\widetilde{0}\widetilde{a}01^n\widetilde{0}$ и $M \in K_3$, а если $(a, a') = (0, 1)$ и $S' = \text{С}$, то $\widetilde{0}(M)\widetilde{0}$, M неприменима к $\widetilde{0}\widetilde{a}01^n\widetilde{0}$ и $M \in K_9$. Если $(a, a') = (1, 0)$ и $S' = \text{С}$, то $\widetilde{0}(M)\widetilde{0}1\widetilde{0}$, $\widetilde{0}\widetilde{a}01^n\widetilde{0}(M)\widetilde{0}\widetilde{a}01^n\widetilde{0}$ и $M \in K_5$; если $(a, a') = (1, 0)$ и $S' = \text{П}$, то $\widetilde{0}(M)\widetilde{0}1\widetilde{0}$, $\widetilde{0}\widetilde{a}01^n\widetilde{0}(M)\widetilde{0}\widetilde{a}01^{n-1}01\widetilde{0}$ и $M \in K_{10}$; если $(a, a') = (1, 0)$ и $S' = \text{Л}$, то $\widetilde{0}(M)\widetilde{0}1\widetilde{0}$, $\widetilde{0}\widetilde{a}01^n\widetilde{0}(M)\widetilde{0}\widetilde{a}1\widetilde{0}$ и $M \in K_{11}$. Если $(a, a') = (1, 1)$ и $S' = \text{С}$, то $\widetilde{0}(M)\widetilde{0}1\widetilde{0}$, M неприменима к $\widetilde{0}\widetilde{a}01^n\widetilde{0}$ и $M \in K_{12}$; если $(a, a') = (1, 1)$ и $S' = \text{П}$, то $\widetilde{0}(M)\widetilde{0}1\widetilde{0}$, $\widetilde{0}\widetilde{a}01^n\widetilde{0}(M)\widetilde{0}\widetilde{a}01^{n+1}\widetilde{0}$ и $M \in K_{13}$. Если $(a, a') = (1, 1)$ и $S' = \text{Л}$, то $\widetilde{0}(M)\widetilde{0}1\widetilde{0}$, $\widetilde{0}\widetilde{a}01^n\widetilde{0}(M)\widetilde{0}\widetilde{a}01^{n+1}\widetilde{0}$ и $M \in K_{14}$.

В итоге получаем 14 различных классов эквивалентности (или 14 неэквивалентных машин Тьюринга). Заметим, что если исключить начальную конфигурацию с пустым словом на ленте, то задача упрощается и в этом случае получается лишь 7 неэквивалентных машин Тьюринга.

4. а) Подходящую машину Тьюринга можно представить в виде композиции нескольких машин, например, так. Вначале работает машина M_1, которая выясняет значение параметра m. Если $m = 1$, то далее работает машина M_2 (для которой состояние q_3 является начальным), а если $m \geqslant 2$, то далее работает машина M_3 (для неё начальным состоянием будет q_4).

M_1:
1) $q_1 1 \to q_2 1\text{Л}$,
2) $q_2 0 \to q_3 0\text{П}$,
3) $q_2 1 \to q_4 1\text{П}$.

$$M_2:\quad \begin{array}{l} 4)\ q_3 1 \to q_3 1\Pi, \\ 5)\ q_3 0 \to q_4 0\Pi, \\ 6)\ q_4 0 \to q_0 1\text{С}. \end{array}$$

Машину M_3, в свою очередь, удобно представить в виде композиции машин $M_{3,1}$–$M_{3,4}$. Машина $M_{3,1}$ начинает составлять новый (правый) массив A_2 из единиц; она переносит всего лишь одну (крайнюю правую) единицу из заданного (левого) массива A_1 в A_2 и определяется следующими тремя командами.

$$M_{3,1}:\quad \begin{array}{l} 7)\ q_4 1 \to q_4 0\Pi, \\ 8)\ q_4 0 \to q_5 0\Pi, \\ 9)\ q_5 0 \to q_6 1\text{С}. \end{array}$$

Машина $M_{3,2}$ выясняет текущие размеры массива A_1. Для неё «заключительными» являются два состояния: q_9, если в A_1 осталась одна единица, и q_{10}, если в A_1 осталось не менее двух единиц.

$$M_{3,2}:\quad \begin{array}{l} 10)\ q_6 1 \to q_6 1\text{Л}, \\ 11)\ q_6 0 \to q_7 0\text{Л}, \\ 12)\ q_7 0 \to q_7 0\text{Л}. \\ 13)\ q_7 1 \to q_8 1\text{Л}, \\ 14)\ q_8 0 \to q_9 0\Pi, \\ 15)\ q_8 1 \to q_{10} 1\Pi. \end{array}$$

При переходе $M_{3,2}$ в состояние q_9 в A_1 имеется одна единица, и перенос этой единицы в A_2, а также восстановление левого массива A_1 осуществляет машина $M_{3,3}$.

$$M_{3,3}:\quad \begin{array}{l} 16)\ q_9 1 \to q_9 0\Pi, \\ 17)\ q_9 0 \to q_{11} 1\Pi, \\ 18)\ q_{11} 0 \to q_{11} 1\Pi. \\ 19)\ q_{11} 1 \to q_{12} 0\Pi, \\ 20)\ q_{12} 1 \to q_{12} 1\Pi, \\ 21)\ q_{12} 0 \to q_{13} 1\Pi. \\ 22)\ q_{13} 0 \to q_0 1\text{С}. \end{array}$$

Если же $M_{3,2}$ после выполнения 15-й команды переходит в состояние q_{10}, то это свидетельствует о наличии в A_1 всё ещё не менее двух единиц, и в этом случае далее работает машина $M_{3,4}$ — она переносит очередную единицу из A_1 в A_2, после чего начинает работать машина $M_{3,2}$.

23) $q_{10}1 \to q_{10}0\Pi$,
24) $q_{10}0 \to q_{14}0\Pi$,
$M_{3,4}$: 25) $q_{14}0 \to q_{14}0\Pi$.
26) $q_{14}1 \to q_{15}1\Pi$,
27) $q_{15}1 \to q_{15}1\Pi$,
28) $q_{15}0 \to q_{6}1\text{С}$.

Для некоторых пар qa (например, для $q_1 1$) команды с qa в левой части в приведённом списке команд нет; это означает, что соответствующая ситуация (когда головка в состоянии q считывает в обозреваемой клетке символ a) при работе машины с заданными начальными словами не встретится. Заметим также, что нашей главной заботой было то, чтобы работа машины была представлена достаточно наглядно (наверное, существуют подходящие машины и с меньшим числом состояний).

б) Подходящую машину Тьюринга можно задать следующей системой команд:

1) $q_1 1 \to q_2 1\text{Л}$,
2) $q_2 0 \to q_0 0\Pi$,
3) $q_2 1 \to q_3 1\Pi$,
4) $q_3 1 \to q_3 0\Pi$,
5) $q_3 0 \to q_4 1\text{С}$,
6) $q_4 1 \to q_5 1\text{Л}$,
7) $q_5 0 \to q_4 0\text{Л}$,
8) $q_5 1 \to q_6 0\Pi$,
9) $q_4 0 \to q_7 0\Pi$,
10) $q_6 1 \to q_8 1\Pi$,
11) $q_8 0 \to q_6 0\Pi$,
12) $q_6 0 \to q_4 1\text{С}$,
13) $q_7 0 \to q_9 0\Pi$,
14) $q_9 1 \to q_7 1\Pi$,
15) $q_9 0 \to q_{10} 0\text{Л}$,
16) $q_{10} 0 \to q_{10} 0\text{Л}$,
17) $q_{10} 1 \to q_0 1\text{С}$.

При выполнении первых пяти команд выясняется, какое из условий выполняется: $m = 1$ или $m > 1$? В случае $m = 1$ выполняются первые две команды и на этом работа машины заканчивается. Если же $m > 1$, то выполняются первые пять команд, конфигурация $1q_1$ переводится в конфигурацию $1^{m-1}01q_4$ и машина дальше работает так.

Головка перемещается влево до тех пор, пока не обнаруживает на ленте два рядом стоящих нуля или две рядом стоящие единицы (команды 6–9). В первом случае нужное слово на ленте

уже записано и нужно только, чтобы головка сместилась вправо до крайней правой единицы и остановилась; это осуществляется командами 13–17. Во втором случае левая из обнаруженных двух рядом стоящих единиц заменяется на нуль, а очередная единица «переносится» и добавляется к правой части подслова $0101\ldots01$, головка возвращается в состояние q_4 (команды 10–12) и далее осуществляется очередная итерация, начинающаяся с выполнения 6-й команды.

в) *Указание.* Воспользоваться результатом задачи а). Взять машину M, для которой $\widetilde{0}1^m\widetilde{0}(M)\widetilde{0}1^m01^m\widetilde{0}$, заключительное состояние q_0 для M сделать начальным состоянием q_1' для машины M', задаваемой командами

$$1)\ q_1'1 \to q_2'0\text{Л},$$
$$2)\ q_2'1 \to q_2'1\text{Л},$$
$$3)\ q_2'0 \to q_3'1\text{П},$$
$$4)\ q_3'1 \to q_3'1\text{П},$$
$$5)\ q_3'0 \to q_0'0\text{Л}.$$

Машина M' перерабатывает слово $\widetilde{0}1^m01^m\widetilde{0}$ в слово $\widetilde{0}1^{2m}\widetilde{0}$; композиция машин M и M' даёт нужную машину.

г) Одну из возможных машин зададим следующей системой команд:

$$1)\ q_11 \to q_20\text{Л},$$
$$2)\ q_21 \to q_21\text{Л},$$
$$3)\ q_20 \to q_30\text{Л},$$
$$4)\ q_31 \to q_31\text{Л},$$
$$5)\ q_30 \to q_40\text{Л},$$
$$6)\ q_40 \to q_51\text{П},$$
$$7)\ q_50 \to q_60\text{П},$$
$$8)\ q_61 \to q_61\text{П},$$
$$9)\ q_60 \to q_70\text{П},$$
$$10)\ q_70 \to q_80\text{Л},$$
$$11)\ q_80 \to q_00\text{Л},$$
$$12)\ q_71 \to q_91\text{П},$$
$$13)\ q_91 \to q_91\text{П},$$
$$14)\ q_90 \to q_{10}0\text{Л},$$
$$15)\ q_{10}1 \to q_{11}0\text{Л},$$
$$16)\ q_{11}1 \to q_{11}1\text{Л},$$
$$17)\ q_{11}0 \to q_{12}0\text{Л},$$
$$18)\ q_{12}1 \to q_{12}1\text{Л},$$
$$19)\ q_{12}0 \to q_{13}0\text{Л},$$
$$20)\ q_{13}1 \to q_{13}1\text{Л},$$
$$21)\ q_{13}0 \to q_{14}1\text{П},$$

22) $q_{14}1 \to q_{14}1\Pi$,
23) $q_{14}0 \to q_6 0\Pi$.

Требуемое слово получается «переносом» s единиц из правого массива единиц A_1 через средний массив A_2 (с неизменным числом единиц, равным m) в левый массив A_3 (отсутствующий в начальной конфигурации).

После выполнения первых семи команд получается конфигурация

$$\ldots 0010\underset{q_6}{1}\overbrace{1\ldots 1}^{m}0\overbrace{11\ldots 1}^{s-1}00\ldots.$$

В массив A_3 перенесена первая единица из массива A_1. В состоянии q_6 головка проходит массив A_2, а затем нуль, разделяющий массивы A_1 и A_2 (команды 8 и 9). Если в состоянии q_7 головка считывает нуль, то это означает, что в A_1 не осталось единиц и работа заканчивается (после выполнения команд 10 и 11).

Если в состоянии q_7 головка считывает единицу, то работа продолжается далее и головка в состоянии q_8 проходит через массив A_1, после достижения первого нуля переходит в состояние q_{10} (команды 12–14) и возвращается на шаг назад (к массиву A_1). В массиве A_1 крайняя правая единица заменяется на нуль, в состоянии q_{11} головка проходит справа налево массив A_1 и достигает нуля, разделяющего массивы A_1 и A_2 (команды 15–17). Далее в состоянии q_{12} головка проходит массив A_2, «разделительный» нуль (между массивами A_2 и A_3), в состоянии q_{13} проходит справа налево массив A_3, достигает клетки, в которой записан нуль, и заменяет этот нуль на единицу (команды 18–21). В состоянии q_{14} головка проходит массив A_3 слева направо, проходит через разделительный (между A_3 и A_2) нуль и меняет состояние на q_6 (команды 22 и 23). После этого снова выполняется очередная итерация (начиная с выполнения 8-й команды).

5. При вычислении $x + y$ можно использовать машину Тьюринга, приведенную в гл. 8, § 1.

При вычислении $x - y$ возможные заключительные конфигурации определим, например, так:

$$0\underset{q_0}{1}\overbrace{1\ldots 1}^{x-y+1}0 \text{ в случае } x - y > 0,$$

$$0\underset{q_0}{1}\overbrace{1\ldots 1}^{y-x+1}0 \text{ в случае } x - y < 0,$$

$$0\underset{q_0}{1}0 \text{ в случае } x - y = 0.$$

Одну из возможных машин можно задать следующей системой команд:

1) $q_1 1 \to q_2 0 Л$,
2) $q_2 0 \to q_0 0 Л$,
3) $q_2 1 \to q_3 1 Л$,
4) $q_3 1 \to q_3 1 Л$,
5) $q_3 0 \to q_4 0 Л$,
6) $q_4 1 \to q_5 1 Л$,
7) $q_5 0 \to q_6 0 П$,
8) $q_6 1 \to q_7 0 П$,
9) $q_7 0 \to q_0 1 С$,
10) $q_5 1 \to q_8 1 Л$,
11) $q_8 1 \to q_8 1 Л$,
12) $q_8 0 \to q_9 0 П$,
13) $q_9 1 \to q_{10} 0 П$,
14) $q_{10} 1 \to q_{10} 1 П$,
15) $q_{10} 0 \to q_{11} 0 П$,
16) $q_{11} 1 \to q_{11} 1 П$,
17) $q_{11} 0 \to q_1 0 Л$.

Машину Тьюринга для вычисления xy можно представить в виде композиции следующих машин. Первая машина M_1 проверяет: отличны ли от нуля множители x и y? Если хотя бы один множитель равен нулю, то M_1 переходит в заключительную конфигурацию

$$K_1 : \underset{q_0}{0\,1\,0}$$

и на этом работа заканчивается. Если оба множителя x и y отличны от нуля, то заключительной конфигурацией для M_1 будет

$$K_2 : 0\overbrace{11\ldots1}^{x-1}0\overbrace{11\ldots1}^{y}0\overbrace{11\ldots1}^{y+1}\underset{q_0}{0}.$$

Далее работает машина M_2, для которой конфигурация K_2 будет начальной, а i-й $(i = 1, 2, \ldots, x)$ этап работы начинается с

$$K_i : 0\overbrace{11\ldots1}^{x-i}0\overbrace{11\ldots1}^{y}0\overbrace{11\ldots1}^{iy+1}0$$
$$\qquad\qquad\quad A \qquad\qquad B \qquad\qquad C$$

(головка обозревает крайнюю правую клетку с единицей). На i-м этапе M_2 проверяет: остались ли единицы в массиве A? Если остались, то в A стирается крайняя левая единица, а в массив C добавляются y единиц, после чего i-й этап заканчивается и начинается $(i+1)$-й. Если на i-м этапе обнаруживается, что в A не осталось единиц, то слово на ленте остаётся неизменным, а головка возвращается назад (вправо) до крайней правой клетки с единицей и на этом работа заканчивается.

6. Пусть $A = \{a_0, a_1, \ldots, a_r\}$ и $Q = \{q_0, q_1, \ldots, q_n\}$ — соответственно внешний и внутренний алфавиты заданной машины M. Добавим к Q ещё n новых состояний $q'_1, \ldots, q'_n$, а в системе команд машины M каждую команду

$$q_i a_j \to q_k a_l \text{С},$$

$k \in \{1, \ldots, n\}$, $l \in \{0, 1, \ldots, r\}$, заменим на $(r+1)$ команд

$$\begin{gathered} q_i a_j \to q'_k a_l \text{П}, \\ q'_k a_0 \to q_k a_0 \text{Л}, \\ q'_k a_1 \to q_k a_1 \text{Л}, \\ \ldots\ldots\ldots\ldots \\ q'_k a_r \to q_k a_r \text{Л}. \end{gathered}$$

Получим машину Тьюринга M', в программе которой отсутствует символ С и которая эквивалента исходной машине M.

7. а) Возможна следующая программа:

$$\begin{gathered} 1)\ q_1 1 \to q_2 0 \text{Л}, \\ 2)\ q_2 1 \to q_3 0 \text{Л}, \\ 3)\ q_3 1 \to q_3 1 \text{Л}, \\ 4)\ q_3 0 \to q_4 1 \text{Л}, \\ 5)\ q_4 0 \to q_5 1 \text{П}, \\ 6)\ q_5 1 \to q_5 1 \text{П}, \\ 7)\ q_5 0 \to q_0 0 \text{Л}, \\ 8)\ q_2 0 \to q_6 0 \text{Л}, \\ 9)\ q_6 0 \to q_7 0 \text{Л}, \\ 10)\ q_7 0 \to q_0 1 \text{С}. \end{gathered}$$

б) Возможна следующая программа:

$$\begin{gathered} 1)\ q_1 1 \to q_1 1 \text{Л}, \\ 2)\ q_1 0 \to q_2 0 \text{Л}, \end{gathered}$$

$$
\begin{aligned}
&3)\ q_2 0 \to q_3 0\text{Л},\\
&\ldots\ldots\ldots\ldots\ldots\\
&m+1)\ q_m 0 \to q_0 0\text{Л},\\
&m+2)\ q_2 1 \to q_1 1\text{С},\\
&m+3)\ q_3 1 \to q_1 1\text{С},\\
&\ldots\ldots\ldots\ldots\ldots\\
&2m)\ q_m 1 \to q_1 1\text{С}.
\end{aligned}
$$

8. Одну из возможных машин Тьюринга представим в виде композиции машин M_1–M_3. Пусть $a_0 = 0$, $a_1 = 1$, $a_2 = 2$ и $\widetilde{a} = a^{(1)}a^{(2)}\ldots a^{(m)}$, где $a^{(i)} \in \{1, 2\}$, $i = 1, \ldots, m$. Машина M_1 проверяет, какое из условий выполняется: $m = 1$ или $m > 1$? Если $m = 1$, то уже само исходное слово $\widetilde{a}$ можно считать «инвертированным», так что начальная конфигурация становится и заключительной, работа всей машины на этом заканчивается. Если $m > 1$, то M_1 переводит начальную конфигурацию

$$K_1 = \ldots 00a^{(1)}a^{(2)}\ldots \underset{q_1}{a^{(m)}}\, 00\ldots$$

в заключительную (для M_1) конфигурацию

$$K_2 = \ldots 00a^{(1)}a^{(2)}\ldots a^{(m-1)}0\, \underset{q_6}{a^{(m)}}\, 00\ldots.$$

Система команд для M_1:

$$
\begin{aligned}
&1)\ q_1 1 \to q_2 1\text{Л},\\
&2)\ q_1 2 \to q_2 2\text{Л},\\
&3)\ q_2 0 \to q_0 0\text{П},\\
&4)\ q_2 1 \to q_3 1\text{П},\\
&5)\ q_2 2 \to q_3 2\text{П},\\
&6)\ q_3 1 \to q_4 0\text{П},\\
&7)\ q_3 2 \to q_5 0\text{П},\\
&8)\ q_4 0 \to q_6 0\text{С},\\
&9)\ q_5 0 \to q_6 2\text{С}.
\end{aligned}
$$

Конфигурация K_2 является начальной для машины M_2, которая осуществляет $m - 1$ итераций и получает (в заключительной конфигурации) инвертированное слово. В результате i-й итерации ($i = 1, \ldots, m-1$) осуществляется переход от конфигурации

$$K_i = \ldots 00a^{(1)}a^{(2)}\ldots a^{(m-i)}\underbrace{0\ldots 0}_{i}a^{(m)}\ldots \underset{q_6}{a^{(m-i+1)}}\, 00\ldots$$

к конфигурации

$$K_{i+1}=\ldots 00\underbrace{a^{(1)}a^{(2)}\ldots a^{(m-i-1)}}_{A}\underbrace{0\ldots 0}_{i+1}\underbrace{a^{(m)}\ldots a^{(m-i+1)}\,a^{(m-i)}_{q_6}}_{B}00\ldots .$$

Система команд для машины M_2 выглядит так.

10) $q_6 1 \to q_6 1Л$,
11) $q_6 2 \to q_6 2Л$,
12) $q_6 0 \to q_7 0Л$,
13) $q_7 0 \to q_7 0Л$,
14) $q_7 1 \to q_8 1Л$,
15) $q_7 2 \to q_8 1Л$,
16) $q_8 0 \to q_9 0П$,
17) $q_8 1 \to q_{10} 1П$,
18) $q_8 2 \to q_{10} 2П$,

Головка проходит справа налево массив B (из единиц и двоек), а затем «разделительные» нули. Головка доходит до клетки с ненулевой буквой в слове A. Если в состоянии q_8 головка считывает нуль, то это означает, что в A осталась последняя буква $a^{(1)}$, которую надо перенести в B. Если в состоянии q_8 головка считывает единицу или двойку, то это означает, что в A не менее двух букв и выполняемая итерация ещё не последняя.

19) $q_9 1 \to q_{10} 1П$,
20) $q_9 2 \to q_{11} 2П$,
21) $q_{10} 0 \to q_{12} 0П$,
22) $q_{11} 0 \to q_{13} 0П$,
23) $q_{12} 1 \to q_{12} 1П$,
24) $q_{12} 2 \to q_{12} 2П$,
25) $q_{12} 0 \to q_0 1С$,
26) $q_{13} 1 \to q_{13} 1П$,
27) $q_{13} 2 \to q_{13} 2П$,
28) $q_{13} 0 \to q_0 2С$,

«Перенос» последней буквы (a_1) из левого (исходного) слова A в правое (инвертированное) слово B. Состояния q_{10}, q_{12} отвечают переносу единицы, а состояния q_{11}, q_{13} — переносу двойки.

29) $q_{10} 1 \to q_{14} 0П$,
30) $q_{10} 2 \to q_{15} 0П$,
31) $q_{14} 0 \to q_{14} 0П$,
32) $q_{14} 1 \to q_{16} 1П$,
33) $q_{14} 2 \to q_{16} 2П$,
34) $q_{16} 1 \to q_{16} 1П$,
35) $q_{16} 2 \to q_{16} 2П$,
36) $q_{16} 0 \to q_6 1С$,
37) $q_{15} 0 \to q_{15} 0П$,
38) $q_{15} 1 \to q_{17} 1П$,

Перенос очередной буквы $a^{(m-i)}$ $(m-i>1)$ из A в B. Состояния q_{14}, q_{16} отвечают переносу единицы, а состояния q_{15}, q_{17} — переносу двойки.

39) $q_{15}2 \to q_{17}2\text{П}$,

40) $q_{17}1 \to q_{17}1\text{П}$,

41) $q_{17}2 \to q_{17}2\text{П}$,

42) $q_{17}0 \to q_6 2\text{С}$.

9. Пусть $A = \{a_0, a_1, \ldots, a_r\}$ — внешний алфавит, $Q = \{q_0, q_1, \ldots, q_n\}$ — множество внутренних состояний головки машины Тьюринга M и $\widetilde{a} = a^{(1)}a^{(2)}\ldots a^{(l)}$ — непустое слово, записанное на ленте в начальной конфигурации K_1 (символы $a^{(1)}$ и $a^{(l)}$ в слове $\widetilde{a}$ — непустые, а слева и справа от $\widetilde{a}$ на ленте только пустые символы). Обозначим через $B_1, \ldots, B_l$ клетки на ленте с записанными в них символами $a^{(1)}, \ldots, a^{(l)}$ слова $\widetilde{a}$, через B — $(n+2)$-ю влево от B_1 клетку на ленте, а через B' — $(n+2)$-ю вправо от B_l клетку.

Если головка достигнет B (или B'), то далее она будет двигаться влево (соответственно вправо). Действительно, предположим, что в какой-то момент времени t_1 головка достигла B. В таком случае она в какой-то момент t_2, $t_2 < t_1$, обозревала клетку B_1 и затем сдвинулась влево от неё, а во все последующие моменты t_2+1, t_2+2, ..., t_1-1 обозревала только клетки (с пустыми символами) слева от B_1. Это означает, что в моменты t_2+1, t_2+2, ..., t_1-1 выполнялась последовательность команд вида

$$t_2+1)\ q'_1a_0 \to q'_2a_0S_1,$$
$$t_2+2)\ q'_2a_0 \to q'_3a_0S_2,$$
$$\ldots\ldots\ldots\ldots\ldots$$
$$t_1-1)\ q'_{t_1-t_2-1}a_0 \to q'_{t_1-t_2}a_0S_{t_1-t_2},$$

где $q'_1, q'_2, \ldots, q'_{t_1-t_2-1} \in \{q_1, \ldots, q_n\}$, а $S_1, S_2, \ldots, S_{t_1-t_2} \in \{\text{Л}, \text{П}, \text{С}\}$. От B_1 до B головка должна сдвинуться влево на $n+2$ клеток, для чего потребуется не менее, чем $n+2$ моментов; отсюда вытекает неравенство $t_1 - t_2 - 1 \geqslant n+2$. Из последнего неравенства следует, что в приведенной последовательности среди первых $t_1 - t_2 - 2$ команд найдутся по крайней мере две с одинаковыми левыми частями:

$$t_2+i)\ q'_ma_0 \to q'_{i+1}a_0S_i,$$
$$t_2+j)\ q'_ma_0 \to q'_{i+1}a_0S_i,$$

где $1 \leqslant i < j \leqslant t_1 - t_2 - 1$, а $q'_m \in \{q_1, \ldots, q_n\}$. Обозначим через D и D' клетки на ленте, обозреваемые головкой в моменты времени t_2+i и t_2+j соответственно. Совпадение D и D' означало бы, очевидно, зацикливание; в этом случае головка не смогла бы

достигнуть B. При нахождении D' справа от D головка, как нетрудно заметить, вернулась бы к B_1, а это исключается нашим предположением о том, что в рассматриваемые моменты от t_2+1 до t_1 головка обозревает клетки слева от B_1. Остаётся только допустить, что D' находится левее D, а это означает, что головка будет двигаться в левую сторону.

Предположим теперь, что в первые $n(l+2n+3)2^l$ моментов времени головка не достигает ни одной из клеток B, B'. Рассмотрим конфигурации, в каждой из которых головка обозревает клетку ленты, расположенную между B_1 и B'. По условию головка может только стирать непустые символы (точнее, заменять их на a_0) на ленте, поэтому в процессе работы машины в клетках $B_1,\ldots,B_l$ могут оказаться записанными не более, чем 2^l попарно различных слов. Головка может оказаться в одном из n состояний и обозревать одну из $l+2n+2$ клеток между B и B'. Следовательно, рассматриваемых конфигураций меньше, чем $n(l+2n+3)2^l$, и в процессе работы машины (если, конечно, она не остановится) до момента времени $t=n(l+2n+3)2^l$ какая-нибудь конфигурация повторится дважды, а это будет означать зацикливание машины.

10. *Указание.* Требуемую машину Тьюринга представить, например, в виде композиции машин M_1–M_4. Машина M_1 выясняет, какое из условий выполняется: $m=1$ или $m>1$? Если $m=1$, то M_1 переводит конфигурацию

$$K_1=\ldots 01q_100\ldots$$

в заключительную конфигурацию

$$K_0=\ldots 101q_000\ldots$$

и на этом работа всей машины заканчивается. Если же $m>1$, то M_1 переводит K_1 в

$$K_2=\ldots 0q_2\overbrace{11\ldots1}^{m}00\overbrace{11\ldots1}^{m}00\ldots$$

и далее начинает работать машина M_2 (для неё состояние q_2 — начальное).

Машина M_2 работает с конфигурациями вида

$$K^{(i)}=\ldots 00q_2\overbrace{11\ldots1}^{i}0\overbrace{11\ldots1}^{i+1}0\ldots0\overbrace{11\ldots1}^{m}00\overbrace{11\ldots1}^{m}00\ldots,$$

где $i \in \{1, \ldots, m\}$. Эта машина проверяет, какое из условий выполняется: $i = 1$ или $i > 1$? Если $i > 1$, то M_2 переводит $K^{(i)}$ в $K^{(i-1)}$; если же $i = 1$, то M_2 переводит $K^{(1)}$ в

$$K' = \ldots 0q_3 1011011101 \ldots 10\overbrace{11\ldots1}^{m}00\overbrace{11\ldots1}^{m}00\ldots$$

и далее работает машина M_3, которая переводит конфигурацию K' в

$$K'' = \ldots 0010100100010 \ldots 01\overbrace{00\ldots0}^{m}11\overbrace{00\ldots0}^{m}1q_4 00\ldots$$

После M_3 работает машина M_4, которая переводит конфигурацию K'' в заключительную конфигурацию

$$K_0 = \ldots 0010100100010 \ldots 01\overbrace{00\ldots0}^{m}\overbrace{11\ldots1}^{m}q_0 00\ldots.$$

К главе IX

1. а) $0, 10, 110, 111$.
б) $0, 100, 101, 1100, 1101, 1110, 1111$.
в) $00, 01, 100, 101, 110, 1110, 1111$.
Указание. Использовать кодовые деревья.

2. а), г) Нет. б), в) Да. *Указание.* Воспользоваться следствием 1 из теорем 28 (неравенство Крафта–Макмиллана) и 29.

3. а), б) Нет.

4. а) $0, 10, 110, 111$.
б) $1, 00, 0100, 0101, 0110, 0111$.

5. а) Например, $P = (0{,}5; 0{,}26; 0{,}06; 0{,}06; 0{,}06; 0{,}06)$, $\sum(\{0,1\}) = \{0, 10, 1100, 1101, 1110, 1111\}$.
б) Например, $P = (0{,}5; 0{,}13; 0{,}13; 0{,}06; 0{,}06; 0{,}06; 0{,}03; 0{,}03)$, $\sum(\{0,1\}) = \{0, 100, 101, 1110, 1111, 1100, 11010, 11011\}$.

6. Предположим, что в оптимальном двоичном коде $\sum(\{0,1\})$ число элементарных кодов максимальной длины нечётно. В этом случае для некоторого элементарного кода $\widetilde{b} = \widetilde{b}'a$ из $\sum(\{0,1\})$ в $\sum(\{0,1\})$ не окажется элементарного кода $\widetilde{b}'a$ (отличающегося от $\widetilde{b}$ последней буквой). Но в таком случае, не нарушая префиксности кода, можно заменить $\widetilde{b}$ на $\widetilde{b}'$, а это противоречит оптимальности кода.

7. а), б) Нет. См. задачу 6.
в), д) Да (выполняются условия теоремы 29).
г) Нет (нарушается неравенство Крафта–Макмиллана).

8. Для набора вероятностей $P = \left\{\frac{1}{2} + \frac{1}{2^{15}} - \frac{1}{2^{20}}, \frac{1}{2^2}, \frac{1}{2^3}, \ldots, \right.$ $\left. \ldots, \frac{1}{2^{14}}, \frac{1}{2^{15}}, \frac{1}{2^{20}} \right\}$ код $\{0, 10, 110, \ldots, \underbrace{11\ldots1}_{14}0, \underbrace{11\ldots11}_{15}\}$, как нетрудно убедиться, является оптимальным; наибольшая длина слова в этом коде равна 15. Предположим, что существует оптимальный код Σ с подходящим набором вероятностей и наибольшей длиной элементарного кода $\widetilde{b}$, равной l, где $l \geqslant 16$. Пусть $\mathcal{D}$ — кодовое дерево для Σ, содержащее 16 концевых вершин (по числу элементарных кодов в Σ), v_l — концевая вершина в $\mathcal{D}$, отвечающая $\widetilde{b}$, а $v_0 e_1 v_1 e_2 v_2 \ldots v_{l-1} e_l v_l$ — путь, ведущий из корня v_0 дерева $\mathcal{D}$ в v_l ($v_0, v_1, \ldots, v_l$ — вершины, а $e_1, e_2, \ldots, e_l$ — дуги этого пути). Из каждой вершины v_i, $i = 1, \ldots, l-1$, помимо дуги e_{i+1} должна выходить вторая дуга e'_{i+1} — иначе была бы возможна операция усечения. Но тогда через дуги $e'_1, \ldots, e'_{l-1}$ проходили бы пути из корня v_0 по крайней мере в $l-1$ попарно различных концевых вершин, отличных от v_l, а значит, всего концевых вершин в $\mathcal{D}$ (и элементарных кодов в Σ) оказалось бы не менее 17; получаем противоречие. В итоге имеем $\max\limits_{\Sigma \in S} l(\Sigma) = 15$.

Число концевых вершин в m-ярусном кодовом дереве не превосходит 2^m (формально это легко доказать индукцией по m). В нашем случае кодовое дерево должно содержать 16 концевых вершин; следовательно, в нём должно быть не менее 4 ярусов, а значит, и наибольшая длина кодового слова будет не менее 4. Для набора вероятностей $p_1 = \ldots = p_{15} = \frac{1-2^{-20}}{15}$, $p_{16} = 2^{-20}$ оптимальный код, как нетрудно убедиться, будет состоять из 16 наборов длины 4. С учётом нижней оценки получаем $\min\limits_{\Sigma \in S} l(\Sigma) = 4$.

9. $H_6 = \{(000000), (010101), (100110), (110011), (111000), (101101), (011110), (001011)\}$.

10. а) 010101. б) 101101. в) 1110000. г) 0110011. д) 11101101001. е) 001110101001. ж) 00010110010111. з) 11101010011100100.

11. а) 110011. б) 110011. в) 101100111. г) 101100111. д) 00111000101000. е) 00111000101000.

К главе X

1. а), б) *Указание.* Воспользоваться жадным алгоритмом (или алгоритмом ближайшего соседа).

2. $vafgcdekv'$.

3. а) Простое сечение $S = \{b, g, h, l\}$ обладает наименьшей пропускной способностью $c_{min} = 9$. Согласно теореме Форда–Фалкерсона максимальная величина потока в заданной сети равна 9 и в качестве максимального потока можно взять пару $\{f, w\}$, полагая $f(d) = 0$, $f(e) = f(h) = 1$, $f(g) = f(l) = 2$, $f(a) = 3$, $f(b) = f(k) = 4$, $f(i) = 6$, $f(c) = 7$ и задавая (функцией w) ориентацию рёбер g, h, l слева направо относительно сечения S (чтобы эти рёбра оказались в S прямыми).

б) *Указание.* В качестве сечения с минимальной пропускной способностью взять $\{c, h, e, k\}$.

4. Для произвольной вершины v заданного графа S через $P(v)$ обозначим сумму пропускных способностей всех рёбер, инцидентных вершине v. Если вершину v выбрать в качестве одного из полюсов, то инцидентные вершине v рёбра составят сечение получающейся сети и максимальный поток через такую сеть по теореме Форда–Фалкерсона не может превосходить $P(v)$. Наибольшей величины, равной 9, $P(v)$ достигает на вершинах из $\{\alpha, \beta, \gamma, \delta\}$, указанных на рис. 62. Существуют две неизоморфные сети S_1 и S_2 с полюсами из множества $\{\alpha, \beta, \gamma, \delta\}$. В первой полюсами являются вершины α и β (ей изоморфна сеть с полюсами γ и δ), а во второй — вершины α и γ (этой сети изоморфны 3 сети с полюсами β и γ, α и β, β и δ). Задавая ориентацию и нагрузки рёбер сети S_1, как указано на рис. 62 (нагрузка каждого неотмеченного ребра предполагается равной нулю, а ориентация выбирается произвольным образом), получим поток максимальной величины. Для сети S_2 поток максимальной величины получается при переориентации рёбер (γ, β), $(\varkappa, \nu)$, (λ, μ), $(\gamma, \varkappa)$, $(\varkappa, \lambda)$, (β, ν), (ν, μ) (рис. 62) на противоположное направление.

5. Одно из максимальных паросочетаний может быть составлено из 2^{n-1} рёбер $((0, \widetilde{\sigma}), (1, \widetilde{\sigma}))$, где $\widetilde{\sigma} \in E^{n-1}$. В этом паросочетании для каждой вершины $(\alpha, \widetilde{\sigma})$, $\alpha \in \{0, 1\}$, n-мерного куба E^n есть инцидентное этой вершине ребро, а это означает, что паросочетание является максимальным.

6. Максимальная величина потока равна n. Для $n = 1, 2$ это очевидно. Ниже докажем это для $n \geqslant 3$. Без ограничения общности в качестве выбранных вершин можно рассматривать

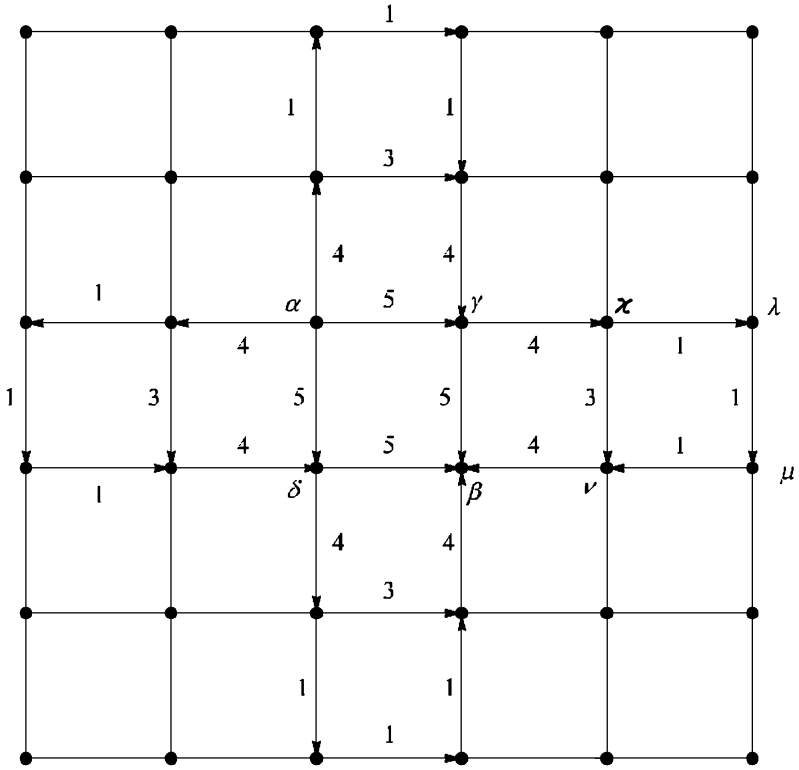


Рис. 62

вершину $\widetilde{0} = (00\ldots0)$ и вершину $\widetilde{\alpha}_k = (11\ldots100\ldots0)$, содержащую k единиц, где $k \in \{1, \ldots, n\}$. Индукцией по n докажем следующее

Утверждение. *Для любой вершины* $\widetilde{\alpha}_k = (1\ldots10\ldots0)$ *n-мерного куба* B^n *с k единицами, где* $k \geqslant 1$, *существуют n попарно не пересекающихся по рёбрам цепей, соединяющих* $\widetilde{\alpha}_k$ *с вершинами* $(10\ldots0)$, $(01\ldots0)$, ..., $(00\ldots1)$ *и не содержащих вершину* $(00\ldots0)$.

Базис индукции для 3-мерного куба B^3 легко установить непосредственной проверкой. Пусть утверждение справедливо для B^3, ..., B^{n-1}. Докажем его для B^n. Вначале предположим, что $k = 1$. Первая цепь (вырожденная) Z_1 соединяет $\widetilde{\alpha}_1$ с $\widetilde{\alpha}_1$ и не содержит рёбер. Цепь Z_i, $i = 2, \ldots, n$, содержит два ребра $(\widetilde{\alpha}_1, \widetilde{\beta}_i)$ и $(\widetilde{\beta}_i, \widetilde{\gamma}_i)$, в которых вершина $\widetilde{\beta}_i$ содержит ровно две единицы в

1-м и в i-м разрядах, а $\widetilde{\gamma}_i$ содержит ровно одну единицу в i-м разряде.

Далее считаем $k \geqslant 2$. Разобьём B^n на два подкуба B_0^{n-1} и B_1^{n-1}, где B_0^{n-1} содержит 2^{n-1} вершин $(0, \sigma_2, \ldots, \sigma_n)$ с нулём в 1-м разряде, а B_1^{n-1} содержит 2^{n-1} вершин $(1, \sigma_2, \ldots, \sigma_n)$ с единицей в 1-м разряде. В соответствии с предположением индукции в подкубе B_1^{n-1} выделим $(n-1)$ штук не пересекающихся по рёбрам и не содержащих вершину $(10\ldots0)$ (из B^n) цепей Z_1, ..., Z_{n-1}, соединяющих $\widetilde{\alpha}_k$ с вершинами $(110\ldots0)$, $(101\ldots0)$, ..., $(10\ldots01)$ соответственно. К цепям $Z_2, \ldots, Z_{n-1}$ добавим по одному ребру: к Z_2 добавим ребро, соединяющее вершину $(101\ldots0)$ из B_1^{n-1} с вершиной $(001\ldots0)$ из B_0^{n-1}, ..., к Z_{n-1} добавим ребро, соединяющее вершину $(10\ldots01)$ с вершиной $(00\ldots01)$. К Z_1 добавим ребро, соединяющее вершину $(110\ldots0)$ с вершиной $(100\ldots0)$.

Ещё одну цепь Z_n, соединяющую $\widetilde{\alpha}_k$ с вершиной $(010\ldots0)$ в B^n, построим так. Вершину $\widetilde{\alpha}_k$ из B_1^{n-1} соединим ребром e_1 с вершиной $\widetilde{v}_1 = (01\ldots10\ldots0)$ из B_0^{n-1}. Далее ребром e_i соединяем вершину $\widetilde{v}_{i-1}$ с вершиной $\widetilde{v}_i$, получающейся из $\widetilde{v}_{i-1}$ заменой в $(01\ldots10\ldots0)$ крайней правой единицы на нуль, $i = 2, 3, \ldots, k-1$. Ребром e_k соединим вершину $\widetilde{v}_{k-1} = (010\ldots0)$ из B_0^{n-1} с вершиной $\widetilde{v}_k = (110\ldots0)$ из B_1^{n-1}. Все рёбра цепи Z_n не принадлежат B_1^{n-1}, а значит, не принадлежат цепям $Z_1, \ldots, Z_n$. В итоге по построению получаем n штук цепей, соединяющих $\widetilde{\alpha}_k$ с вершинами $(10\ldots0)$, $(01\ldots0)$, ..., $(00\ldots1)$, попарно не пересекающихся по рёбрам и не проходящих через вершину $(0\ldots0)$.

Из утверждения вытекает

Следствие. *Вершины $\widetilde{\alpha}_k$ и $\widetilde{0}$ в B^n можно соединить не пересекающимися между собой по рёбрам цепями.*

Действительно, в соответствии с утверждением возьмем n непересекающихся цепей $Z_1, \ldots, Z_n$ с общим началом в $\widetilde{\alpha}_k$ и с концами в $(010\ldots0)$, $(0010\ldots0)$, $(00\ldots01)$, $(10\ldots0)$; конец каждой цепи Z_i соединим ребром с вершиной $\widetilde{0}$.

Далее остаётся ориентировать указанные в следствии цепи от истока $\widetilde{\alpha}_k$ к стоку $\widetilde{0}$, а нагрузку каждого ребра принять равной 1. Нагрузку на рёбра из B^n, не вошедшие в $Z_1, \ldots, Z_n$, полагаем равной нулю, а ориентацию этих рёбер выбираем произвольным образом. В итоге получим поток величины n. Этот поток в соответствии с теоремой Форда–Фалкерсона будет максимальным,

поскольку пропускная способность каждой (входной и выходной) звезды, составляющей сечение, равна n.

7. а) Пусть $\mathcal{D}$ — какое-то минимальное остовное дерево в B^n. Выделим в B^n $(n-1)$-мерный подкуб B_0^{n-1}, содержащий 2^{n-1} вершин $(0,\widetilde{\sigma})$ (с нулем в первом разряде), и $(n-1)$-мерный подкуб B_1^{n-1}, содержащий 2^{n-1} вершин $(1,\widetilde{\sigma})$. Докажем следующее

Утверждение 1. *В $\mathcal{D}$ нет ни одного ребра, соединяющего две вершины из B_1^{n-1}.*

Действительно, предположим противное, т. е. в B_1^{n-1} имеется ребро e, соединяющее две соседние вершины $(1,\widetilde{\sigma})$ и $(1,\widetilde{\sigma}')$ из B_1^{n-1}. Удалим e из $\mathcal{D}$; дерево $\mathcal{D}$ распадается на две связные компоненты $\mathcal{D}_1$ и $\mathcal{D}_2$, и вершины $(1,\widetilde{\sigma})$ и $(1,\widetilde{\sigma}')$ окажутся в разных компонентах: скажем, $(1,\widetilde{\sigma})\in\mathcal{D}_1$, а $(1,\widetilde{\sigma}')\in\mathcal{D}_2$. Для вершин из B_0^{n-1} возможны два случая: все вершины подкуба B_0^{n-1} окажутся в одной компоненте, например, в $\mathcal{D}_1$; найдутся две вершины $(0,\widetilde{\alpha})$ и $(0,\widetilde{\alpha}')$ такие, что $(0,\widetilde{\alpha})\in\mathcal{D}_1$, а $(0,\widetilde{\alpha}')\in\mathcal{D}_2$.

В первом случае при добавлении в $\mathcal{D}$ (вместо e) ребра $e'=$ $=((1,\widetilde{\sigma}'),(0,\widetilde{\sigma}'))$, соединяющего вершины $(1,\widetilde{\sigma}')$ из $\mathcal{D}_2$ и $(0,\widetilde{\sigma}')$ из B_0^{n-1}, снова получим некоторое остовное дерево $\mathcal{D}'$ для B^n. Но вес ребра e' меньше веса ребра e, вследствие чего и вес остова $\mathcal{D}'$ окажется меньше веса остова $\mathcal{D}$, а это противоречит предположению о минимальности $\mathcal{D}$.

Во втором случае вершины $(0,\widetilde{\alpha})$ и $(0,\widetilde{\alpha}')$ соединим простой цепью Z, целиком лежащей в B_0^{n-1}. В Z найдётся некоторое ребро e', которое не входит ни в $\mathcal{D}_1$, ни в $\mathcal{D}_2$ (т. е. оба конца ребра e' не принадлежат одновременно ни одной из компонент $\mathcal{D}_1$, $\mathcal{D}_2$). Поэтому при добавлении e' в $\mathcal{D}$ (вместо e) циклы не появятся, но с учётом третьего условия (характеристического свойства дерева) из теоремы 3 получится некоторое остовное дерево $\mathcal{D}'$ с весом, меньшим чем у $\mathcal{D}$, а это противоречит предположению о минимальности $\mathcal{D}$. Утверждение доказано.

В кубе B^n каждая вершина из подкуба B_1^{n-1} смежна ровно с одной вершиной из подкуба B_0^{n-1}; из этого факта и утверждения 1 следует

Утверждение 2. *Всякая вершина $(1,\widetilde{\sigma})$ из $\mathcal{D}\cap B_1^{n-1}$ является концевой и смежной с вершиной $(0,\widetilde{\sigma})$ из $\mathcal{D}\cap B_0^{n-1}$.*

Укажем теперь (по индукции) способ построения минимального остовного дерева для B^n.

На первом шаге находим минимальный остов $\mathcal{D}_1$ для B^1; очевидно, $\mathcal{D}_1$ совпадает с B^1.

Пусть выполнено i шагов и найден минимальный остов $\mathcal{D}_{i-1}$ для B^{i-1}, $i \geqslant 2$. Имея минимальный остов для подкуба B_0^i (в кубе B^i), минимальный остов $\mathcal{D}_i$ для B^i в соответствии с утверждением 2 можно получить «подвешиванием» к каждой вершине $(0, \widetilde{\sigma})$ из подкуба B_0^i (а все вершины из B_0^i входят в $\mathcal{D}_{i-1}$) висячего ребра $((0, \widetilde{\sigma}), (1, \widetilde{\sigma}))$, соединяющего вершину $(0, \widetilde{\sigma})$ из $\mathcal{D}_{i-1}$ с вершиной $(1, \widetilde{\sigma})$ из B_1^i (заметим, что сумма весов всех подвешиваемых на этом шаге рёбер равна сумме номеров всех вершин из B_0^i и, вообще говоря, не зависит от того, какой именно минимальный остов $\mathcal{D}_{i-1}$ был построен для B^{i-1}).

Пусть $Q(n)$ — сумма номеров всех вершин из B^n, а $P(n)$ — вес минимального остовного дерева для B^n. Из указанного выше способа построения минимального остовного дерева вытекает рекуррентное соотношение

$$P(n) = P(n-1) + Q(n-1),$$

из которого получаем

$$P(n) = P(1) + \sum_{m=1}^{n-1} Q(m).$$

Из последнего соотношения, учитывая, что $P(1) = 0$, а $Q(m) = \sum_{i=1}^{2^m-1} i = 2^{2m-1} - 2^{m-1}$, получаем

$$P(n) = \sum_{m=1}^{n-1} \left(2^{2m-1} - 2^{m-1}\right) = \frac{4^n}{6} - 2^{n-1} + \frac{1}{3}.$$

б) $(2^n - 1)2^{n-1}$. Для построения минимального остовного дерева $\mathcal{D}$ воспользуемся алгоритмом ближайшего соседа. В качестве исходной вершины, с которой начинается построение дерева, возьмём $v_0 = \widetilde{0}$ и на первом шаге ребром e_1 наименьшего веса 1 соединим вершину $\widetilde{0}$ с вершиной $v_1 = (0, \ldots, 0, 1)$. Получим дерево $\mathcal{D}_1$, включающее ребро e_1 веса 1 и вершины v_0 и v_1 с номерами 0 и 1. Предположим, что после выполнения $(i-1)$-го шага получено дерево $\mathcal{D}_{i-1}$, включающее рёбра $e_1, \ldots, e_{i-1}$ с весами $1, \ldots, i-1$ и вершины $v_0, v_1, \ldots, v_{i-1}$ с номерами $0, 1, \ldots, i-1$ соответственно. Из отсутствующих в $\mathcal{D}_{i-1}$ рёбер наименьший вес i имеют рёбра, инцидентные вершине v_i с номером i. Пусть вершине v_{i-1} отвечает набор $\widetilde{\sigma} = (\sigma_1, \ldots, \sigma_n)$ и σ_j — нулевой разряд из $\widetilde{\sigma}$ с наибольшим номером j, т. е.

$\widetilde{\sigma} = (\sigma_1, \ldots, \sigma_{j-1}, 0, 1, 1, \ldots, 1)$. Вершине v_i отвечает набор $\widetilde{\sigma}'$ такой, что $|\widetilde{\sigma}'| = |\widetilde{\sigma}| + 1$; из последнего соотношения следует, что $\widetilde{\sigma}' = (\sigma_1, \ldots, \sigma_{j-1}, 1, 0, 0, \ldots, 0)$. Вершина $\widetilde{\sigma}'$ соединена в B^n ребром e_i с вершиной $\widetilde{\sigma}'' = (\sigma_1, \ldots, \sigma_{j-1}, 0, 0, 0, \ldots, 0)$, уже вошедшей в $\mathcal{D}_{i-1}$. Указанное ребро e_i в соответствии с алгоритмом и будет добавлено к $\mathcal{D}_{i-1}$, после чего будет получено дерево $\mathcal{D}_i$, включающее рёбра $e_1, \ldots, e_{i-1}, e_i$ с весами $1, \ldots, i-1, i$ и вершины $v_0, v_1, \ldots, v_{i-1}, v_i$ с номерами $0, 1, \ldots, i-1, i$.

В итоге получаем, что после $2^n - 1$ шагов будет построено минимальное остовное дерево $\mathcal{D}$, содержащее все 2^n вершин гиперкуба B^n, причём вес этого дерева равен $1 + 2 + \ldots + 2^n - 1 = (2^n - 1)2^{n-1}$.

8. а) $n \cdot 2^{n-1} - 2^n + 1$. Все вершины n-мерного куба распределим по слоям, помещая вершину $\widetilde{\sigma} = (\sigma_1, \ldots, \sigma_n)$ в слой с номером $||\widetilde{\sigma}|| = \sum_{i=1}^{n} \sigma_i$. Отметим следующее очевидное свойство гиперкуба B^n: для любой вершины v из i-го яруса, $i \in \{1, 2, \ldots, n-1\}$, существуют ребро e, соединяющее v с некоторой вершиной v' из $(i-1)$-го яруса, и ребро e', соединяющее v с некоторой вершиной v'' из $(i+1)$-го яруса. Используя это свойство и алгоритм ближайшего соседа, построение минимального остовного дерева опишем (индуктивно) так.

На первом шаге возьмём вершину $\widetilde{0} = (0, \ldots, 0)$ и инцидентные ей n рёбер; каждое из этих рёбер, очевидно, имеет наименьший — нулевой — вес и соединяет вершину $\widetilde{0}$ с некоторой вершиной $\widetilde{\sigma}$ из 1-го яруса ($\widetilde{\sigma}$ содержит ровно одну единицу). Получим некоторое дерево $\mathcal{D}_1$, содержащее все $n+1$ вершин из нулевого и первого ярусов и n рёбер из 1-го яруса. Среди рёбер, инцидентных вершинам из 2-го, ..., n-го ярусов, наименьшим весом, равным единице, обладают рёбра из 2-го яруса, соединяющие вершины из 2-го яруса с вершинами из 1-го яруса. На втором шаге добавим к дереву $\mathcal{D}_1$ очередные C_n^2 рёбер из 2-го яруса и получим дерево $\mathcal{D}_2$. После добавления к $\mathcal{D}_1$ ребра, соединяющего очередную новую вершину из 2-го яруса с некоторой вершиной из 1-го яруса, получается опять-таки дерево.

На i-м шаге к дереву $\mathcal{D}_{i-1}$ добавляются C_n^i рёбер и C_n^i вершин из i-го яруса. Эти рёбра обладают наименьшим весом, равным $i-1$, среди всех рёбер, инцидентных вершинам из i-го, ..., n-го ярусов (ещё не вошедших в дерево), а это означает, что именно их надо добавлять к $\mathcal{D}_{i-1}$ в соответствии с алгоритмом ближайшего соседа. После n-го шага в дерево $\mathcal{D}_n$ попадут все

вершины куба B^n и $\mathcal{D}_n$ окажется минимальным остовным деревом. Вес этого дерева равен

$$0 \cdot C_n^1 + 1 \cdot C_n^2 + 2 \cdot C_n^3 + \ldots + (i-1) \cdot C_n^i + \ldots + (n-1) \cdot C_n^n = \\ = \sum_{i=0}^{n} iC_n^i - \sum_{j=0}^{n} C_n^j + C_n^0 = n2^{n-1} - 2^n + 1$$

(последнее равенство в приведённой цепочке равенств следует из соотношения $\sum_{i=0}^{n} iC_n^i = n \cdot 2^{n-1}$, полученного в задаче 2.г) к главе I).

б) $n \cdot 2^{n-1}$. *Указание*. См. задачу а).

Литература

1. *Алексеев В. Б., Ложкин С. А.* Элементы теории графов и схем. — М.: МГУ, 1991.

2. *Ахо А., Хопкрофт Дж., Ульман Дж.* Построение и анализ вычислительных алгоритмов. — М.: Мир, 1979.

3. *Берж К.* Теория графов и её применения. — М.: ИЛ, 1962.

4. *Гаврилов Г. П., Сапоженко А. А.* Задачи и упражнения по дискретной математике. — 3-е изд., перераб. — М.: ФИЗМАТЛИТ, 2006.

5. *Гэри М., Джонсон Д.* Вычислительные машины и трудноразрешаемые задачи. — М.: Мир, 1982.

6. Дискретная математика и математические вопросы кибернетики. Т. 1. / Под ред. *С. В. Яблонского и О. Б. Лупанова.* — М.: Наука, 1974.

7. *Емеличев В. А., Мельников О. И. Сарванов В. А. Тышкевич Р. И.* Лекции по теории графов. — М.: Наука, 1990.

8. Комбинаторный анализ. Задачи и упражнения / Под ред. *К. А. Рыбникова.* — М.: Наука, 1982.

9. *Лупанов О. Б.* Лекции по математической логике (в двух частях). — М.: МГУ, 1970.

10. *Лупанов О. Б.* Асимптотические оценки сложности управляющих систем. — М.: МГУ, 1984.

11. *Мендельсон Э.* Введение в математическую логику. — М.: Наука, 1971.

12. *Нигматуллин Р. Г.* Сложность булевых функций. — М.: Наука, 1991.

13. *Новиков П. С.* Элементы математической логики. — М.: Наука, 1973.

14. *Оре О.* Теория графов. — М.: Наука, 1980.

15. *Пападимитриу Х., Стайглиц К.* Комбинаторная оптимизация. Алгоритмы и сложность. — М.: Мир, 1985.

16. *Пензов Ю. Е.* Элементы математической логики и теории множеств. Саратов: Изд-во Саратовского университета, 1968.

17. *Редькин Н. П.* Надежность и диагностика схем. — М.: МГУ, 1992.

18. *Уилсон Р.* Введение в теорию графов. — М.: Мир, 1977.

19. *Фролов А. Б., Андреев А. Е., Болотов А. А., Строгалов А. С.* Прикладные задачи дискретной математики в энергетике. — М.: МЭИ, 1988.

20. *Харари Ф.* Теория графов. — М.: Мир, 1973.

21. *Холл М.* Комбинаторика. — М.: Мир, 1970.

22. *Чегис И. А., Яблонский С. В.* Логические способы контроля работы электрических схем // Труды МИ АН СССР. 1958. Т. 51. С. 270–360.

23. *Шоломов Л. А.* Основы теории дискретных логических и вычислительных устройств. — М.: Наука, 1980.

24. *Яблонский С. В.* Введение в дискретную математику. — 2-е изд. — М.: Наука, 1986.

РЕДЬКИН Николай Петрович

ДИСКРЕТНАЯ МАТЕМАТИКА

Редактор *В.С. Аролович*
Оригинал-макет: *Я.В. Жабицкий*
Оформление переплета: *Н.В. Гришина*

Подписано в печать 16.03.09. Формат 60×90/16.
Бумага офсетная. Печать офсетная. Усл. печ. л. 16,5.
Уч.-изд. л. 16,5. Тираж 700 экз. Заказ №

Издательская фирма «Физико-математическая литература»
МАИК «Наука/Интерпериодика»
117997, Москва, ул. Профсоюзная, 90
E-mail: fizmat@maik.ru, fmlsale@maik.ru;
http://www.fml.ru

Отпечатано в ООО «Чебоксарская типография № 1»
428019, г. Чебоксары, пр. И. Яковлева, 15

ISBN 978-5-9221-1093-8