Кибернетический сборник

НОВАЯ СЕРИЯ

ВЫПУСК

14

Сборник переводов

Под редакцией

О. Б. ЛУПАНОВА

ИЗДАТЕЛЬСТВО «МИР»
МОСКВА 1977

УДК 519.95

Научный совет по кибернетике
Академии наук СССР

Продолжение новой серии кибернетических сборников, публикация которых была начата издательством «Мир» в 1965 г. В данном выпуске содержатся обзорные статьи и оригинальные работы известных зарубежных ученых по наиболее актуальным проблемам теоретической кибернетики. Большой интерес представляют работа Ю. Куликовского по распознаванию образов, работа Д. Гриса и Р. Констейбла о классах схем программ, а также статья У. Татта по теории графов.

Книга предназначена для научных работников, инженеров-исследователей, аспирантов и студентов, занимающихся и интересующихся теоретической кибернетикой и ее приложениями.

Редакция литературы по математическим наукам

К $\frac{20205-033}{041(01)-77}$ 33—77

Графы на непомеченных вершинах с большим числом ребер [1])

Е. М. Райт

1. ВВЕДЕНИЕ

В работе рассматриваются графы простейшего вида, т. е. (n, q)-графы, которые состоят из n вершин и q неупорядоченных пар вершин, называемых ребрами. Петли или кратные ребра не допускаются. Число различных (n, q)-графов на помеченных вершинах обозначается через $F = F(n, q)$, если же вершины не помечены, то через $T = T(n, q)$. Существует самое большее $N = n(n-1)/2$ возможных ребер, и поэтому $F = B(N, q)$, где $B(h, k) = h!/\{k!\,(h-k)!\}$ — обычный биномиальный коэффициент. Наша задача заключается в нахождении асимптотики T для больших значений n и q. Из обозначений $N = n(n-1)/2$ ясно, что наибольшее значение q равно N.

В дальнейшем под символами A, C и η будут подразумеваться числа, но не всегда одни и те же. Числа A и C положительны и не зависят от n и q. Символ A используется для обозначения любого положительного числа, которое можно выбрать, в то время как C — подходящее положительное число, которое может зависеть от данного или полученного значения A, присутствующего в изучаемом выражении или подразумевающегося. Из чего, если не оговорено обратное, во всех наших утверждениях предполагается, что n и q — достаточно большие, т. е. что $q > C$ и $n > C$. Обозначение $O(\ )$ приводится для мажорирования величин, зависящих от n и q при стремлении к бесконечности, а под мажорирующей константой понимается величина C. Величина η — произвольное число, равное $O(q^{-C})$ для некоторого C.

Ясно, что $F(n, q) = F(n, N-q)$ и аналогично $T(n, q) = T(n, N-q)$. Следовательно, далее можно считать, что

$$q \leqslant \tfrac{1}{2}N. \tag{1.1}$$

---

[1]) Wright E. M., Graphs on unlabelled nodes with a large number of edges, *Proc. London Math. Soc.* (3), **28** (1974), 577—594. Это исследование частично финансировалось правительством США.

Обозначим

$$\Lambda_p = \Lambda(p, q) = B\left\{\tfrac{1}{2}p(p-1), q\right\}/p!,$$

$$\mu = (2q/n) - \log n, \qquad J = J(v) = \left\{\tfrac{1}{2}(1 + \log v)/v\right\}^{1/2},$$

$$\delta = \tfrac{1}{2}\mu/J(n), \qquad K(v) = \pi^{1/2}e^{-1}/J(v),$$

$$\operatorname{Erf} x = 2\pi^{-1/2}\int_0^x e^{-t^2}\,dt, \quad \lambda(x) = \tfrac{1}{2}(1 + \operatorname{Erf} x).$$

Таблица значений $\operatorname{Erf} x$ приведена в работе [2]. Обозначим через v положительное число, такое, что $v \log v = 2q$, а $V = [v]$ — наибольшее целое, не превосходящее v. Пойа [9] доказал, что если выполняется условие $\frac{1}{2}N - q < An$, то $T \sim \Lambda_n$ при $n, q \to \infty$, а Обершельп [8] ослабил это условие до $\frac{1}{2}N - q < Cn^{3/2}$. В работе [12] автором доказана

Теорема 1. *Условие* $\mu \to +\infty$, *при* $n \to \infty$ *является необходимым и достаточным для того, чтобы* $T \sim \Lambda_n$. *Если это условие выполнено, то*

$$T = \Lambda_n\{1 + O(e^{-\mu})\}.$$

Коршунов [7] сформулировал без доказательства следующую теорему.

Теорема 2. *Если* $\mu > A$, *то*

$$T \sim \Lambda_n \exp(-e^{-\mu})/(1 - e^{-\mu}).$$

В работе [13] автором сформулированы результаты, среди которых имеется

Теорема 3. *Если* $\mu < -An^{-1/2}$, *то*

$$T \sim K(V)\,\Lambda_V.$$

Асимптотическое поведение функции T в теореме 3 не зависит от n. Это является асимптотическим аналогом того факта, что $T(n, q) = T(2q, q)$ при $n > 2q$.

Между условиями теорем 2 и 3 существует очевидный пробел. В этой работе автор заполняет его, но результаты формулируются более сложно. Для приложений интерес представляет более точная информация об асимптотическом поведении функции T, чем та, которая дается теоремами 2 и 3; для этого нужно заменить подразумевающийся множитель $1 + o(1)$ на $1 + \eta$. Теоремы 2 и 3 переформулируются тогда так.

Теорема 4. *Если* $\mu > An^{A-1/2}$, *то*

$$T = \Lambda_n \exp(-e^{-\mu})(1-e^{-\mu})^{-1}(1+\eta).$$

Теорема 5. *Если* $\mu < -An^{-1/2}$, *то*

$$T = K(V)\Lambda_V(1+\eta) = K(V+1)\Lambda_{V+1}(1+\eta).$$

Следующие две теоремы станут понятнее, если заметить, что $\lambda(x) = \frac{1}{2} + O(x)$ при малых значениях x, что

$$\lambda(-x) = \left(\tfrac{1}{2}\pi^{-1/2}e^{-x^2}/x\right)(1+O(x^{-2})) \tag{1.2}$$

при большом и положительном x и что $\lambda(x) \to 1$ при $x \to +\infty$. В частности,

$$\lambda(-x) = 1+\eta, \tag{1.3}$$

если $x < -A\sqrt{\log q}$.

Теорема 6. *Если* $\mu < An^{-1/2}(\log n)^{1/2}$, *так что* $\delta < A$, *то*

$$T = \lambda(-\delta)K(V)\Lambda_V(1+\eta) = \lambda(-\delta)K(V+1)\Lambda_{V+1}(1+\eta).$$

Теорема 7. *Если* $0 \leqslant \mu \leqslant n^{-1/4}$, *то*

$$T = K(n)e^{\delta^2}\lambda(-\delta)\Lambda_n(1+\eta).$$

Теперь интервалы значений q в предположениях наших теорем перекрываются. Из этих и из некоторых последующих теорем нашей работы выведем следующие две теоремы, которые окажутся полезными в приложениях, особенно для улучшения результата работы [14] относительно связности непомеченного графа (см. [18]). Обозначим

$$R(n, q) = T(n, q+1)/T(n, q).$$

Теорема 8. *Если* $\mu \to +\infty$, *то*

$$R(n, q) = \{(N-q)/(q+1)\}\{1+O(e^{-\mu})\}.$$

Если $\frac{1}{2}n\log n \leqslant q \leqslant An\log n$, *то*

$$R(n, q) = (n^2/2q)(1+\eta).$$

Если $q \leqslant \frac{1}{2}n\log n$, *то*

$$R(n, q) = (v^2/2q)(1+\eta).$$

Теорема 9. *Если* $n > C$ *и* $1 \leqslant q \leqslant \frac{1}{2}N - 1$, *то*

$$T(n, q+1) > T(n, q).$$

Только в одной теореме 9 не требуется выполнения условия $q > C$, которое во всех остальных случаях подразумевается. Можно ожидать, что результат теоремы 9 справедлив при $n \geqslant 6$, но автор этого доказать не может. Это утверждение неверно при $n = 5$, когда $N = 10$ и $T(5, 4) = T(5, 5) = 6$, а из таблиц работы [10] следует справедливость утверждения при $6 \leqslant n \leqslant 18$.

Эти теоремы были сформулированы без доказательства в [17].

Коршунов [7] распространяет свой результат (теорема 2 выше) с соответствующими модификациями на другие классы графов (например, на рассмотренные в работах [6] и [11] весемь различных классов). Все результаты автора этой работы могут быть обобщены аналогичным образом, но для простоты изложение ограничивается одним классом графов.

2. ПЕРЕЧИСЛИТЕЛЬНАЯ ТЕОРЕМА БЁРНСАЙДА — ПОЙА

Бёрнсайд и Пойа разработали принципиальный способ перечисления непомеченных графов [1, 9]. Пусть π — произвольная перестановка n помеченных вершин, и пусть $F_\pi = F_\pi(n, q)$ — число помеченных (n, q)-графов, которые остаются неизменными (инвариантными) при перестановке π его вершин и соответствующей перестановке ребер, порожденной π. Единичную перестановку будем обозначать символом I. Тогда имеем

$$n!\,T = \sum F_\pi, \tag{2.1}$$

где суммирование ведется по всем $n!$ возможным перестановкам π, т. е. по всем членам симметрической группы степени n.

Перестановка π может быть представлена в виде произведения непересекающихся циклов, причем единственным образом. (Здесь и далее термин *цикл* применяется как понятие теории групп подстановок, а не как понятие теории графов.) Предположим, что в этом произведении имеется ровно p_j циклов длины j, а в соответствующем произведении, представляющем порожденную перестановку ребер, — P_j циклов длины j.

Тогда имеем

$$\sum_{j=1}^{n} j p_j = n \tag{2.2}$$

и

$$\sum_{j=1}^{N} j P_j = N. \tag{2.3}$$

Также имеют место равенства (см. [8])

$$P_1 = \frac{1}{2} p_1 (p_1 - 1) + p_2, \tag{2.4}$$

$$P_2 = p_1 p_2 + p_2 (p_2 - 1) + p_4, \quad P_3 = p_1 p_3 + \frac{1}{2} p_3 (3p_3 - 1) + p_6. \tag{2.5}$$

Назовем множество j вершин, инвариантное при действии некоторого цикла из π длины j, *вершинным j-множеством*. Аналогично, множество из j ребер, инвариантное при действии некоторого цикла длины j из порожденной перестановки ребер, называется *реберным j-множеством*. Любой граф, инвариантный при действии перестановки π, должен содержать либо все ребра произвольного реберного j-множества, либо не содержать ни одного ребра этого j-множества. Существует ровно P_j реберных j-множеств, и каждые s_j из них, $s_j \leqslant P_j$, могут появиться в инвариантном (n, q)-графе, причем

$$\sum_{j=1}^{N} j s_j = q. \tag{2.6}$$

Можно выбрать s_j реберных j-множеств $B(P_j, s_j)$ различными способами, и поэтому

$$F_\pi = \sum_{(q)} D(s_1, \ldots, s_n) = \left[\prod_{j=1}^{N} (1 + X^j)^{P_j} \right]_q, \tag{2.7}$$

где $\sum_{(q)}$ означает суммирование по всем $s_1, \ldots, s_n$, удовлетворяющим (2.6),

$$D(s_1, \ldots, s_n) = \prod_{j=1}^{n} B(P_j, s_j),$$

а $[Q(X)]_q$ означает коэффициент при X^q в многочлене $Q(X)$. Далее, мы видим, что F_π зависит только от чисел $P_1, \ldots, P_N$, а следовательно, только от чисел $p_1, \ldots, p_n$.

Существует ровно

$$n! \prod_{j=1}^{n} (p_j!\, j^{p_j})^{-1}$$

перестановок π с одной и той же цикловой структурой $(p_1, p_2, \ldots, p_n)$, а значение F_π для каждой такой перестановки π постоянно. Следовательно,

$$T = \sum_{((n))} (p_j!\, j^{p_j})^{-1} \sum_{(q)} D(s_1, \ldots, s_n),$$

где $\sum_{((n))}$ означает суммирование по всем $p_1, \ldots, p_n$, удовлетворяющим (2.2).

Обозначим

$$S_p = (n!)^{-1} \sum_1 F_\pi,$$

где $\sum_1$ означает суммирование по всем π, для которых $p_1 = p$. Тогда

$$T = T_1 + T_2,$$

где

$$T_1 = \sum_{p=M+1}^{n} S_p, \quad T_2 = \sum_{p=0}^{M} S_p$$

и

$$M = [(q^4 n^2 \log n)^{1/5}].$$

Обозначим через $H_k(n)$ число тех π, цикловая структура которых не содержит циклов длины, меньшей чем $k+1$, и пусть $h_k(n) = H_k(n)/n!$. В частности, $H_1(n)$ — число эйлеровых дуэлей [3, 4]. Ясно, что

$$H_k(n) = n! \sum \prod_{j=k+1}^{n} \{p_j!\, j^{p_j}\}^{-1},$$

где суммирование ведется по всем множествам $p_{k+1}, \ldots, p_n$, таким, что

$$\sum_{j=k+1}^{n} j p_j = n.$$

Итак,

$$F_\pi \geqslant D(q, 0, 0, \ldots, 0) = B(P_1, q) \geqslant B\left(\tfrac{1}{2} p_1 (p_1 - 1), q\right),$$

и существует ровно $B(n, p) H_1(n-p)$ различных перестановок π, таких, что $p_1 = p$. Следовательно,

$$S_p \geqslant (n!)^{-1} B(n, p) H_1(n-p) B\left(\tfrac{1}{2} p(p-1), q\right) = h_1(n-p) \Lambda_p, \quad (2.8)$$

и поэтому

$$T_1 \geqslant \sum_{p=M+1}^{n} h_1(n-p) \Lambda_p. \quad (2.9)$$

Позже мы увидим, что при подходящих условиях можно заменить знак $\geqslant$ в (2.8) и (2.9) на знак $\sim$, но доказательство становится более сложным. Найдем сначала верхнюю границу для T_2, которая, как будет показано ниже, мала по сравнению с T_1.

3. ВЕРХНЯЯ ГРАНИЦА ДЛЯ T_2

Лемма 1. *Если* $An \leqslant q < An \log n$, *то*

$$T_2 \leqslant n^{\frac{1}{2} q (1+A)}.$$

Напомним читателю, что, согласно нашему определению, три символа A в утверждении леммы не обязательно означают одно и то же число. Возьмем такое $p_1 \leqslant M$, что $P_1 < CM^2$ в силу соотношения (2.4). Для любого $X > 0$, в силу соотношения (2.7), имеем

$$F_\pi \leqslant X^{-q} \prod_{j=1}^{N} (1 + X^j)^{P_j}.$$

Далее, если $j \geqslant 2$, получаем неравенство $(1 + X^j)^2 \leqslant (1 + X^2)^j$ и поэтому, согласно (2.3),

$$\prod_{j=2}^{N} (1 + X^j)^{2P_j} \leqslant (1 + X^2)^{(N - P_1)}.$$

Следовательно, $F_\pi \leqslant Y_1 \cdot Y_2$, где

$$Y_1 = X^{-q} (1 + X^2)^{\frac{1}{2} N}, \quad Y_2 = \{(1 + X)^2 / (1 + X^2)\}^{\frac{1}{2} P_1}.$$

Если положить $X = \sqrt{\{q/(N - q)\}}$, то получим

$$\log Y_1 = \tfrac{1}{2} q \log (N/q) + \tfrac{1}{2} (N - q) \log \{N/(N - q)\}.$$

Теперь

$$\tfrac{1}{2} q \log (N/q) \leqslant \tfrac{1}{2} q \log n + O(q),$$

$$\tfrac{1}{2} (N - q) \log \{N/(N - q)\} = \tfrac{1}{2} (N - q) \log (1 + \{q/(N - q)\}) \leqslant \tfrac{1}{2} q,$$

так что

$$\log Y_1 \leqslant \tfrac{1}{2} q \log n + O(q).$$

Снова

$$(1 + X)^2 / (1 + X^2) = 1 + O(q^{1/2}/n);$$

значит,

$$\log Y_2 \leqslant CP_1 q^{1/2}/n < Cq (\log n)^{3/4},$$

поскольку $P_1 < CM^2$ и $q < An \log n$. Отсюда

$$\log F_\pi \leqslant \tfrac{1}{2} q \log n + Cq (\log n)^{3/4} \leqslant \tfrac{1}{2} q (1 + A) \log n,$$

и наконец

$$T_2 = \sum_{p=0}^{M} s_p = (n!)^{-1} \sum F_\pi \leqslant n^{\frac{1}{2} q (1 + A)},$$

так как во второй сумме имеется меньше чем $n!$ слагаемых, для которых перестановка π такова, что $p_1 \leqslant M$.

4. СВОЙСТВА $h_k(n)$

Известно [3, 4], что

$$h_1(n) = 1 - 1 + \frac{1}{2!} - \frac{1}{3!} + \ldots + \frac{(-1)^n}{n!}. \tag{4.1}$$

и поэтому

$$|h_1(n) - e^{-1}| < 1/(n+1)!. \tag{4.2}$$

Выполняется также соотношение

$$H_k(n) = (n-1)H_k(n-1) + (n-1)\ldots(n-k)H_k(n-k-1).$$

(Последнее получаем, просматривая, в скольких перестановках из $H_k(n)$ некоторый элемент появляется в цикле длины, большей чем $k+1$, и в скольких перестановках тот же самый элемент в цикле длины $k+1$.) Отсюда получаем, что

$$nh_k(n) = (n-1)h_k(n-1) + h_k(n-k-1). \tag{4.3}$$

В работе [15] изучены арифметические свойства $H_1(n)$, а в работе [16] исследованы асимптотика и арифметические свойства $H_k(n)$ для любого k. Однако все, что необходимо для целей нашей работы, дает

Лемма 2. *Если* $n > k$, *то*

$$C < h_k(n) \leqslant 1.$$

Правая часть неравенства тривиальна. Ясно, что $H_k(n) > 0$ для $n > k$ и поэтому левая часть неравенства справедлива для $k < n \leqslant 2k+1$. Для всех $n > k$ утверждение леммы доказывается из (4.3) по индукции.

5. ФУНДАМЕНТАЛЬНАЯ ЛЕММА

Докажем теперь следующую фундаментальную лемму.

Лемма 3. *Если* $An < q < An\log n$ *и* $M < p \leqslant n-2$, *то*

$$S_p = \Lambda_p h_1(n-p)\{1 + O(\varepsilon)\},$$

где

$$\varepsilon_1 = nq^2p^{-4}, \quad \varepsilon_2 = qp^{-2}, \quad \varepsilon = \max(\varepsilon_1, \varepsilon_2) = o(1).$$

Рассмотрим те значения π, для которых $p_1 = p > M$. Имеем

$$F_\pi = \sum_{(q)} D(s_1, \ldots, s_n) = \sum_{j=1}^{N} F^{(j)},$$

где слагаемое $F^{(j)}$ равно сумме всех таких D, что $s_j > 0$, а $s_\omega = 0$ для всех $\omega > j$. Если $j \geqslant 4$, то имеем

$$Y_3 = D(s_1, s_2, \ldots, s_j, 0, \ldots, 0)/D(s_1 + j, s_2, \ldots, s_j - 1, 0, \ldots, 0) =$$
$$= B(P_j, s_j)\, B(P_1, s_1)/\{B(P_j, s_j - 1)\, B(P_1, s_1 + j)\} =$$
$$= \{(P_j - s_j + 1)/s_j\} \prod_{k=1}^{j} \{(s_1 + k)/(P_1 - s_1 - k + 1)\} \leqslant$$
$$\leqslant Cn^2 q^j p_1^{-2j} = C\varepsilon_1^2 \varepsilon_2^{j-4}.$$

Если числитель Y_3 принимает значения всех слагаемых $F^{(j)}$, то знаменатель принимает значения членов суммы для F_π из (2.7) без повторений одного и того же члена. Отсюда

$$\sum_{j=4}^{n} F^{(j)}/F_\pi \leqslant C\varepsilon_1^2 \sum_{j=4}^{n} \varepsilon_2^{j-4} \leqslant C\varepsilon_1^2.$$

Обозначим $E_1 = F^{(1)} + F^{(2)} + F^{(3)}$. Сразу видно, что

$$F_1 = [(1 + X)^{P_1} (1 + X^2)^{P_2} (1 + X^3)^{P_3}]_q$$

и $F_\pi = E_1 \{1 + O(\varepsilon_1^2)\}$. Запишем

$$Q_1 = p_1 (p_1 - 1)/2,$$
$$Q_2 = Q_2(p_2) = p_2 (p_1 + p_2 - 1),$$
$$Q_3 = Q_3(p_3) = \tfrac{1}{2} p_3 (2p_1 + 3p_3 - 1)$$

(так из (2.5) вытекает, что $P_2 = Q_2 + p_4$, $P_3 = Q_3 + p_6$) и

$$E_2(p_1, p_2, p_3) = [(1 + X)^{P_1} (1 + X^2)^{Q_2} (1 + X^3)^{Q_3}]_q.$$

Находим, что

$$E_1 - E_2(p_1, p_2, p_3) < C\varepsilon E_1$$

с помощью тех же соображений, что и выше для слагаемых $F^{(j)}$, поскольку в случае $1 \leqslant r \leqslant p_4 = P_2 - Q_2$ получаем

$$\left(\frac{P_2 - Q_2 - r + 1}{r}\right)\left(\frac{s_1 + 1}{P_1 - s_1}\right)\left(\frac{s_1 + 2}{P_1 - s_1 - 1}\right) < \frac{Cnq^2}{p^4} = C\varepsilon_1,$$

и аналогично для P_3, Q_3, $\varepsilon_1\varepsilon_2$. Из этого следует, что

$$F_\pi = E_2(p_1, p_2, p_3)\{1 + O(\varepsilon)\}.$$

Отсюда

$$\sum_1 F_\pi = \sum_1 E_2(p, p_2, p_3)\{1 + O(\varepsilon)\}.$$

Поскольку существует ровно

$$\frac{n!\,h_3(n-p-2p_2-3p_3)}{p!\,2^{p_2}\cdot p_2!\cdot 3^{p_3}\cdot p!}$$

различных перестановок π с $p_1=p$ и заданными p_2, p_3, то

$$\sum_1 F_\pi = \frac{n!}{p!}\sum_{p_2=0}^{[(n-p)/2]}\frac{Y_4}{2^{p_2}(p_2!)}\{1+O(\varepsilon)\},$$

где $m=n-p-2p_2$ и

$$Y_4=\sum_{p_3=0}^{[m/3]}h_3(m-3p_3)\,E_2(p,\,p_2,\,p_3)/(p_3!\,3^{p_3}).$$

Теперь, если $p_3\geqslant 1$, имеем

$$E_2(p_1,\,p_2,\,p_3)=\left[(1+X)^{P_1}(1+X^2)^{Q_2}(1+X^3)^{Q_3(p_3)}\right]_q=$$
$$=\left[(1+X)^{P_1}(1+X^2)^{Q_2}(1+X^3)^{Q_3(p_3-1)+r}\right]_q,$$

где

$$r=Q_3(p_3)-Q_3(p_3-1)=p_1+3p_3-2<n.$$

Если $1\leqslant s\leqslant r$, получаем

$$\frac{r-s+1}{s}\cdot\frac{(s_1+1)(s_1+2)(s_1+3)}{(P_1-s_1)(P_1-s_1-1)(P_1-s_1-2)}\leqslant\frac{Cnq^3}{p^6}\leqslant C\varepsilon^2.$$

Откуда, как и выше,

$$0\leqslant E_2(p_1,\,p_2,\,p_3)-E_2(p_1,\,p_2,\,p_3-1)\leqslant C\varepsilon E_2(p_1,\,p_2,\,p_3),$$

$$E_2(p_1,\,p_2,\,p_3-1)\leqslant E_2(p_1,\,p_2,\,p_3)\leqslant(1+C\varepsilon)\,E_2(p_1,\,p_2,\,p_3-1),$$

и, следовательно,

$$E_2(p_1,\,p_2,\,0)\leqslant E_2(p_1,\,p_2,\,p_3)\leqslant(1+C\varepsilon)^{p_3}E_2(p_1,\,p_2,\,0).$$

Теперь

$$(1+C\varepsilon)^{p_3}-1\leqslant Cp_3\varepsilon(1+C\varepsilon)^{p_3-1},$$

и поэтому

$$Y_4=E_2(p_1,\,p_2,\,0)\left(h_3(m)+\sum_{p_3=1}^{[m/3]}\{h_3(m-3p_3)/(p_3!\,3^{p_3})\}+Y_5\right),$$

где $Y_5=0$, если $m<3$, а в остальных случаях

$$Y_5\leqslant C\varepsilon\sum_{p_3=1}^{[m/3]}\{h_3(m-3p_3)(1+C\varepsilon)^{p_3-1}/3^{p_3}(p_3-1)!\}\leqslant C\varepsilon e^{C(1+\varepsilon)}<C\varepsilon.$$

Снова

$$h_2(m)=h_3(m)+\sum_{p_3=1}^{[m/3]}h_3(m-3p_3)/(p_3!\,3^{p_3}),$$

таким образом,

$$Y_4 = E_2(p_1,\ p_2,\ 0)\, h_2(m)\{1 + O(\varepsilon)\}$$

и

$$\sum_1 F_\pi = \frac{n!}{p!} \sum_{p_2=0}^{[(n-p)/2]} \frac{h_2(n-p-2p_2)\, E_3(p,\ p_2)}{2^{p_2} p_2!}(1 + O(\varepsilon)),$$

где

$$E_3(p,\ p_2) = \left[(1+X)^{Q_1+p_2}(1+X^2)^{Q_2(p_2)}\right]_q.$$

Теперь, если $p_2 \geqslant 1$, то

$$Q_2(p_2) - Q_2(p_2-1) = p_1 + 2p_2 - 2$$

и

$$E_3(p_1,\ p_2) = \left[(1+X)^{Q_1+p_2}(1+X^2)^{Q_2(p_2-1)}(1+X^2)^{p_1+2p_2-2}\right]_q.$$

Снова если $s \geqslant 1$, то

$$\frac{p_1+2p_2-1-s}{s} \cdot \frac{s_1+1}{Q_1+p_2-s_1} \cdot \frac{s_1+2}{Q_1+p_2-s_1-1} \leqslant$$

$$\leqslant \frac{nq^2}{(Q_1-q)^2} \leqslant Cnq^2p^{-4} \leqslant C\varepsilon.$$

Следовательно,

$$E_3(p_1,\ p_2) = \left[(1+X)^{Q_1+p_2}(1+X^2)^{Q_2(p_2-1)}\right]_q \{1 + O(\varepsilon)\} =$$
$$= \{E_3(p_1,\ p_2-1) + Y_6\}\{1 + O(\varepsilon)\},$$

где

$$Y_6 = \left[(1+X)^{Q_1+p_2-1}(1+X^2)^{Q_2(p_2-1)}\right]_{q-1} \leqslant$$
$$\leqslant Cq(Q_1-q)^{-1} E_3(p_1,\ p_2-1) \leqslant C\varepsilon E_3(p_1,\ p_2-1).$$

Отсюда

$$E_3(p_1,\ p_2) = E_3(p_1,\ p_2-1)(1+C\varepsilon) = E_3(p_1,\ 0)(1+C\varepsilon)^{p_2},$$

и поэтому

$$S_p = \frac{1}{p!} E_3(p,\ 0)\left\{\sum_{p_2=0}^{[(n-p)/2]} \frac{h_2(n-p-2p_2)}{p_2!\, 2^{p_2}} + Y_7\right\}\{1 + O(\varepsilon)\} =$$
$$= (p!)^{-1} E_3(p,\ 0)(h_1(n-p) + Y_7)(1 + O(\varepsilon)),$$

где

$$Y_7 \leqslant \sum_{p_2=1}^{[(n-p)/2]} h_2(n-p-2p_2)\{(1+C\varepsilon)^{p_2} - 1\}/(p_2!\, 2^{p_2}) <$$
$$< C\varepsilon \sum_{p_2=1}^{[(n-p)/2]} (1+C\varepsilon)^{p_2-1}/(p_2-1)! < C\varepsilon.$$

Таким образом,

$$E_3(p_1,\ 0) = \left[(1+X)^{Q_1}\right]_q = B(Q_1,\ q) = p!\, \Lambda_p.$$

Этим завершается доказательство леммы 3.

6. ДВЕ ПОСЛЕДУЮЩИЕ ЛЕММЫ

Лемма 4. *Если* $An < q < An\log n$, *то*

$$S_p = \Lambda_p h_1 (n-p)(1+\eta) \quad (n^{7/8} \leqslant p \leqslant n)$$

и

$$S_p \leqslant C\Lambda_p \qquad (M \leqslant p \leqslant n).$$

Если $n^{7/8} < p \leqslant n$, то из леммы 3 получаем $\varepsilon = \eta$.

Если $M \leqslant p \leqslant n$, то имеем $\varepsilon = o(1)$. Снова $h_1(0) = 1$, $h_1(1) = 0$, $S_{n-1} = 0$.

Лемма 5. *Если* $An < q < An\log n$, *то*

$$\log \Lambda_p = \omega - \varphi + \eta \quad (n^{7/8} < p \leqslant n)$$

и

$$\log \Lambda_p \leqslant \omega - \varphi + \eta \quad (M \leqslant p \leqslant n),$$

где

$$\omega = \omega(p) = 2q\log p - p\log p + p - q\log(2q) + q,$$

$$\varphi = \varphi(p) = \tfrac{1}{2}\log(4\pi^2 pq) + (q/p) + (q^2/p^2).$$

Предположим, что условия леммы выполнены. Имеем

$$p!\,q!\,\Lambda_p = P(P-1)\ldots(P-q+1),$$

где $P = p(p-1)/2$ и поэтому

$$\log(p!\,q!\,\Lambda_p) = q\log\left(\tfrac{1}{2}p^2\right) + \sum_{s=0}^{q-1}\log\{1 - p^{-2}(p+2s)\} =$$

$$= 2q\log p - q\log 2 - \frac{q}{p} - \frac{q^2}{p^2} - Y_8,$$

где

$$0 < Y_8 < Cq^3p^4.$$

Теперь по формуле Стирлинга

$$\log(p!) = \left(p + \tfrac{1}{2}\right)\log p - p + \tfrac{1}{2}\log(2\pi) + O\left(\tfrac{1}{p}\right),$$

и аналогично для $\log(q!)$, откуда вытекает справедливость нашей леммы.

Если в качестве p взять непрерывную переменную, то имеем

$$\omega'(p) = (2q/p) - \log p, \quad \omega''(p) = -(2q+p)/p^2, \tag{6.1}$$

$$\omega'''(p) = (4q+p)/p^3, \qquad \varphi'(p) = -(4q^2 + 2pq - p^2)/p^3. \tag{6.2}$$

7. СЛУЧАЙ $0 \leqslant \mu \leqslant A \log n$

Предположим теперь, что

$$0 \leqslant \mu < A \log n,$$

так что

$$\tfrac{1}{2} n \log n \leqslant q \leqslant Cn \log n.$$

Из леммы 5 имеем

$$\log \Lambda_n = q \log (n^2/q) + O(q) \geqslant q \log q - Cq \log \log q,$$

и поэтому по лемме 1

$$T_2 = O(\Lambda_n e^{-Cq \log q}) = \eta \Lambda_n. \tag{7.1}$$

Обозначим $\xi = \frac{1}{2}(2q + n)/n^2$ и $f = [n^{7/12}]$. По (6.1) и (6.2) $\omega'(n) = \mu$, $\omega''(n) = -2\xi$ и

$$0 < \omega'''(p) < Cq/p^3.$$

Отсюда, если $t = n - p$, получаем

$$\omega(p) = \omega(n) - \mu t - \xi t^2 - t^3 \omega'''(p')/6$$

для некоторого p', такого, что $p \leqslant p' \leqslant n$, и поэтому

$$\omega(p) \leqslant \omega(n) - \mu t - \xi t^2 \quad (M < p \leqslant n),$$

а также

$$\omega(p) = \omega(n) - \mu t - \xi t^2 + \eta \quad (0 \leqslant t \leqslant f).$$

Снова

$$-Cq^2/p^3 < \varphi'(p) < 0 \quad (M \leqslant p \leqslant n),$$

откуда по лемме 5 имеем

$$\log (\Lambda_p/\Lambda_n) \leqslant -\mu t - \xi t^2 + \eta \quad (M < p \leqslant n)$$

и

$$\log (\Lambda_p/\Lambda_n) = -\mu t - \xi t^2 + \eta \quad (0 \leqslant t \leqslant f).$$

Теперь ограничимся областью

$$An^{A_1 - 1/2} \leqslant \mu \leqslant A \log n, \tag{7.2}$$

где $A_1 < \frac{1}{2}$, и обозначим

$$g = 2 + [(\log n)/\mu] \leqslant C + Cn^{1/2 - A_1} \log n,$$

а

$$Y_9 = \sum_{t=0}^{\infty} e^{-\mu t} h_1(t).$$

Воспользовавшись (4.1) и изменяя порядок суммирования, находим, что

$$Y_9 = \exp(-e^{-\mu})/(1 - e^{-\mu}).$$

Поскольку $\xi = O(n^{-1}\log n)$, то

$$\log(\Lambda_p/\Lambda_n) = -\mu t + \eta \quad (0 \leqslant t \leqslant g)$$

и, естественно,

$$\log(\Lambda_p/\Lambda_n) \leqslant -\mu t + \eta \quad (M < p \leqslant n).$$

Следовательно, по лемме 4

$$(T_1/\Lambda_n) = Y_9 + Y_{10} + Y_{11} - Y_{12}, \tag{7.3}$$

где

$$Y_{10} = \eta \sum_{t=0}^{g} e^{-\mu t} h_1(t) = \eta Y_9,$$

$$0 \leqslant Y_{11} \leqslant C \sum_{t=g+1}^{n-M} e^{-\mu t+\eta} h_1(t) \leqslant C e^{-g\mu}(1-e^{-\mu})^{-1} = \eta Y_9,$$

$$0 \leqslant Y_{12} = \sum_{t=g+1}^{\infty} e^{-\mu t} h_1(t) \leqslant e^{-g\mu}(1-e^{-\mu})^{-1} = \eta Y_9.$$

Отсюда, в силу (7.1) и (7.3),

$$T = \Lambda_n Y_9 (1+\eta).$$

В этом и заключается теорема 4, когда выполняется условие (7.2). Если же $\mu > 3\log n$, то теорема 4 вытекает из теоремы 1.

Рассмотрим теперь область

$$0 \leqslant \mu \leqslant An^{-1/4}.$$

Для краткости далее будем писать $J = J(n)$. Имеем

$$\xi = \tfrac{1}{2}(2q+n)/n^2 = J^2 + \left(\tfrac{1}{2}\mu/n\right).$$

Отсюда

$$\log(\Lambda_p/\Lambda_n) = -\mu t - J^2t^2 + \eta \quad (0 \leqslant t \leqslant f)$$

и

$$\log(\Lambda_p/\Lambda_n) \leqslant -\mu t - J^2t^2 + \eta \quad (M < p \leqslant n).$$

В дальнейшем будем писать

$$Y_{13} = \sum_{t=0}^{f} e^{-\mu t - J^2t^2}, \quad Y_{14} = \sum_{t=f+1}^{\infty} e^{-\mu t - J^2t^2}.$$

По лемме 4 получаем

$$T_1/\Lambda_n = \sum_{t=0}^{f} e^{-\mu t - J^2t^2} h_1(t) + \eta Y_{13} + O(Y_{14}).$$

Из соотношения (4.2) вытекает $h_1(t) = e^{-1} + \eta$ при $t > \log n$. Следовательно,

$$\sum_{t=0}^{f} e^{-\mu t - J^2 t^2} h_1(t) = e^{-1} Y_{13}(1+\eta) + O(\log n).$$

Легко видеть, что $\log n = \eta Y_{13}$. Снова $J^2 f^2 > Cn^{1/6}$, и поэтому

$$Y_{14} \leqslant \sum_{t=f+1}^{\infty} e^{-J^2 t^2} \leqslant \int_{f}^{\infty} e^{-J^2 u^2}\, du = J^{-1} e^{-J^2 f^2} = \eta.$$

Откуда

$$T_1/\Lambda_n = e^{-1} Y_{13}(1+\eta)$$

и далее, в силу (7.1),

$$T/\Lambda_n = T_1(1+\eta)/\Lambda_n = e^{-1} Y_{13}(1+\eta) = e^{-1} Y_{15}(1+\eta). \quad (7.4)$$

где

$$Y_{15} = Y_{13} + Y_{14} = \sum_{t=0}^{\infty} e^{-\mu t - J^2 t^2}.$$

Теперь

$$Y_{15} = \int_{0}^{\infty} e^{-\mu u - J^2 u^2}\, du + O(1),$$

и, поскольку $\delta = \frac{1}{2}\mu/J$, получаем

$$\int_{0}^{\infty} e^{-\mu u - J^2 u^2}\, du = J^{-1} e^{\delta^2} \int_{\delta}^{\infty} e^{-\omega^2}\, d\omega = \pi^{1/2} J^{-1} e^{\delta^2} \lambda(-\delta).$$

Опять

$$e^{\delta^2} \lambda(-\delta) = \frac{1}{2} + O(\delta),$$

если $0 \leqslant \delta < A$, и, в силу (1.2),

$$J^{-1} e^{\delta^2} \lambda(-\delta) = (2J\delta)^{-1} \{1 + O(\delta^2)\} \sim \mu^{-1}$$

при большом δ. Так как $J = \eta$ и $\mu = \eta$, то имеем

$$Y_{15} = \pi^{1/2} J^{-1} e^{\delta^2} \lambda(-\delta)(1+\eta).$$

Отсюда следует справедливость теоремы 7.

8. ДОКАЗАТЕЛЬСТВО ТЕОРЕМ 5 И 6

Предположим, что $\delta < A$ и $q > An$ (последнее ограничение позже будет снято). Поскольку $v \sim 2q/\{\log(2q)\}$, то выполняются неравенства

$$Cn/\log n < v < Cn.$$

Будем теперь писать $J = J(v)$. Напомним, что $\omega'(v) = 0$, $\omega''(v) = -2J$ и $\omega''(p) < 0$, так что $\omega(v)$ — наибольшее значение функции $\omega(p)$. Обозначим

$$\sigma = \omega(v) - \varphi(v), \qquad X = v^{7/12}, \qquad W = [v - X],$$

$$U = \min(n, [v + X]), \quad Z_1 = \sum_{p=M+1}^{W-1} S_p, \quad Z_2 = \sum_{p=W}^{n} S_p.$$

Рассмотрим первое значение p, для которого

$$W \leqslant p \leqslant U, \tag{8.1}$$

так что $|p - v| < CX$. Тогда

$$\omega(p) = \omega(v) - J^2(p - v)^2 + \frac{1}{2}\omega'''(p')(p - v)^3,$$

где p' лежит между v и p, так что

$$|\omega'''(p')(p - v)^3| < C(q/v^3)X^3 = \eta.$$

Аналогично

$$\varphi(p) = \varphi(v) + \varphi'(p'')(p - v) = \varphi(v) + \eta.$$

Следовательно, по лемме 5

$$\log \Lambda_p = \sigma - J^2(p - v)^2 + \eta \tag{8.2}$$

каждый раз, когда выполняется (8.1). Поскольку $J = \eta$, мы также имеем

$$\log \Lambda_V = \sigma + \eta, \quad \log \Lambda_{V+1} = \sigma + \eta. \tag{8.3}$$

Если $M < p \leqslant W - 1$, то с помощью простых вычислений показываем, что $\omega'(p) - \varphi'(p) > 0$, и поэтому по лемме 5

$$\log \Lambda_p \leqslant \omega(W - 1) - \varphi(W - 1) + \eta = \sigma - J^2X^2 + \eta \leqslant \sigma - Cv^{1/6}. \tag{8.4}$$

Если $U = v + X \leqslant p \leqslant n$, то $\omega'(p) - \varphi'(p) < 0$ и аналогично

$$\log \Lambda_p \leqslant \sigma - Cv^{1/6}. \tag{8.5}$$

По лемме 3 и в силу (8.4) получаем

$$Z_1 \leqslant C \sum_{p=M+1}^{W-1} \Lambda_p \leqslant Cn \exp(\sigma - Cv^{1/6}) \leqslant C\eta e^{\sigma}.$$

Снова $W > Cn/\log n$, и поэтому, если $p \geqslant W$, то в лемме 3 имеем $\varepsilon = \eta$ и $Z_2 = Z_3(1 + \eta)$, где

$$Z_3 = \sum_{p=W}^{n} \Lambda_p h_1(n - p).$$

Используя (4.2) в случае $p \leqslant n - [\log n]$, имеем

$$h_1(n - p) = e^{-1} + \eta,$$

в то время как для $n - \log n \leqslant p \leqslant n$ получаем

$$h_1(n-p) = e^{-1} + O(1).$$

Следовательно, $Z_3 = (e^{-1} + \eta) Z_4 + Z_5$, где

$$Z_4 = \sum_{p=W}^{n} \Lambda_p, \quad Z_5 < C \sum_{p=n-[\log n]}^{n} \Lambda_p,$$

так что, в силу (8.2), имеем

$$Z_5 < Ce^{\sigma} \log n.$$

Теперь обозначим

$$Z_6 = \sum_{p=W}^{U} \Lambda_p, \quad Z_7 = \sum_{p=U+1}^{n} \Lambda_p,$$

где $Z_7 = 0$ при $U = n$. При $U < n$, в силу (8.5), получаем

$$Z_7 \leqslant C\eta e^{\sigma}.$$

Опять, в силу (8.2), $Z_6 = e^{\sigma} Z_8 (1 + \eta)$, где $Z_8 = \sum_{p=W}^{U} e^{-J^2(p-v)^2}$. Объединив вместе полученные результаты, мы получим, что

$$Z_1 + Z_2 = e^{\sigma - 1} Z_8 (1 + \eta) + O(e^{\sigma} \log n).$$

Простыми вычислениями можно показать, что

$$Z_8 = \int_{W}^{U} e^{-J^2(t-v)^2} dt + O(1).$$

Если $U = [v + X]$, то имеем

$$Z_8 = \int_{-X}^{X} e^{-J^2u^2} du + O(1) = \int_{-\infty}^{+\infty} e^{-J^2u^2} du + O(1) = \pi^{1/2} J^{-1} (1 + \eta). \quad (8.6)$$

Если $U = n < v + X$, аналогично получается

$$\int_{W}^{U} e^{-J^2(t-v)^2} dt = \pi^{1/2} J^{-1} \lambda(-d) + O(1), \quad (8.7)$$

где $d = (v - u) J$. В этом случае $n \leqslant v(1 + \eta)$. Снова $\mu < A\{(\log n)/n\}^{1/2}$ и

$$v \log v = 2q = n \log n + \mu n. \quad (8.8)$$

Следовательно, $v \leqslant n(1 + \eta)$, и поэтому $v = n(1 + \eta)$. Откуда, в силу (8.8),

$$v - n = \mu n (1 + \eta)(1 + \log n)^{-1} < C(n/\log n)^{1/2}$$

и $J(v) = J(n)(1+\eta)$. Поэтому

$$d = (v-n)J(v) = \delta(1+\eta) < A.$$

Если $\delta < -A\sqrt{\log n}$, то $d < -A\sqrt{\log n}$ и, таким образом, $\lambda(-d) = 1+\eta$ в силу (1.3). Следовательно, из (8.7) вытекает (8.6). Если $\delta > -A\sqrt{\log n}$, то $|\delta| < A\sqrt{\log n}$ и $|d| < A\sqrt{\log n}$. Отсюда получаем, что $\lambda(-d) = \lambda(-\delta)(1+\eta)$, а из (8.7) следует

$$Z_8 = \pi^{1/2} J^{-1} \lambda(-\delta)(1+\eta). \tag{8.9}$$

Из равенства (1.3) ясно, что (8.6) и (8.9) эквивалентны при выполнении условия $\delta < -A\sqrt{\log n}$. Собирая вместе последние результаты и напоминая, что $K = \pi^{1/2}/(Je)$, имеем

$$T_1 = K(v) e^{\sigma} \lambda(-\delta)(1+\eta).$$

По лемме 1

$$\log T_2 \leqslant \tfrac{3}{4} q \log n.$$

Но

$$\sigma = \omega(v) - \varphi(v) = 2q \log v + O(q \log\log v) =$$
$$= 2q \log n + O(q \log\log n).$$

Откуда $T_2 = \eta T_1$ и

$$T = K(v) e^{\sigma} \lambda(-\delta)(1+\eta).$$

Из (1.3), (8.3) и некоторых тривиальных вычислений следуют утверждения теорем 5 и 6 в случае $q > An$.

Теперь возвратимся к ограничению $q > An$. Предположим, что $q < n/2$, так что $n > 2q$. Тогда $\delta < -Cn^{1/2}$ и $\lambda(-\delta) = 1+\eta$. Следовательно, этого достаточно для доказательства теоремы 5. Но утверждение теоремы 5 зависит только от q и не зависит от n. Снова если $n > 2q$, то имеем $T(n, q) = T(2q, q)$. Любой (n, q)-граф может быть получен ровно из одного $(2q, q)$-графа добавлением $n - 2q$ изолированных вершин, и обратно. Отсюда если $q < n/2$, то выберем новое n, скажем n', такое что $q > n'/3$ и $n' \to \infty$ при $q \to \infty$.

Таким образом, мы доказали теорему 5 для последних значений n и q.

9. ДОКАЗАТЕЛЬСТВО ТЕОРЕМ 8 И 9

Имеем

$$\Lambda(n, q+1)/\Lambda(n, q) = (N-q)/(q+1),$$

так что первая часть теоремы 8 следует из теоремы 1. Далее, если $q < An \log n$, то получаем

$$(N-q)/(q+1) = n'(1+\eta)/(2q).$$

Если $\mu \geqslant n^{-1/4}$, то воспользуемся теоремой 4. Если положить

$$\psi(\mu) = \exp(-e^{-\mu})(1-e^{-\mu})^{-1},$$

то достаточно доказать, что

$$\frac{d}{dq}\log\psi(\mu) = \eta.$$

Имеем

$$\frac{d\mu}{dq} = \frac{1}{n}, \quad \frac{d}{d\mu}\log\psi(\mu) = e^{-\mu} - \frac{e^{-\mu}}{1-e^{-\mu}} = -\frac{e^{-2\mu}}{1-e^{-\mu}},$$

и поэтому

$$\frac{d}{dq}\log\psi(\mu) = O\left(\frac{1}{n(1-e^{-\mu})}\right) = O(n^{-3/4}) = \eta.$$

Если $0 \leqslant \mu < n^{-1/4}$, то нужно доказать, что

$$\frac{d}{dq}\log\{e^{\delta^2}\lambda(-\delta)\} = \eta. \tag{9.1}$$

Получаем

$$\frac{d\delta}{dq} = O(n^{-1/2}), \quad \frac{d}{d\delta}\log(e^{\delta^2}\lambda(-\delta)) = 2\delta - \{e^{-\delta^2}/\lambda(-\delta)\}.$$

Если δ велико, то воспользуемся (1.2); имеем

$$\frac{d}{d\delta}\log(e^{\delta^2}\lambda(-\delta)) = O\left(\frac{1}{\delta}\right) < C.$$

При $0 \leqslant \delta < A$ справедливо такое же утверждение. Отсюда следует (9.1).

Если $2q \leqslant n\log n$, имеем $\delta \leqslant 0$ и далее применяем теорему 6. Если

$$V\log V \leqslant 2q \leqslant 2(q+1) \leqslant (V+1)\log(V+1), \tag{9.2}$$

то положим $u = V(q) = V(q+1)$, если же (9.2) не выполняется, то $u = V(q)+1 = V(q+1)$. Легко доказать, что $\lambda(-\delta')/\lambda(-\delta) = 1+\eta$, где δ' — результат замены q на $q+1$ в определении δ. Отсюда получаем

$$R(n, q) = \{\Lambda(u, q+1)/\Lambda(u, q)\}(1+\eta) =$$

$$= \left\{\left(\frac{1}{2}u(u-1) - q\right)/(q+1)\right\}(1+\eta) = (v^2/2q)(1+\eta).$$

Это и завершает доказательство теоремы 8.

Поскольку $v \sim 2q/\log(2q)$, из теоремы 8 следует, что $R(n, q) > 1$ для

$$q_0 < q \leqslant (N/2) - 1$$

при некотором $q_0 = C$. Теперь обозначим $n_0 = 2q_0 + 2$ и предположим, что $1 \leqslant q \leqslant q_0$, $n \geqslant n_0$. Тогда в любом (n, q)-графе

должны быть по крайней мере две изолированные вершины. Пусть c — наибольшая степень вершин произвольного (n, q)-графа, и пусть G есть (n, q)-граф, содержащий такую вершину наибольшей степени. Можно построить $(n, q+1)$-граф, соединяя вершину наибольшей степени в G с одной из изолированных вершин; новый граф содержит вершину степени $c+1$. Но из любого (n, q)-графа можно построить $(n, q+1)$-граф, соединив две изолированные вершины новым ребром; наибольшая степень произвольной вершины этих $(n, q+1)$-графов равна c, и число таких графов равно $T(n, q)$, причем все они различны. Следовательно,

$$T(n,\ q+1) \geqslant T(n,\ q) + 1 > T(n,\ q).$$

Этим завершается доказательство теоремы 9.

СПИСОК ЛИТЕРАТУРЫ

1. De Bruijn N. G., Applied combinatorial mathematics, Ch. 5 (ed. E. F. Bechenbach) (New York, 1964).
2. Comrie L. J., Chambers' six-figure mathematical tables, II (1963), Table XIIA, pp. 518—19.
3. Euler L., Calcul de la probabilité dans le jeu de recontre, *Mém. Acad. Sçi. Berlin*, **7** (1753) 255—70; *Opera omnia* (1), **7** (1923), 11—25.
4. Euler L., Solutio questionis curiosae ex doctrina combinationum, *Mém. Acad. Sci. St. Petersbourg*, **3** (1811), 57—64; *Opera omnia* (1), **7** (1923), 435—48.
5. Ford G. W., Uhlenbeck G. E., Combinatorial problems in the theory of graphs, *Proc. Nat. Acad. Sci. U. S. A.*, **43** (1957), 163—67.
6. Gilbert E. N., Enumeration of labelled graphs, *Canad. J. Math.*, **8** (1956), 405—11.
7. Коршунов А. Д., О мощности некоторых классов графов, *ДАН СССР*, **193** (1970), 1230—33.
8. Oberschelp W., Kombinatorische Anzahlbestimmungen in Relationen, *Math. Ann.*, **174** (1967), 53—78.
9. Pólya G., Kombinatorische Anzahlbestimmungen fuer Gruppen, Graphen und Chemische Verbindungen, *Acta Math.*, **68** (1937), 145—254.
10. Stein M. L., Stein P. R., Enumeration of linear graphs and connected linear graphs up to $P = 18$ points (Los Alamos Scientific Laboratory, 1963).
11. Wright E. M., Asymptotic enumeration of connected graphs, *Proc. Roy. Soc. Edinburgh*, **A 68** (1970), 298—308.
12. Wright E. M., Graphs on unlabelled nodes with a given number of edges, *Acta Math.*, **126** (1971), 1—9.
13. Wright E. M., Unlabelled graphs with many nodes and edges, *Amer. Math. Soc. Notices*, **19** (1972), A-3.
14. Wright E. M. The probability of connectedness of an unlabelled graph can be less for more edges, *Proc. Amer. Math. Soc.*, **35**, (1972), 21—25.
15. Wright E. M., Arithmetical properties of Euler's recontre number, *J. London Math. Soc.*, (2), **4** (1972), 437—42.
16. Wright E. M., A generalisation of Euler's recontre number, in preparation.
17. Wright E. M., The number of unlabelled graphs with many nodes and edges, *Bull. Amer. Math. Soc.*, **78** (1972), 1032—34.
18. Wright E. M., The probability of connectedness of a large unlabelled graph, *Bull. Amer. Math. Soc.*, **79** (1973), 767—69.

Перечисление плоских триангуляций [1])

У. Т. Татт

1. ТРИАНГУЛЯЦИИ

Пусть P — замкнутая область на плоскости, ограниченная простой замкнутой кривой, и пусть S — симплициальное разбиение области P. Другими словами, S есть разбиение области P на конечное число α треугольников, так что ни одна из вершин одного треугольника не может быть внутренней точкой ребра любого другого треугольника. Треугольники являются «топологическими»; их ребра — замкнутые дуги — не обязательно являются отрезками прямой. Каждые две вершины соединены не более чем одним ребром, и любые два треугольника имеют не более двух общих вершин.

На границе области P лежат $k \geqslant 3$ вершин разбиения S, которые разделяют границу на k ребер (из S). Назовем эти ребра *внешними*, а остальные ребра разбиения из S, если такие существуют, *внутренними*. Пусть r — число внутренних ребер, тогда справедливы равенства

$$3\alpha = 2r + k, \tag{1.1}$$

$$r \equiv k \pmod 3. \tag{1.2}$$

Назовем S *триангуляцией* области P, если *не существует внутреннего ребра разбиения S, оба конца которого принадлежат границе области P*. Заметим, что в случае $k = 3$ каждое симплициальное разбиение является триангуляцией.

Пусть T_1 и T_2 — триангуляции области P, имеющие одни и те же внешние ребра. Эти триангуляции будем называть *изоморфными*, если существует взаимно однозначное отображение f вершин триангуляции T_1 на вершины триангуляции T_2, удовлетворяющее следующим условиям.

(i) *Каждая вершина границы области P отображается f на себя.*

(ii) *две различные вершины v и w триангуляции T_1 смежны в T_1 тогда и только тогда, когда вершины $f(v)$ и $f(w)$ смежны в T_2.*

---

[1] Tutte W. T., A census of planar triangulations, *Canad. J. Math.*, **XIV**, 1 (1962), 21—38.

(iii) *Три различные вершины u, v и w триангуляции T_1 образуют треугольник в T_1 тогда и только тогда, когда вершины $f(u)$, $f(v)$ и $f(w)$ образуют треугольник в T_2.*

Триангуляции многоугольника $abcd$, показанные на рис. 1A и 1B, изоморфны, а триангуляции, представленные на рис. 1B и 1C, не изоморфны.

Число неизоморфных триангуляций области P с k внешними и r внутренними ребрами зависит только от k и r, так как все

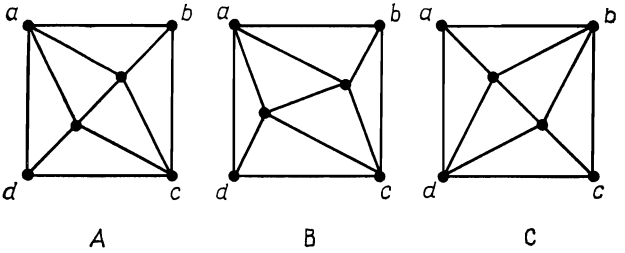


Рис. 1.

возможные области P гомеоморфны. Обозначим число таких триангуляций через $\psi_{n,m}$, где

$$k = m + 3, \tag{1.3}$$

$$r = 3n + m. \tag{1.4}$$

Ясно, что m и n должны быть неотрицательными целыми числами (см. [1, 2]). Заметим, что число триангуляций с тремя внешними ребрами и без внутренних ребер $\psi_{0,0} = 1$. Можно также проверить, что $\psi_{1,0} = 1$ и $\psi_{2,0} = 3$. В последнем случае триангуляции получаются тремя вращениями диаграммы, изображенной на рис. 2. Далее находим, что $\psi_{3,0} = 13$. Соответствующие триангуляции представлены четырьмя диаграммами на рис. 3. Из первой диаграммы при помощи вращения и отражения получаются шесть триангуляций. Следующие две диаграммы дают три триангуляции, а последняя — лишь одну.

Будет доказано, что при $n \geqslant 2$

$$\psi_{n,0} = \frac{2}{(n+1)!}(3n+3)(3n+4)\ldots(4n+1). \tag{1.5}$$

Основная цель настоящей статьи — точно оценить функцию $\psi_{n,m}$ (разд. 5).

Триангуляция называется *простой*, если любые три ребра, хотя бы одно из которых внутреннее, не образуют простую замкнутую кривую, ограничивающую область, разбитую на три или более треугольников. Таким образом, триангуляция, удовлетворяющая условию $n = 1$, $m = 0$, простая. В качестве другого примера можно взять последнюю диаграмму рис. 3.

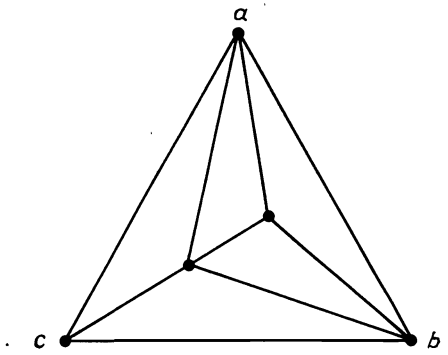


Рис. 2.

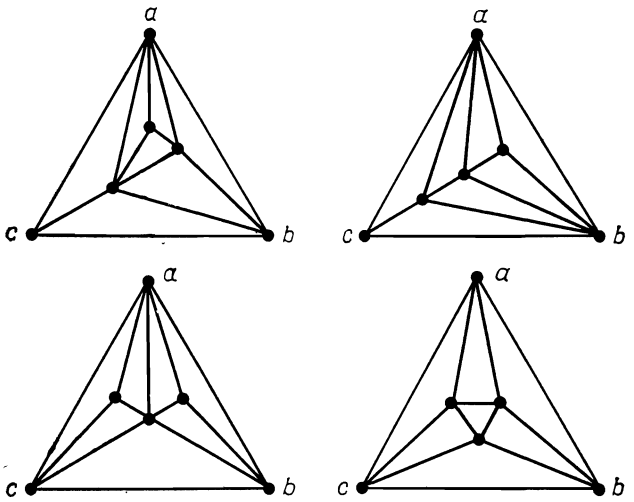


Рис. 3.

Обозначим через $\varphi_{n,m}$ число простых триангуляций с заданными значениями n и m. В разд. 7 получено явное выражение для функции $\varphi_{n,0}$, а в разд. 8 изучается поведение функций $\psi_{n,0}$ и $\varphi_{n,0}$ при $n \to \infty$.

Можно показать, что если T — триангуляция выпуклого многоугольника и вершины триангуляции, принадлежащие границе, являются вершинами этого многоугольника, то триангуляция T изоморфна некоторой триангуляции, ребра которой — отрезки прямой. Из [1] и теоремы Уитни следует, что 3-связный планарный граф можно уложить на плоскости, по существу, лишь одним способом [2]. В настоящей статье изучаются общие топологические триангуляции, так как в разд. 3 для доказательства

требуется рассмотреть триангуляции областей, не являющихся выпуклыми.

В разд. 9 исследовано при $n \to \infty$ поведение числа симплициальных разбиений 2-сфер, разделяющих 2-сферы на $2n$ треугольников.

2. ПРОИЗВОДЯЩИЕ ФУНКЦИИ

Далее будем пользоваться следующими формальными степенными рядами:

$$\psi(x, y) = \sum_{n=0}^{\infty} \sum_{m=0}^{\infty} \psi_{n,m} x^n y^m, \tag{2.1}$$

$$g(x) = \sum_{n=0}^{\infty} \psi_{n,0} x^n, \tag{2.2}$$

$$\varphi(x, y) = \sum_{n=0}^{\infty} \sum_{m=0}^{\infty} \varphi_{n,m} x^n y^m, \tag{2.3}$$

$$h(x) = \sum_{n=0}^{\infty} \varphi_{n,0} x^n. \tag{2.4}$$

В этом разделе будут получены формулы, позволяющие вычислять ряды φ и h, когда известны ψ и g.

Ясно, что каждой триангуляции T (с числом внутренних ребер $r > 0$) соответствует единственная простая триангуляция T' (с $r > 0$), так что или $T = T'$, или T можно получить из T' с помощью дальнейшего разбиения треугольников. Треугольник триангуляции T' является замкнутой областью, ограниченной тремя ребрами триангуляции T, хотя бы одно из которых внутреннее, и не содержащейся внутри другой такой области.

Пусть T' — простая триангуляция с r внутренними ребрами, k внешними ребрами и $\alpha > 1$ треугольниками. Триангуляции, которые можно получить из T', перечисляются при помощи ряда $\{g(x)\}^\alpha$. Коэффициент при x^p этого ряда — это число триангуляций с k внешними ребрами и $3p + \frac{1}{2}(3\alpha - k)$ внутренними ребрами. Определим m и n через параметры триангуляции T' с помощью равенств (1.3) и (1.4) и запишем

$$3p + \tfrac{1}{2}(3\alpha - k) = 3s + m,$$

$$s = p + \tfrac{1}{2}\alpha - \tfrac{1}{2}m - \tfrac{1}{2}.$$

Пусть число триангуляций с данным значением s, полученных из T', равно M_s. Тогда

$$\{g(x)\}^\alpha = \sum_{s=0}^{\infty} M_s x^{s-\frac{1}{2}\alpha+\frac{1}{2}m+\frac{1}{2}},$$

$$\left\{x^{\frac{1}{2}} g(x)\right\}^\alpha = \sum_{s=0}^{\infty} M_s x^s x^{\frac{1}{2}m+\frac{1}{2}}.$$

Пусть $q(\alpha, m)$ — число простых триангуляций с α треугольниками и $m+3$ данными внешними ребрами. Тогда

$$\sum_{\alpha=3}^{\infty} q(\alpha, m) \left\{x^{\frac{1}{2}} g(x)\right\}^\alpha = x^{\frac{1}{2}} \sum_{s=1}^{\infty} \psi_{s,m} x^s x^{\frac{1}{2}m}.$$

Из (1.1), (1.3) и (1.4) получаем $\alpha = 2n + m + 1$. Поэтому предыдущее равенство можно переписать так:

$$g(x) \sum_{n=1}^{\infty} \varphi_{n,m} \{x^{1/2} g(x)\}^{2n} \{g(x)\}^m = \sum_{s=1}^{\infty} \psi_{s,m} x^s.$$

Умножая обе части равенства на y^m и суммируя по m, получаем

$$g(x)\{\varphi(xg^2(x), yg(x)) - 1\} = \psi(x, y) - 1. \qquad (2.5)$$

Заметим, что из этого тождества можно получить коэффициенты для φ, если известны коэффициенты ряда ψ. В предлагаемой работе это выполняется для случая $m = 0$. Полагая $y = 0$ в (2.5), получаем

$$h(xg^2(x)) = 2 - \{g(x)\}^{-1}. \qquad (2.6)$$

До сих пор вопрос о сходимости производящих функций не обсуждался. Но в этом нет необходимости, если рассматривать уравнения типа (2.5) просто как равенства коэффициентов при $x^n y^m$ в разложениях функций, стоящих в правой и левой частях уравнений. Ниже будет ясно, что все такие коэффициенты являются конечными выражениями через коэффициенты функций ψ, g, φ и h. Однако в дальнейшем будем выражать производящие ряды через простые аналитические функции и, таким образом, установим их сходимость для соответствующих ненулевых значений x и y.

3. УРАВНЕНИЕ ДЛЯ ψ

Пусть T — триангуляция области P с k внешними и r внутренними ребрами, где $r \neq 0$. Пусть A — внешнее ребро с концами x и y, а z — третья вершина треугольника, границе которого принадлежит ребро A. Удалив из области P внутренность треугольника xyz и все точки ребра A, за исключением x и y,

получим другую область P', ограниченную простой замкнутой кривой. Если L — дуга, дополнительная к ребру A на границе области P, то граница области P' состоит из кривой L и двух ребер xz и yz (см. рис. 4).

Треугольники триангуляции T, отличные от xyz, определяют симплициальное разбиение S' области P'. Это не обязательно триангуляция, но любое внутреннее ребро разбиения S', оба конца которого находятся на границе области P', должно иметь один конец в z, а другой в отличной от x и y точке кривой L. Если существует c таких ребер и $c > 0$, то обозначим их через $E_1, \ldots, E_c$, а соответствующие концы ребер, принадлежащие

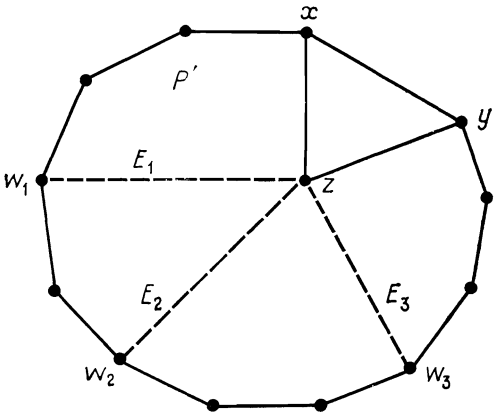


Рис. 4.

L, обозначим через $w_1, w_2, \ldots, w_c$, предполагая, что точки x, y, w_i встречаются на L в следующем порядке: x, w_1, w_2, ..., w_c, y.

При $c > 0$ ребра $E_1, \ldots, E_c$ разбивают область P' на $c+1$ замкнутых областей $P_0, P_1, \ldots, P_c$, ограниченных простыми замкнутыми кривыми zxw_1, zw_1w_2, ..., zw_cy соответственно, и триангуляция T порождает триангуляции $T_0, T_1, \ldots, T_c$ этих областей.

Если $c = 0$, то полагаем $P_0 = P'$ и $T_0 = S'$. В этом случае T_0 имеет по меньшей мере четыре внешних ребра. Если же $c > 0$, то триангуляции T_q таким образом не ограничиваются.

Ясно, что можно получить все триангуляции области P с k фиксированными внешними ребрами, одно из которых — ребро A, и заданными величинами r и c. Для этого начинаем с диаграммы, изображенной на рис. 4, и рассматриваем все множество триангуляций областей P_i, совпадающих с T на кривой L, которые содержат ребра xz, zw_1, ..., zw_c, zy и которые приводят к допустимому значению r.

Пусть величины m и n определены для триангуляции T из равенств (1.3) и (1.4). Пусть триангуляция T_q имеет k_q внешних и r_q внутренних ребер; m_q и n_q определяются соответственно. Тогда

$$\sum_{q=0}^{c} r_q = r - c - 2, \tag{3.1}$$

$$\sum_{q=0}^{c} k_q = k + 2c + 1, \tag{3.2}$$

$$\sum_{q=0}^{c} m_q = m - c + 1, \tag{3.3}$$

$$\sum_{q=0}^{c} n_q = n - 1. \tag{3.4}$$

Из приведенных выше соображений следует, что если $r > 0$, то

$$\psi_{n,m} = \sum_{c=0}^{\infty} \sum \prod_{q=0}^{c} \psi_{n_q, m_q}, \tag{3.5}$$

где второе суммирование ведется по всем упорядоченным множествам неотрицательных целых чисел $(n_0, \ldots, n_c, m_0, \ldots, m_c)$, удовлетворяющих равенствам (3.3) и (3.4). Но сумма

$$\sum \prod_{q=0}^{c} \psi_{n_q, m_q},$$

встречающаяся в равенстве (3.5), является коэффициентом при $x^{n-1} y^{m-c+1}$ в выражении $\{\psi(x, y)\}^{c+1}$, т. е. коэффициентом при $x^n y^m$ в

$$xy^{-2}\{y\psi(x, y)\}^{c+1}.$$

Отсюда не следует, что, просуммировав это выражение по c, получим ряд $\psi(x, y)$. Во-первых, $\psi(x, y)$ имеет свободный член $\psi_{00} = 1$, а равенство (3.5) не применимо в случае $r = 0$, т. е. $m = n = 0$. Во-вторых, по построению, связанному с рис. 4, требуется $k_0 \geqslant 4$, т. е. $m_0 \geqslant 1$, если $c = 0$. Принимая во внимание эти ограничения, получаем

$$\psi + xy^{-1}g = 1 + xy^{-2} \sum_{c=0}^{\infty} (y\psi)^{c+1}, \tag{3.6}$$

где $\psi = \psi(x, y)$ и $g = g(x)$.

Умножая обе части равенства (3.6) на $y(1 - y\psi)$ и перегруппировав члены, получаем

$$(y\psi + xg - y)(1 - y\psi) = x\psi, \tag{3.7}$$

что можно переписать следующим образом:

$$y^2\psi^2 + (x + xgy - y - y^2)\psi + y - xg = 0. \tag{3.8}$$

Теперь наша задача состоит в нахождении из уравнения (3.8) функции ψ в виде степенного ряда по степеням x и y и функции g в виде степенного ряда по неотрицательным степеням x. Из равенств (2.1) и (2.2) вытекает требование $g(x) = \psi(x, 0)$. Однако это условие не является независимым — оно получается из (3.8) подстановкой $y = 0$.

Переходим к доказательству того, что уравнение (3.8) имеет единственное решение относительно ψ. Полагая $x = 0$ в любом таком решении, получим степенной ряд ψ_0 по y; ряд ψ_0 должен удовлетворять условию

$$y^2\psi_0^2 - y(y+1)\psi_0 + y = 0,$$

т. е.

$$y(y\psi_0 - 1)(\psi_0 - 1) = 0.$$

Но единственный ряд по неотрицательным степеням y, который удовлетворяет тождественно этому уравнению, — это ряд с единственным членом 1.

Запишем теперь

$$\psi = \sum_{n=0}^{\infty} \psi_n x^n,$$

где ψ_i — степенной ряд по y. Было показано, что $\psi_0 = 1$. Приравнивая коэффициенты при x^n в (3.8) для $n > 0$, получим выражение $(y^2 - y)\psi_n$ через $\psi_0, \psi_1, \ldots, \psi_{n-1}$. Следовательно, по индукции ряд ψ определяется из равенства (3.8) единственным образом. (Используем тот факт, что $g(x) = \psi(x, 0) = \sum(\psi_n)_{y=0} x^n$.)

4. РЕШЕНИЕ УРАВНЕНИЯ (3.8)

Решение уравнения (3.8) можно получить из решения квадратного уравнения

$$Y^2 + Y(1 - t + \theta t) + \theta t = 0, \tag{4.1}$$

где θ и t — независимые переменные. Существуют два решения Y_1 и Y_2 уравнения (4.1) относительно Y в виде ряда по степеням t и θ, соответствующие постоянным членам 0 и -1. Эти решения можно найти, применив биномиальную теорему к выражению

$$\frac{1}{2}\{-(1 - t + \theta t) \pm \{(1 - t + \theta t)^2 - 4\theta t\}^{1/2}\}. \tag{4.2}$$

Отсюда получаем Y_1 или Y_2 в зависимости от знака перед корнем. Эти ряды сходятся для достаточно малых значений θ и t. Если взят положительный знак, то, когда одна из величин θ или

t равна нулю и какой бы ни была другая величина, выражение (4.2) обращается в нуль. Кроме того, получено, что в этом случае коэффициент при θt равен -1. Это утверждение следует из того, что $\frac{Y_1}{\theta t}$ и $\frac{\theta t}{Y_2}$ являются рядами по степеням θ и t (без отрицательных степеней) с постоянным членом -1. Значит, существует степенной ряд Φ относительно переменных θ и t (без отрицательных степеней), определяемый следующим уравнением:

$$(1-\theta)^3 t\Phi = \frac{\theta t}{Y_1(1+Y_1)^2} + 1 = \\ = 1 + \theta t\left\{\frac{1}{Y_1} - \frac{1}{1+Y_1} - \frac{1}{(1+Y_1)^2}\right\}. \qquad (4.3)$$

Обозначим

$$U_i = \frac{\theta t}{Y_i(1+Y_i)^2} \qquad (i = 1, 2).$$

Тогда

$$U_1U_2 = \frac{\theta^2 t^2}{Y_1Y_2(1+Y_1+Y_2+Y_1Y_2)^2} = \\ = \frac{\theta^2 t^2}{\theta t(1-(1-t+\theta t)+\theta t)^2} = \theta t^{-1}, \qquad (4.4)$$

$$U_1+U_2 = \frac{\theta t\left(Y_1+2Y_1^2+Y_1^3+Y_2+2Y_2^2+Y_2^3\right)}{Y_1Y_2(1+Y_1+Y_2+Y_1Y_2)^2} = \\ = t^{-2}\left\{(Y_1+Y_2)(Y_1+Y_2+1)^2 - Y_1Y_2(3Y_1+3Y_2+4)\right\} = \\ = t^{-2}\left\{(-1+(1-\theta)t)(1-\theta)^2t^2 - \theta t(1+3(1-\theta)t\right\} = \\ = -\theta t^{-1} - (1+\theta-2\theta^3) + t(1-\theta)^3. \qquad (4.5)$$

Следовательно,

$$(U_1+1)(U_2+1) = -\theta(1-2\theta) + t(1-\theta)^3,$$
$$(U_1+1)+(U_2+1) = -\theta t^{-1} + (1-\theta+2\theta^2) + t(1-\theta)^3.$$

Отсюда получаем, что Φ — решение квадратного уравнения

$$(1-\theta)^6 t^2\Phi^2 + (\theta t^{-1} - (1-\theta+2\theta^2) - \\ - t(1-\theta)^3)(1-\theta)^3 t\Phi - \theta(1-2\theta) + t(1-\theta)^3 = 0. \qquad (4.6)$$

Это уравнение становится эквивалентным уравнению (3.8), если положить $\psi = \Phi$ и

$$y = (1-\theta)^3 t, \qquad (4.7)$$
$$x = \theta(1-\theta)^3, \qquad (4.8)$$
$$xg = \theta(1-2\theta). \qquad (4.9)$$

Предположим, что θ — то решение, полученное из теоремы Лагранжа, уравнения (4.8), представленное в виде ряда по

степеням x, постоянный член которого равен нулю. Тогда из равенства (4.9) g определяется как аналитическая функция от x [3, стр. 132]. Действительно, по теореме Лагранжа имеем

$$g = \sum_{n=1}^{\infty} \frac{x^{n-1}}{n!} \left[\frac{d^{n-1}}{d\theta^{n-1}} \frac{1-4\theta}{(1-\theta)^{3n}} \right]_{\theta=0} =$$

$$= 1 + \sum_{n=1}^{\infty} \frac{x^n}{(n+1)!} \frac{d^n}{d\theta^n} \left[\frac{4}{(1-\theta)^{3n+2}} - \frac{3}{(1-\theta)^{3n+3}} \right]_{\theta=0} =$$

$$= 1 + \sum_{n=1}^{\infty} \frac{x^n}{(n+1)!} \{4(3n+2)(3n+3)\ldots(4n+1) -$$

$$- 3(3n+3)(3n+4)\ldots(4n+2)\},$$

$$g = 1 + x + 2\sum_{n=2}^{\infty} \frac{x^n}{(n+1)!}(3n+3)(3n+4)\ldots(4n+1). \qquad (4.10)$$

Из замечаний в конце разд. 3 заключаем, что ψ совпадает с функцией Φ, определенной равенством (4.3), а функция g действительно удовлетворяет равенству (4.10). Отсюда следует равенство (1.5).

5. КОЭФФИЦИЕНТЫ ФУНКЦИИ ψ

Уравнение (4.1) можно записать следующим образом:

$$Y = (t-1) - \theta t\left(1 + \frac{1}{Y}\right). \qquad (5.1)$$

Мы считаем t фиксированным ненулевым числом и используем теорему Лагранжа для разложения в ряд по степеням θ функции

$$U = \frac{\theta t}{Y(1+Y)^2}.$$

Получаем

$$U = \theta t\left\{\frac{1}{a(1+a)^2} + \sum_{j=1}^{\infty} \frac{(-\theta t)^j}{j!} \frac{d^{j-1}}{da^{j-1}}\left\{\left(\frac{1+a}{a}\right)^j \frac{d}{da}\left(\frac{1}{a(1+a)^2}\right)\right\}\right\}, \qquad (5.2)$$

где a должно быть приравнено к $t-1$ после того, как выполнено дифференцирование. Разумеется, функция U соответствует только одному из корней Y уравнения (5.1). Теперь можно написать

$$U = \frac{-\theta}{t(1-t)} - \theta t \sum_{j=1}^{\infty} \frac{(\theta t)^j}{j!} \frac{d^{j-1}}{dt^{j-1}}\left\{\left(\frac{t}{1-t}\right)^j \frac{1}{t^3}\left(\frac{-3}{1-t} + \frac{1}{(1-t)^2}\right)\right\} =$$

$$= -\frac{\theta}{t} - \frac{\theta}{1-t} - \theta t \sum_{j=1}^{\infty} \frac{\theta^j t^j}{j!} \frac{d^{j-1}}{dt^{j-1}}\left\{t^{j-3}\left(\frac{-3}{(1-t)^{j+1}} + \frac{1}{(1-t)^{j+2}}\right)\right\}. \qquad (5.3)$$

Запишем

$$A = t^{3j+3}\frac{d}{dt}\left(\frac{t^{1-3j}}{1-t}\right) = \frac{t^3}{(1-t)^2}(1-3j+3jt) =$$

$$= 3j(1-t)^2 - (9j+1)(1-t) + (9j+3) - \frac{3(j+1)}{1-t} + \frac{1}{(1-t)^2}. \quad (5.4)$$

Тогда

$$\frac{dA}{dt} = -6j(1-t) + (9j+1) - \frac{3\cdot 1\cdot (j+1)}{(1-t)^2} + \frac{2}{(1-t)^3}, \quad (5.5)$$

$$\frac{d^2A}{dt^2} = 6j - \frac{3\cdot 2\cdot 1\cdot (j+1)}{(1-t)^3} + \frac{3\cdot 2}{(1-t)^4}, \quad (5.6)$$

и, если $j > 2$, имеем

$$\frac{d^jA}{dt^j} = (j+1)!\left\{\frac{-3}{(1-t)^{j+1}} + \frac{1}{(1-t)^{j+2}}\right\}. \quad (5.7)$$

Применяя эти формулы к равенству (5.3), разложением функции $t^{1-3j}(1-t)^{-1}$ в ряд по степеням t и почленным дифференцированием получаем

$$U = -\theta t\sum_{j=1}^{\infty}\left[\frac{\theta^j t^j}{j!\,(j+1)!}\frac{d^{j-1}}{dt^{j-1}}\left\{t^{j-3}\frac{d^j}{dt^j}\left\{t^{3j+3}\frac{d}{dt}\left(\frac{t^{1-3j}}{1-t}\right)\right\}\right\}\right] -$$

$$-\frac{\theta}{t} - \frac{\theta}{1-t} + \theta t\left\{\frac{\theta t}{1!2!}t^{-2}(4+6t) + \frac{\theta^2t^2}{2!3!}\frac{d}{dt}12t^{-1}\right\} =$$

$$= -\theta t\sum_{j=1}^{\infty}\frac{\theta^j t^j}{j!\,(j+1)!}\sum_{m=1}^{\infty}\frac{(m+2)!\,(m-1)!\,(m-3j)}{(m-j+2)!\,(m-j)!}t^{m-j} -$$

$$-\frac{\theta}{t} - \frac{\theta}{1-t} + \theta^2(2+3t) - \theta^3 t.$$

Мы должны предположить, что $|t| < 1$. Пусть теперь U — одна из функций U_1 и U_2 из разд. 4. Через U' обозначим другую функцию. По равенству (4.5) функция U' не содержит членов с t^{-1} и может быть представлена степенным рядом относительно θ и t без отрицательных степеней, а функцию U, которая содержит член $-\theta/t$, представить так нельзя. Следовательно, U' — это функция U_1, использованная в (4.3). Поскольку ψ отождествлена с функцией Φ этого уравнения, из равенств (4.5) и (4.7) получаем

$$y\psi = \theta t\sum_{m=1}^{\infty}t^m\sum_{j=1}^{\infty}\frac{(m+2)!\,(m-1)!\,(m-3j)}{j!\,(j+1)!\,(m-j+2)!\,(m-j)!}\theta^j +$$

$$+\frac{\theta}{1-t} - \theta^2(2+3t) + \theta^3 t - (1+\theta-2\theta^2) + t(1-\theta)^3 + 1,$$

$$y\psi = t(1-2\theta) + \theta t\sum_{m=1}^{\infty}t^m\left\{1 + \sum_{j=1}^{m}\frac{(m+2)!\,(m-1)!\,(m-3j)}{j!\,(j+1)!\,(m-j+2)!\,(m-j)!}\theta^j\right\}.$$

Следовательно, по равенству (4.7) имеем

$$\psi = \frac{1-2\theta}{(1-\theta)^3} + \\ + \sum_{m=1}^{\infty} \frac{y^m}{(1-\theta)^{3m+3}} \sum_{j=0}^{m} \frac{(m+2)!\,(m-1)!\,(m-3j)}{j!\,(j+1)!\,(m-j+2)!\,(m-j)!}\,\theta^{j+1}. \qquad (5.8)$$

Для того чтобы выразить функцию ψ непосредственно через x и y, найдем разложение функции $\theta^p(1-\theta)^{-q}$ в ряд по степеням x, где p и q — положительные целые числа:

$$\frac{\theta^p}{(1-\theta)^q} = \sum_{s=1}^{\infty} \frac{x^s}{s!} \left[\frac{d^{s-1}}{d\theta^{s-1}} \left\{ \frac{1}{(1-\theta)^{3s}} \frac{d}{d\theta} \frac{\theta^p}{(1-\theta)^q} \right\}\right]_{\theta=0} =$$

$$= \sum_{s=1}^{\infty} \frac{x^s}{s!} \left[\frac{d^{s-1}}{d\theta^{s-1}} \left\{ \theta^{p-1} \left\{ \frac{q}{(1-\theta)^{3s+q+1}} + \frac{p-q}{(1-\theta)^{3s+q}} \right\}\right\}\right]_{\theta=0} =$$

$$= \sum_{s=0}^{\infty} \frac{x^s}{s!} \left\{ \frac{q}{(3s+q)!} \left[\frac{d^{s-1}}{d\theta^{s-1}} \left\{ \theta^{p-1} \frac{d^{3s+q}}{d\theta^{3s+q}} \left(\frac{1}{1-\theta}\right)\right\}\right]_{\theta=0} + \right.$$

$$\left. + \frac{p-q}{(3s+q-1)!} \left[\frac{d^{s-1}}{d\theta^{s-1}} \left\{ \theta^{p-1} \frac{d^{3s+q-1}}{d\theta^{3s+q-1}} \left(\frac{1}{(1-\theta)}\right)\right\}\right]_{\theta=0}\right\} =$$

$$= \sum_{s=p}^{\infty} \frac{x^s}{s!} \left\{ \frac{q\,(4s+q-p)!\,(s-1)!}{(3s+q)!\,(s-p)!} + \frac{(p-q)\,(4s+q-p-1)!\,(s-1)!}{(3s+q-1)!\,(s-p)!} \right\} =$$

$$= \sum_{s=p}^{\infty} \frac{x^s\,(4s+q-p-1)!}{s\,(s-p)!\,(3s+q)!} \{q\,(4s+q-p) + (p-q)\,(3s+q)\},$$

$$\frac{\theta^p}{(1-\theta)^q} = (3p+q) \sum_{s=p}^{\infty} \frac{(4s+q-p-1)!\,x^s}{(s-p)!\,(3s+q)!}. \qquad (5.9)$$

Приведенное доказательство справедливо также для $p > 0$ и $q = 0$. Следовательно, равенство (5.9) выражает величины θ, θ^2, θ^3 и т. д. через x.

Объединяя формулы (5.8) и (5.9), находим, что для $m > 0$

$$\psi_{n,m} = \sum_{j=0}^{m} \frac{(m+2)!\,(m-1)!\,(m-3j)}{j!\,(j+1)!\,(m-j+2)!\,(m-j)!} \frac{(4n+3m-j+1)!\,(3m+3j+6)}{(n-j-1)!\,(3n+3m+3)!},$$

где рассматриваются только значения $j \leqslant n-1$. Следовательно, для $m > 0$ имеем

$$\psi_{n,m} = \frac{3\,(m+2)!\,(m-1)!}{(3n+3m+3)!} \times$$

$$\times \sum_{j=0}^{m} \frac{(4n+3m-j+1)!\,(m+j+2)\,(m-3j)}{j!\,(j+1)!\,(m-j)!\,(m-j+2)!\,(n-j-1)!}. \qquad (5.10)$$

Для случая $m = 0$ из соотношения (4.10) имеем

$$\psi_{n,0} = \frac{2\,(4n+1)!}{(3n+2)!\,(n+1)!}. \qquad (5.11)$$

Частными случаями формулы (5.10) являются

$$\psi_{n,1} = \frac{72\,(4n+3)!}{(3n+6)!\,(n-1)!}, \qquad (5.12)$$

$$\psi_{n,2} = \frac{60\,(13n+9)\,(4n+5)!}{(3n+9)!\,(n-1)!}, \qquad (5.13)$$

$$\psi_{n,3} = \frac{120\,(67n^2+138n+68)\,(4n+7)!}{(3n+12)!\,(n-1)!}. \qquad (5.14)$$

6. ЗАМЕЧАНИЯ

Решение уравнения (3.8), приведенное выше, было получено следующим непрямым методом. Разложение функции ψ в ряд по степеням x, y и неизвестной функции g приводит к формуле

$$-x^{m+1}\psi_m = \sum_{\substack{0 \leqslant l \leqslant m+1 \\ 0 \leqslant k \leqslant \frac{l}{2}}} \frac{(-1)^l\,(2m-l-k+2)!\,x^k\,(xg+1)^{l-2k}}{(m-l+1)!\,(m-l+2)!\,k!\,(l-2k)!}. \qquad (6.1)$$

Здесь ψ_m — сумма членов функции ψ, содержащих y^m. Разложив $(xg+1)^{l-2k}$ и просуммировав по l, эту формулу можно заменить следующей:

$$-x^{m+1}\psi_m =$$

$$= \sum_{\substack{0 \leqslant j < \infty \\ 0 \leqslant k < \infty}} \frac{x^{j+k}\,(-g)^j\,(m-k+1)!\,(m-k)!}{j!\,k!\,(j+k-1)!\,(m-k-j+2)!\,(m-2k-j+1)!}. \qquad (6.2)$$

Если бы коэффициенты функции g были известны, то это соотношение дало бы коэффициент при x^n в левой части в виде многочлена от m при условии, что m достаточно велико. Ранее было доказано, что этот полином должен тождественно равняться

нулю. Следовательно, можно положить $m = 0$ в выражении слева и затем приравнять полученное выражение нулю:

$$\sum_{j,\ k} \frac{-(xg)^j x^k (2k+j-2)!(2k+j-3)!}{j!\,k!\,(k-2)!\,(k-1)!\,(k+j-1)!} = 0. \tag{6.3}$$

Используя это уравнение для вычисления первых нескольких членов ряда g, получим

$$g = 1 + x + 3x^2 + 13x^3 + 68x^4 + 399x^5 + 2530x^6 + \\ + 16\,965x^7 + 118\,668x^8 + 857\,956x^9 + 6\,369\,838x^{10} + \ldots. \tag{6.4}$$

(Последнее согласуется с формулой (5.11).) Затем была вычислена функция λ, обратная к xg, и оказалось, что ее коэффициенты имеют только небольшие простые множители. Изучение разложения на множители довольно быстро приводит к следующему выражению для функции λ:

$$\frac{1}{32}(1 + 20x - 8x^2 - (1 - 8x)^{3/2}).$$

Затем соотношение между g и x было представлено в параметрическом виде (4.8) и (4.9). После этого коэффициенты при y, y^2, ..., y^8 ряда ψ выражаются через θ с помощью приравнивания коэффициентов при y^m в равенстве (3.8). Опять оказалось, что числовые коэффициенты имеют простые множители, в результате был найден общий вид (5.8) ряда ψ.

На этом этапе получаем решение уравнения (3.8), но без доказательства. Автор весьма обязан Дж. Ф. Д. Даффу, показавшему как привести ряд (5.8) к виду, который можно рассматривать как разложение функции по теореме Лагранжа, что сделало возможным построения, выполненные в разд. 4 и 5.

7. ПРОСТЫЕ ТРИАНГУЛЯЦИИ

Перейдем теперь к нахождению функции h из тождества (2.6). Положим $x = z^2$ и $\xi = zg(z^2)$; тогда

$$\xi h(\xi^2) = 2\xi - z. \tag{7.1}$$

Поэтому, если обозначить через $H(x)$ функцию, обратную к $xg(x^2)$, получаем

$$h(\xi^2) = 2 - \xi^{-1} H(\xi), \tag{7.2}$$

и исходная задача сводится к определению H. Вычисляя первые несколько членов и записывая x вместо ξ^2, находим

$$h(x) = 1 + x + x^3 + 3x^4 + 12x^5 + 52x^6 + 241x^7 + 1173x^8 + \ldots. \tag{7.3}$$

Чтобы получить общую формулу, поступим следующим образом. Определим величину θ, как в разд. 4, и положим $x = z^2$. Тогда имеем

$$\xi H(\xi) = xg(x) = \theta(1-2\theta), \tag{7.4}$$

$$\xi^2 = \frac{(xg(x))^2}{x} = \frac{\theta(1-2\theta)^2}{(1-\theta)^3}, \tag{7.5}$$

$$\theta = \frac{\xi^2(1-\theta)^3}{(1-2\theta)^2}. \tag{7.6}$$

Применяя теорему Лагранжа, находим

$$\xi H(\xi) = \sum_{n=0}^{\infty} \frac{\xi^{2n+2}}{(n+1)!} \left[\frac{d^n}{d\theta^n} \left\{ \frac{(1-\theta)^{3n+3}}{(1-2\theta)^{2n+2}} (1-4\theta) \right\} \right]_{\theta=0},$$

$$\xi^{-1} H(\xi) =$$

$$= \sum_{n=0}^{\infty} \frac{\xi^{2n}}{(n+1)!} \left[\frac{d^n}{d\theta^n} \left\{ \frac{2^{-3n-3}(1+1-2\theta)^{3n+3}(2(1-2\theta)-1)}{(1-2\theta)^{2n+2}} \right\} \right]_{\theta=0} =$$

$$= \sum_{n=0}^{\infty} \frac{\xi^{2n}}{2^{3n+3}(n+1)!} \left[\frac{d^n}{d\theta^n} \left\{ \sum_{j=0}^{3n+3} \left\{ \binom{3n+3}{j} (1-2\theta)^j \right\} \times \right.\right.$$

$$\left.\left. \times \left\{ \frac{2}{(1-2\theta)^{2n+1}} - \frac{1}{(1-2\theta)^{2n+2}} \right\} \right\} \right]_{\theta=0} =$$

$$= \sum_{n=0}^{\infty} \frac{\xi^{2n}}{2^{3n+3}(n+1)!} \left[\frac{d^n}{d\theta^n} \left\{ 2 \sum_{j=0}^{3n+3} \binom{3n+3}{j} (1-2\theta)^{j-2n-1} - \right.\right.$$

$$\left.\left. - \sum_{j=0}^{3n+3} \binom{3n+3}{j} (1-2\theta)^{j-2n-2} \right\} \right]_{\theta=0} =$$

$$= \sum_{n=0}^{\infty} \frac{\xi^{2n}}{2^{3n+3}(n+1)!} \left[\frac{d^n}{d\theta^n} \left\{ 2(1-2\theta)^{n+2} - (1-2\theta)^{-2n-2} + \right.\right.$$

$$\left.\left. + \sum_{j=1}^{3n+3} \left\{ 2\binom{3n+3}{j-1} - \binom{3n+3}{j} \right\} (1-2\theta)^{j-2n-2} \right\} \right]_{\theta=0}$$

Но

$$2\binom{3n+3}{j-1} - \binom{3n+3}{j} = \frac{(3n+3)!}{j!(3n-j+4)!} \{2j - (3n-j+4)\} =$$

$$= \frac{(3n+3)!(3j-3n-4)}{j!(3n-j+4)!}.$$

Поэтому

$$\xi^{-1}H(\xi)=\sum_{n=0}^{\infty}\frac{\xi^{2n}}{2^{3n+3}(n+1)!}\left\{(-2)^n(n+2)!-2^n\frac{(3n+1)!}{(2n+1)!}+\right.$$

$$\left.+(-2)^n(3n+3)!\sum_{j=1}^{3n+3}\frac{(3j-3n-4)(j-2n-2)(j-2n-3)\ldots(j-3n-1}{j!(3n-j+4)!}\right\}=$$

$$=\sum_{n=0}^{\infty}\frac{\xi^{2n}}{2^{2n+3}(n+1)!}\left\{(-1)^n(n+2)!+(3n+3)!\times\right.$$

$$\left.\times\sum_{j=0}^{2n+1}\frac{(3n+1-j)!(3j-3n-4)}{j!(3n-j+4)!(2n+1-j)!}+(-1)^n(6n+5)(n+1)!+\right.$$

$$\left.+\frac{1}{2}(-1)^n(3n+3)(6n+2)n!\right\}.$$

Следовательно, для $n>0$ по формуле (7.2) имеем

$$\varphi_{n,0}=\frac{1}{2^{2n+3}}\left\{-(-1)^n(16n+10)+\right.$$

$$\left.+\frac{(3n+3)!}{(n+1)!}\sum_{q=0}^{n}\frac{M(n,q)}{(n-q)!(n+q+1)!}\right\},\qquad(7.7)$$

где $j=(n-q,\ n+q+1)$ и

$$M(n,q)=\frac{3q+4}{(2n+q+4)(2n+q+3)(2n+q+2)}-$$

$$-\frac{3q-1}{(2n-q+3)(2n-q+2)(2n-q+1)}.\qquad(7.8)$$

Формулу (7.8) можно привести к виду

$$M(n,q)=D^{-1}\left(40n^3-12n^2\left(6q^2+6q-11\right)-\right.$$

$$\left.-10n(15q^2+15q-14)-6(q^4+2q^3+13q^2+12q-8)\right),\qquad(7.9)$$

где D — произведение знаменателей в формуле (7.8).

8. АСИМПТОТИЧЕСКИЕ ФОРМУЛЫ

Исследуем поведение функции $\psi_{n,0}$ при $n \to \infty$ с помощью применения теоремы Стирлинга к формуле (5.11). Находим

$$\psi_{n,0} \sim \frac{2\,(4n+1)^{4n+1}\,e^{-4n-1}\sqrt{2\pi\,(4n+1)}}{(3n+2)^{3n+2}\,e^{-3n-2}\sqrt{2\pi\,(3n+2)}\,(n+1)^{n+1}\,e^{-n-1}\sqrt{2\pi\,(n+1)}} =$$

$$= \frac{2e^2\,(4n)^{4n+1}\left(1+\frac{1}{4n}\right)^{4n+1}}{(3n)^{3n+2}\left(1+\frac{2}{3n}\right)^{3n+2} n^{n+1}\left(1+\frac{1}{n}\right)^{n+1}} \sqrt{\frac{4n+1}{2\pi\,(n+1)\,(3n+2)}} \sim$$

$$\sim \frac{2\cdot 256^n}{n^2\cdot 27^n}\cdot\frac{4}{9}\sqrt{\frac{4}{6\pi n}},$$

$$\psi_{n,0} \sim \frac{1}{16}\sqrt{\frac{3}{2\pi}}\,n^{-5/2}\left(\frac{256}{27}\right)^{n+1}. \tag{8.1}$$

Поведение функции $\psi_{n,m}$ при $n \to \infty$ для других фиксированных значений m можно получить аналогичным образом из уравнения (5.10) или его частных случаев (5.12)—(5.14). Возможно, поведение функции $\varphi_{n,0}$ более интересно, и его можно определить следующим образом. Заметим, что

$$\sum_{q=0}^{n}\binom{2n+1}{n-q}(x^{n-q}+x^{n+q+1}) = (1+x)^{2n+1},$$

$$\sum_{q=0}^{n}\binom{2n+1}{n-q}\{(n-q)(n-q-1)\,x^{n-q-2}+(n+q+1)(n+q)\,x^{n+q-1}\} =$$

$$= (2n+1)(2n)(1+x)^{2n-1}.$$

Положив в этих тождествах $x=1$, получим

$$\sum_{q=0}^{n}\binom{2n+1}{n-q} = 2^{2n}, \tag{8.2}$$

$$\sum_{q=0}^{n}\binom{2n+1}{n-q}(q^2+q) = \tfrac{1}{2}n2^{2n}. \tag{8.3}$$

Также имеем

$$\binom{2n+1}{n-q} = \binom{2n+1}{n}\prod_{s=0}^{q-1}\frac{(n-s)}{(n+2+s)}. \tag{8.4}$$

Обозначим $J = \left[\frac{1}{2}n^{2/3}\right]$. Тогда по формулам (8.2) и (8.4) для достаточно больших n получаем, учитывая, что, если $|x|>0$,

то $1-x<e^{-x}$,

$$\binom{2n+1}{n-2J}<2^{2n}\left\{\prod_{s=J+1}^{2J+1}\frac{(n-s)}{(n+2+s)}\right\}\frac{(n+2J+2)(n+2J+3)}{(n-2J)(n-2J-1)}<$$

$$<2^{2n}\left\{1-\frac{(J+1)}{n}\right\}^{J+1}\times 2<$$

$$<2\cdot 2^{2n}\exp\left\{\frac{-(J+1)^2}{n}\right\}<2\cdot 2^{2n}\exp\left\{-\tfrac{1}{4}n^{1/3}\right\}. \quad (8.5)$$

Функция D в (7.9) удовлетворяет неравенству $D^{-1}<n^{-6}$, так как $0\leqslant q\leqslant n$. Сумма абсолютных величин коэффициентов многочлена от n и q, который умножается на D^{-1}, меньше 1000. Следовательно,

$$|M(n,\ q)|<1000n^{-2}. \quad (8.6)$$

Далее

$$\binom{2n+1}{n-q}$$

— убывающая функция от q при $0\leqslant q\leqslant n$ (по формуле (8.4)). Таким образом, из неравенств (8.5) и (8.6) получаем

$$\sum_{q=2J+1}^{n}\binom{2n+1}{n-q}<2n2^{2n}\exp\left\{-\tfrac{1}{4}n^{1/3}\right\}, \quad (8.7)$$

$$\sum_{q=2J+1}^{n}\binom{2n+1}{n-q}(q^2+q)<2n(n^2+n)2^{2n}\exp\left\{-\tfrac{1}{4}n^{1/3}\right\}, \quad (8.8)$$

$$\left|\sum_{q=2J+1}^{n}\binom{2n+1}{n-q}M(n,\ q)\right|<2000n^{-1}2^{2n}\exp\left\{-\tfrac{1}{4}n^{1/3}\right\}. \quad (8.9)$$

Ограничимся теперь значениями $q\leqslant 2J$. Для этих значений имеем

$$(2n+2J+4)^{-6}<D^{-1}<(2n-2J+1)^{-6}, \quad (8.10)$$

$$M(n,\ q)=D^{-1}(40n^3-72n^2(q^2+q))+\delta, \quad (8.11)$$

где

$$|\delta|<n^{-6}\cdot 1000n^{8/3}=1000n^{-10/3}. \quad (8.12)$$

Обозначим через ε произвольное действительное положительное число и обозначим

$$X=\sum_{q=0}^{2J}\binom{2n+1}{n-q}M(n,\ q).$$

Тогда для достаточно больших n по формулам (8.10), (8.11) и (8.12) имеем

$$X < \{40n^3(2n-2J+1)^{-6} + 1000n^{-10/3}\} \sum_{q=0}^{2J} \binom{2n+1}{n-q} -$$

$$- \{72n^2(2n+2J+4)^{-6}\} \sum_{q=0}^{2J} \binom{2n+1}{n-q} (q^2+q)$$

$$< \{40n^3(2n-2J+1)^{-6} + 1000n^{-10/3}\} \{2^{2n}\} -$$

$$- \{72n^2(2n+2J+4)^{-6}\} \left\{ \tfrac{1}{2} n 2^{2n} - 2n(n^2+n) 2^{2n} \exp\left\{ -\tfrac{1}{4} n^{1/3} \right\} \right\}$$

и по формулам (8.2) (8.3) и (8.8)

$$< n^3 2^{2n} \{40(2n)^{-6}(1-\varepsilon)^{-6} + \varepsilon n^{-6} - 36(2n)^{-6}(1+\varepsilon)^{-6}(1-\varepsilon)\}.$$

Аналогично для достаточно больших n получаем

$$X > \{40n^3(2n+2J+4)^{-6} - 1000n^{-10/3}\} \left\{ 2^{2n} - 2n 2^{2n} \exp\left\{ -\tfrac{1}{4} n^{1/3} \right\} \right\} -$$

$$- \{72n^2(2n-2J+1)^{-6}\} \left\{ \tfrac{1}{2} n 2^{2n} \right\} >$$

$$> n^3 2^{2n} \{(40(2n)^{-6}(1+\varepsilon)^{-6} - \varepsilon n^{-6})(1-\varepsilon) - 36(2n)^{-6}(1-\varepsilon)^{-6}\}.$$

Так как число ε может быть сколь угодно малым, то $X \sim \frac{1}{16} n^{-3} 2^{2n}$. Следовательно, из (7.7) и (8.9) по формуле Стирлинга имеем

$$\varphi_{n,0} \sim \frac{1}{2^{2n+3}} \left\{ -(-1)^n(16n+10) + \frac{(3n+3)!}{(n+1)!\,(2n+1)!} \frac{2^{2n}}{16n^3} \right\} \sim$$

$$\sim \frac{1}{2^{2n+3}} \left\{ -(-1)^n(16n+10) + \frac{2^{2n}}{16} \sqrt{\frac{3}{\pi}}\, n^{-5/2} \left(\frac{27}{4}\right)^{n+1} \right\}.$$

Отсюда вытекает

$$\varphi_{n,0} \sim \frac{1}{128} \sqrt{\frac{3}{\pi}}\, n^{-5/2} \left(\frac{27}{4}\right)^{n+1}. \qquad (8.13)$$

9. СИМПЛИЦИАЛЬНЫЕ РАЗБИЕНИЯ 2-СФЕРЫ

В этом заключительном разделе обсуждается поведение при $n \to \infty$ функции $Q(n)$, определенной как число комбинаторно различных симплициальных разбиений 2-сферы $2n$ треугольниками.

Изоморфизм таких разбиений можно определить так же, как изоморфизм триангуляций в разд. 1. Но, так как в рассматриваемом случае понятия границы не существует, необходимы

лишь условия (ii) и (iii). Когда говорят, что два разбиения комбинаторно различны, это означает, что они не изоморфны.

Автоморфизм разбиения S — это изоморфное отображение разбиения S на себя. Любое разбиение S имеет тривиальный или тождественный автоморфизм, который отображает каждую вершину на себя. Разбиение S называется *симметричным*, если оно имеет нетривиальный автоморфизм, и *несимметричным* в противном случае.

Заметим, что $n > 1$, так как любая пара треугольников симплициального разбиения имеет не более одного общего ребра. Обозначим через α_0, α_1 и α_2 число вершин, ребер и треугольников соответственно. Затем, используя формулу Эйлера для многогранников, получаем

$$\alpha_0 = n + 2, \quad \alpha_1 = 3n, \quad \alpha_2 = 2n. \tag{9.1}$$

Пусть t — треугольник разбиения S. Стереографическая проекция из точки X, принадлежащей внутренности треугольника t, на плоскость порождает триангуляцию T на этой плоскости с тремя внешними ребрами. Она определяется образами треугольников разбиения S, отличных от t, а ее внешние ребра являются образами ребер треугольника t. Если разбиение S не симметрично, то шесть вращений и отражений диаграммы триангуляции T определяют шесть различных триангуляций. Потому что, если две из них эквивалентны, очевидно, можно построить нетривиальный автоморфизм разбиения S. Аналогично два различных треугольника разбиения S никогда не приводят к эквивалентным триангуляциям. В случае если S симметрично, некоторые из его треугольников могут дать меньше чем шесть различных триангуляций и не все из его треугольников дадут различные множества триангуляций.

Пусть $Q_0(n)$ — число комбинаторно различных несимметричных разбиений, а $Q_1(n)$ — число комбинаторно различных симметричных разбиений. Тогда из приведенных выше соображений имеем

$$12nQ_0(n) \leqslant \psi_{n-1,0} \leqslant 12n\,(Q_0(n) + Q_1(n)). \tag{9.2}$$

Кажется, это верно, что $Q_1(n)/Q_0(n) \to 0$ при $n \to \infty$. Если это так, то из (9.2) и (8.1) следует, что

$$Q(n) \sim \frac{1}{64\sqrt{6\pi}}\, n^{-7/2} \left(\frac{256}{27}\right)^n. \tag{9.3}$$

Подобная аргументация применима и к числу $R(n)$ «простых» симплициальных разбиений сферы, на которой никакие три ребра не образуют простую замкнутую кривую, если они не

ограничивают треугольник разбиения. Вместо (8.1) используем формулу (8.13). Отсюда получаем, что

$$R(n) \sim \frac{1}{512\sqrt{3\pi}}\, n^{-7/2}\left(\frac{27}{4}\right)^n. \tag{9.4}$$

СПИСОК ЛИТЕРАТУРЫ

1. Tutte W. T., Convex representation of graphs, *Proc. London. Soc.*, 3rd ser., **10** (1960), 304—320.
2. Whitney H., 2-isomorphic graphs, *Amer. J. Math.*, **55** (1933) 245—254.
3. Whittaker E. T., Watson G. N., A course of modern analysis (Cambridge, 1940).

Четыре основных параметра кода и их комбинаторное значение[1])

Филипп Дельсарт
Исследовательская лаборатория MBLE,
Брюссель, Бельгия

На основании распределения расстояний произвольного кода и его преобразования Мак-Вильямс определяются четыре параметра кода, которые в линейном случае сводятся к минимальному весу и числу различных весов данного кода и дуального к нему. В общем случае выясняется комбинаторный смысл этих параметров и с помощью их получаются различные результаты относительно метрических свойств кода. В частности, предложен метод вычисления распределения весов классов смежности кода. Кроме того, представлено и подробно исследовано «дуальное понятие» для совершенного кода.

1. ВВЕДЕНИЕ

Распределение расстояний кода C длины n над конечным алфавитом F любого порядка $q \geqslant 2$ определяется как $(n+1)$-последовательность $(A_0(C), A_1(C), \ldots, A_n(C))$, где $A_i(C)$ обозначает среднее число кодовых слов, находящихся на расстоянии Хэмминга i от фиксированного кодового слова. Для линейного кода над некоторым полем или вообще для *аддитивного кода*, т. е. кода, образующего абелеву группу, это просто сводится к классическому распределению весов.

Для такого аддитивного кода определим четыре параметра d, s, d' и s'. Целые d и d' — минимальные веса данного кода C и дуального к нему кода C' соответственно, целые s и s' — числа различных весов кодов C и C' соответственно. Так как распределения весов кодов C и C' можно вывести одно из другого на основании *преобразования Мак-Вильямс* ([15], [7]), эти параметры однозначно определяются распределением весов кода C.

Для произвольного кода параметры d, s, d' и s' формально определяются тем же способом, т. е. на основании распределения весов кода и его преобразования Мак-Вильямс. Тогда d — обычное *минимальное расстояние* и s — *число расстояний*

---

[1]) Delsarte Ph., Four fundamental parameters of a code and their combinatorial significance, *Information and Control*, 23, № 5 (1973), 407—438.

между различными кодовыми словами. Как показано в этой статье, два остальных параметра также обладают интересными комбинаторными свойствами. Любая n-последовательность над F по крайней мере от одного кодового слова удалена не более чем на s'; по этой причине s' называется *внешним расстоянием* кода. С другой стороны, параметр d', называемый *дуальным расстоянием*, представляет собой наибольшее целое, обладающее тем свойством, что каждое $(d'-1)$-подмножество координат кода содержит всевозможные $(d'-1)$-последовательности над F постоянное число раз. Иными словами, $d'-1$ есть *максимальная сила ортогонального списка*, образованного кодовыми словами (по поводу этого понятия см. [4]).

В настоящей статье выведены, исходя из распределения расстояний, некоторые свойства кодов. Материал расположен следующим образом. В разд. 2 в тесной связи с *полиномами Кравчука* напоминаются определение и некоторые свойства формального преобразования Мак-Вильямс. Затем на основании распределения расстояний кода и его преобразования Мак-Вильямс вводятся четыре параметра.

Раздел 3 посвящен исследованию расстояний между словами кода и всего пространства F^n. В п. 3.1 определяется *матрица расстояний* для F^n относительно кода длины n как $(q^n \times (n+1))$-матрица, элемент (e, i) которой равен числу кодовых слов, находящихся на расстоянии i от данной n-последовательности e над F. Показано, что для кодов с данным распределением расстояний эта матрица однозначно определяется своими первыми s' столбцами. В качестве приложения с помощью результата Ассмуса и Мэттсона [2] найдено распределение весов всех классов смежности удлиненного двоичного квадратично-вычетного кода длины 48. В п. 3.2 установлена нижняя граница для числа слов в кодах, обладающих заданным внешним расстоянием s', а именно

$$|C| \geqslant q^n \Big/ \left(\sum_{i=0}^{s'} \binom{n}{i} (q-1)^i \right). \tag{1.1}$$

Показано, что эта оценка достигается для кода тогда и только тогда, когда он является совершенным кодом с исправлением s' ошибок.

В разд. 4 исследуются параметры s и d' с помощью *характеристических матриц* [7]. Сначала в п. 4.1 для q-ичных кодов длины n, имеющих s расстояний, получен дуальный аналог неравенства (1.1)

$$|C| \leqslant \sum_{i=0}^{s} \binom{n}{i} (q-1)^i. \tag{1.2}$$

Коды, для которых достигается равенство в (1.2), называются *обобщенными кодами Адамара* (ОА-кодами) *порядка s*. Для аддитивных кодов, в силу их дуальности, неравенства (1.1) и (1.2) выводятся одно из другого. Следовательно, понятие ОА-кода порядка s можно рассматривать как дуальное понятие к совершенному коду с исправлением s ошибок. Для ОА-кодов найдено сильное необходимое условие, состоящее в том, что расстояния такого кода должны быть нулями *полинома Ллойда* степени s (см. [13], [11] и [7] по поводу соответствующего результата для совершенных кодов). Затем в п. 4.2 выясняется комбинаторный смысл дуального расстояния d', о котором было упомянуто выше. Это приводит к новому доказательству принадлежащего Рао [19] неравенства для q-ичных ортогональных списков силы $d'-1$

$$|C| \geqslant \sum_{i=0}^{t'} \binom{n}{i} (q-1)^i, \tag{1.3}$$

где $t' = [(d'-1)/2]$. Фактически неравенство (1.3) можно считать дуальным к границе сферической упаковки в том же смысле, в каком (1.2) считалось дуальным к (1.1). Наконец, в п. 4.3 показано, что условие достижения равенства в (1.3) можно выбрать в качестве эквивалентного определения ОА-кодов порядка t'.

Раздел 5 посвящен *метрической инвариантности* кода и свойствам некоторых множеств кодовых слов фиксированного веса образовывать *t-блоки*. Показано, что любой код, параметры которого удовлетворяют любому из неравенств $d' \geqslant s$ или $d \geqslant s'$, метрически инвариантен. Результаты по t-блокам подобны результатам, полученным Ассмусом и Мэттсоном [1] для линейных кодов, особенно в двоичном случае; следует также упомянуть связь с работой Семакова, Зиновьева и Зайцева [20]. Приведены некоторые примеры, в частности блок-схемы, построенные из кодов, открытых Кердоком [10].

Обобщенные коды Адамара (ОА-коды) подробно исследованы в разд .6. Найдена явная формула для распределения расстояний (или весов). В качестве применения показано, что единственными существующими ОА-кодами третьего порядка над некоторым полем являются два хорошо известных двоичных линейных кода.

2. ОПРЕДЕЛЕНИЯ И ПРЕДВАРИТЕЛЬНЫЕ ЗАМЕЧАНИЯ

Прежде чем вводить четыре указанных параметра, приведем некоторый материал по распределению расстояний кода и его преобразованию Мак-Вильямс. Для дальнейшего вспомним

определение скалярного произведения в абелевых группах и его связи с весом Хэмминга и полиномами Кравчука. Доказательства теорем, сформулированных в этом разделе, можно найти в недавней работе автора [7]; каждый раз даются точные ссылки на соответствующие результаты этой работы.

2.1. **Преобразование Мак-Вильямс.** При заданных положительных целых числах n и λ *полином Кравчука* $P_k(x)$ степени k над полем рациональных чисел определяется формулой ([23])

$$P_k(x) = \sum_{j=0}^{k} (-1)^j \lambda^{k-j} \binom{x}{j} \binom{n-x}{k-j}, \quad 0 \leqslant k \leqslant n, \tag{2.1}$$

где $\binom{x}{j} = x(x-1)\ldots(x-j+1)/j!$. Если задана $(n+1)$-последовательность $A = (A_0, A_1, \ldots, A_n)$ рациональных чисел A_i, то ее *преобразованием Мак-Вильямс* называется $(n+1)$-последовательность $A' = (A'_0, A'_1, \ldots, A'_n)$, где

$$A'_k = \sum_{i=0}^{n} A_i P_k(i), \quad 0 \leqslant k \leqslant n, \tag{2.2}$$

и $P_k(x)$ — полином Кравчука (2.1). В матричной форме это можно записать в виде $A' = AP$, где P обозначает квадратную матрицу порядка $n+1$, элемент (i, k) которой есть $P_k(i)$:

$$P = [P_k(i);\ 0 \leqslant i,\ k \leqslant n]. \tag{2.3}$$

Будем называть эту матрицу *матрицей Кравчука*. Из равенства $P^2 = (\lambda+1)^n I$ [7, теорема 3] выводится следующий результат, принадлежащий Мак-Вильямс [15]:

Теорема 2.1. *Преобразование Мак-Вильямс взаимно однозначно и удовлетворяет соотношению* $(A')' = (\lambda+1)^n A$ *для любой* $(n+1)$*-последовательности* A.

Для того чтобы установить соответствие между приведенным выше определением и обычной формой тождеств Мак-Вильямс, заметим, что (2.2) можно записать в полиномиальной форме как

$$A'(y, z) = A(z-y, z+\lambda y),$$

где $A(y, z) = \sum A_i y^i z^{n-i}$ и $A'(y, z) = \sum A'_k y^k z^{n-k}$.

Определения. (1) При заданной $(n+1)$-последовательности $A = (A_0, A_1, \ldots, A_n)$ рациональных чисел A_i определим $s(A)$ как число ненулевых компонент A_i, $1 \leqslant i \leqslant n$, и $d(A)$ как наименьшее целое k, $1 \leqslant k \leqslant n$, такое, что A_k отлично от нуля (если такое целое существует).

(2) При заданных n и λ $(n+1)$-последовательность A будем называть *положительной*, если $A_0 = 1$, $A_i \geqslant 0$, $A'_i \geqslant 0$ при $i = 0, 1, \ldots, n$, где $A' = (A'_0, \ldots, A'_n)$ — преобразование Мак-Вильямс (2.2) для A.

Следующие два результата [7, теоремы 4 и 16] иллюстрируют эти определения. Значение их в теории кодирования будет ясно из дальнейшего.

Теорема 2.2 (ср. [14]). *Пусть A есть $(n+1)$-последовательность с $A_0 \neq 0$ и A' — ее преобразование Мак-Вильямс. Тогда*

$$s(A') \geqslant [(d(A) - 1)/2].$$

Теорема 2.3 (граница Хэмминга и теорема Ллойда). *Произвольная положительная $(n+1)$-последовательность A удовлетворяет неравенству*

$$\sum_{k=0}^{t} \binom{n}{k} \lambda^k \sum_{i=0}^{n} A_i \leqslant q^n, \tag{2.4}$$

где $q = \lambda + 1$ и $t = [(d(A) - 1)/2]$. Кроме того, если в (2.4) достигается равенство для положительной последовательности A, то $s(A') = t$ и полином Ллойда

$$Q_t(x) = P_0(x) + P_1(x) + \ldots + P_t(x) \tag{2.5}$$

степени t имеет ровно t различных нулей в множестве $\{1, 2, \ldots, n\}$, а именно таких целых чисел $k \geqslant 1$, что $A'_k \neq 0$.

2.2 Скалярное произведение и вес Хэмминга. Пусть F — конечная абелева группа порядка $q \geqslant 2$ и периода ν, представленная в виде аддитивной группы. Пусть Γ_ν — циклотомическое поле комплексных корней ν-й степени из единицы и χ_u — групповой характер F над Γ_ν, связанный с элементом $u \in F$; нумерация предполагается такой, что $\chi_u(v) = \chi_v(u)$ для любых $u, v \in F$. При заданном положительном целом n введем *скалярное произведение* $\langle\ ,\ \rangle$ на прямом произведении $G = F^n$ из n копий F:

$$\langle a, b \rangle = \prod_{i=1}^{n} \chi_{a_i}(b_i), \tag{2.6}$$

где $a = (a_1, \ldots, a_n)$, $b = (b_1, \ldots, b_n)$ принадлежат G. При фиксированном $a \in G$ соответствие $b \to \langle a, b \rangle$ является характером группы G над Γ_ν, если для любых $a, b, c \in G$

$$\langle a, b + c \rangle = \langle a, b \rangle \langle a, c \rangle. \tag{2.7}$$

Более того, любой характер группы G можно представить подобным образом.

Вес Хэмминга $w_H(a)$ элемента $a = (a_1, \ldots, a_n)$ из G определяется обычно как число ненулевых компонент a_i из F, $1 \leqslant i \leqslant n$. Обозначим через Y_k множество элементов веса k в G:

$$Y_k = \{h \in G \mid w_H(h) = k\}, \tag{2.8}$$

$k = 0, 1, \ldots, n$. Следующий результат [7, теорема 2] устанавливает связь между введенными выше понятиями при условии, что $q = \lambda + 1$. Это условие будем предполагать выполненным на протяжении всей статьи.

Теорема 2.4. *Для произвольного элемента* $a \in G$ *веса* $w_H(a) = u$

$$\sum_{h \in Y_k} \langle a, h \rangle = P_k(u),$$

где $P_k(x)$ *— полином Кравчука степени* k.

Расстояние Хэмминга $d_H(a, b)$ между двумя элементами a и b из G есть число координат, в которых они различаются, или, что то же, $d_H(a, b)$ есть вес Хэмминга разности $a - b$ в G. Пусть a, b — элементы из G, находящиеся на заданном расстоянии $d_H(a, b) = k$ друг от друга. Тогда легко видеть, что число элементов $c \in G$, находящихся на расстоянии i от a и на расстоянии j от b, зависит только от i, j и k. Для таких ассоциативных схем [5] это число обычно обозначается через $p_{i,j}^{(k)}$.

Теорема 2.5. *При заданных* $i, j, u \in \{0, \ldots, n\}$ *полиномы Кравчука удовлетворяют соотношению*

$$P_i(u) P_j(u) = \sum_{k=0}^{n} p_{i,j}^{(k)} P_k(u). \tag{2.9}$$

Доказательство. Пусть $h \in Y_k$. Из определения следует, что $p_{i,j}^{(k)}$ — число пар (f, g), где $f \in Y_i$, $g \in Y_j$ и $f + g = h$. Тогда формула (2.9) выводится из теоремы 2.4 с помощью элементарных вычислений (см. (2.7)).

2.3. Распределение расстояний кода. Для алфавита F порядка $q \geqslant 2$ определяется (n, M)-*код над* F просто как непустое множество из M различных n-последовательностей над F, т. е. подмножество в F^n мощности M, где $1 \leqslant M \leqslant q^n$. Код называется *тривиальным*, если $M = 1$ или $M = q^n$. В дальнейшем предполагается, что F образует аддитивную абелеву группу, так что *слова* кода рассматриваются как элементы группы $G = F^n$.

Последнее, конечно, не налагает никаких ограничений при исследовании существенных свойств кода.

Распределением расстояний (n, M)-кода C называется $(n+1)$-последовательность $A(C) = (A_0(C), A_1(C), \ldots, A_n(C))$ неотрицательных рациональных чисел

$$A_i(C) = \frac{1}{M} |\{(a, b) \in C^2 \mid d_H(a, b) = i\}|. \tag{2.10}$$

Иными словами, $A_i(C)$ есть среднее число кодовых слов, находящихся на расстоянии i от фиксированного кодового слова. Важность понятия положительной $(n+1)$-последовательности в теории кодирования подчеркивается следующим результатом [7, теорема 6], новое доказательство которого получено в этой статье:

Теорема 2.6. *Распределение расстояний любого кода длины n над q-ичным алфавитом является положительной $(n+1)$-последовательностью (при $\lambda = q - 1$).*

В соответствии с этим формальное неравенство (2.4), примененное к распределению расстояний $A = A(C)$, приводит к хорошо известной границе Хэмминга (или границе сферической упаковки). Введем теперь четыре *основных параметра* нетривиального q-ичного кода C. Каждый из них можно вычислить из распределения расстояний $A(C)$:

$d = d(A(C))$ = *минимальное расстояние*,

$s = s(A(C))$ = *число расстояний*,

$d' = d(A'(C))$ = *дуальное расстояние* (= 1 + *максимальная сила*),

$s' = s(A'(C))$ = *внешнее расстояние*,

где $A'(C)$ — преобразование Мак-Вильямс (2.2) последовательности $A(C)$ при $\lambda = q - 1$. Названия, данные s и d, не нуждаются в комментариях. Для s' и d' они обоснованы в разд. 3 и 4 соответственно. Отметим здесь, что теорема 2.2 подразумевает оба неравенства $s' \geqslant [(d-1)/2]$ и $s \geqslant [(d'-1)/2]$, которые будут получены еще раз ниже (ср. (1.1)—(1.3)).

Пусть C — *аддитивный код* над абелевой группой F, т. е. подгруппа $G = F^n$. Тогда очевидно, что распределение расстояний кода C сводится к его распределению весов: $A_i(C)$ — число кодовых слов веса i. *Дуальный код* C' определяется формулой [7]

$$C' = \{a \in G \mid \langle a, b\rangle = 1 \quad \forall b \in C\}$$

и также является аддитивным кодом. Такая дуальность есть инволюция с $C' \cong G/C$ и сводится к классическому понятию для линейных кодов над конечными полями (см. [15]). Как и в

случае линейных кодов, распределение весов кода C' с точностью до постоянного множителя равно преобразованию Мак-Вильямс распределения весов кода C [7, теорема 8]. Следовательно, для аддитивного кода C параметры d' и s' равны соответственно параметрам d и s дуального кода C'.

В остальной части этой статьи исследуются свойства произвольных (не обязательно аддитивных) кодов, связанные с их основными параметрами. Начиная с этого места, все коды предполагаются нетривиальными.

3. МАТРИЦА РАССТОЯНИЙ И ВНЕШНЕЕ РАССТОЯНИЕ

В настоящем разделе вводится в рассмотрение матрица расстояний; она понадобится для описания расстояний между данным кодом и всем хэмминговым пространством, в которое код погружен. Затем исследуются свойства этой матрицы в связи с внешним расстоянием s'.

3.1. Матрица расстояний. Для q-ичного кода C длины n обозначим через $B_i(e)$ число кодовых слов, находящихся на расстоянии i от фиксированной точки e из $G = F^n$, т. е.

$$B_i(e) = |\{a \in C \mid d_H(a, e) = i\}|, \quad 0 \leqslant i \leqslant n. \tag{3.1}$$

Матрицей расстояний группы G относительно C называется $(q^n \times (n+1))$-матрица, строки и столбцы которой нумеруются элементами группы G и множества $\{0, 1, \ldots, n\}$ соответственно и элемент (e, i) которой есть $B_i(e)$:

$$B = [B_i(e);\ e \in G,\ 0 \leqslant i \leqslant n]. \tag{3.2}$$

Ясно, что строка $B(e)$ — это распределение весов «смежного кода» C—e. Теорема 3.1 играет основную роль в этом разделе:

Теорема 3.1. *Пусть C будет q-ичным (n, M)-кодом с заданным распределением расстояний $A(C)$. Тогда матрица расстояний группы G относительно C имеет ранг $s'+1$ и удовлетворяет соотношению*

$$B^T B = (Mq^{-n}) P^T \Lambda P, \tag{3.3}$$

где P — матрица Кравчука (2.3) с $\lambda = q-1$, а Λ — диагональная матрица, определенная с помощью преобразования Мак-Вильямс $A'(C)$ последовательности $A(C)$:

$$\Lambda = \operatorname{diag}(A'_0(C), A'_1(C), \ldots, A'_n(C)). \tag{3.4}$$

Доказательство. Сначала заметим, что при $i, j = 0, 1, \ldots, n$ справедливы тождества

$$\sum_{e \in G} B_i(e) B_j(e) = M \sum_{k=0}^{n} p_{i,j}^{(k)} A_k(C), \tag{3.5}$$

где параметры $p_{i,j}^{(k)}$ определены в разд. 2.2. Действительно, обе части равенства (3.5) равны числу троек (a, b, e) элементов $a,\ b \in C,\ e \in G$, для которых $d_H(a, e) = i,\ d_H(b, e) = j$. Применяя последовательно теоремы 2.1 и 2.5, преобразуем (3.5) в

$$\sum_{e \in G} B_i(e) B_j(e) = Mq^{-n} \sum_{u=0}^{n} A'_u(C) \sum_{k=0}^{n} p_{i,j}^{(k)} P_k(u) =$$

$$= Mq^{-n} \sum_{u=0}^{n} P_i(u) A'_u(C) P_j(u).$$

Последнее есть искомое равенство (3.3). Вспоминая далее, что $s'+1$ равно числу ненулевых компонент в $A'(C)$ и матрица P не вырождена, на основании (3.3) и (3.4) имеем ранг$(B) =$ $=$ ранг$(B^TB) =$ ранг$(\Lambda) = s'+1$, и теорема доказана.

Каждому полиному $\alpha(x)$ степени не выше n над полем рациональных чисел поставим в соответствие $(n+1)$-последовательность $\alpha = (\alpha_0, \alpha_1, \ldots, \alpha_n)$ его компонент в базисе полиномов Кравчука:

$$\alpha(x) = \alpha_0 P_0(x) + \alpha_1 P_1(x) + \ldots + \alpha_n P_n(x). \tag{3.6}$$

Следующая теорема очень полезна для действительного вычисления матрицы расстояний:

Теорема 3.2. *Пусть C будет (n, M)-кодом с заданным внешним расстоянием s', а B — матрицей расстояний группы G относительно C. Тогда первые $s'+1$ столбцов матрицы B (занумерованные числами $0, 1, \ldots, s'$) порождают все пространство столбцов. Точнее, для любого целого i, $s' \leqslant i \leqslant n$, столбец с номером i матрицы B представляет собой линейную комбинацию первых s' столбцов и $(1, 1, \ldots, 1)^T$, коэффициенты которой зависят только от распределения расстояний кода.*

Доказательство. Умножая обе части равенства (3.3) слева на P^T и справа на P, получаем

$$(BP)^T(BP) = Mq^n\Lambda; \tag{3.7}$$

здесь учтено, что $P^2 = q^nI$ (см. теорему 2.1). Пусть $B'(e) =$ $= (B'_k(e))$ обозначает преобразование Мак-Вильямс $(n+1)$-последовательности $B(e) = (B_i(e))$. Равенство диагональных элементов матриц в (3.7) дает

$$\sum_{e \in G} (B'_k(e))^2 = Mq^n A'_k(C), \qquad 0 \leqslant k \leqslant n. \tag{3.8}$$

С другой стороны, пусть J — множество целых k, $1 \leqslant k \leqslant n$, для которых k-я компонента $A'_k(C)$ последовательности $A'(C)$

отлична от нуля. Для любого целого m, $0 \leqslant m \leqslant n - s'$, определим полином

$$\beta(x) = M^{-1} q^n x^m \prod_{i \in J} (1 - x/i) \tag{3.9}$$

степени $m + s'$ (так как $|J| = s'$). По определению $\beta(k) A'_k(C) = 0$ при $k \neq 0$. В силу (3.8), это с очевидностью влечет $\beta(k) B'_k(e) = 0$ для каждого $e \in G$, $k \neq 0$. Используем теперь тождество [7, равенство (18)]

$$\sum_{i=0}^{n} \alpha_i A_i = q^{-n} \sum_{k=0}^{n} \alpha(k) A'_k,$$

справедливое для любых двух $(n+1)$-последовательностей α, A (см. (2.2) и (3.6)). При $\alpha = \beta$ и $A = B(e)$ оно принимает вид

$$\sum_{i=0}^{n} \beta_i B_i(e) = q^{-n} \beta(0) B'_0(e),$$

где $\beta = (\beta_0, \beta_1, \ldots, \beta_{m+s'}, 0, \ldots, 0)$ есть $(n+1)$-последовательность компонент $\beta(x)$ в базисе полиномов Кравчука. Таким образом, применяя (3.9) и замечая, что $B'_0(e) = M$, получаем в матричной форме

$$\beta B^T = \delta_{0,m} j^T, \quad 0 \leqslant m \leqslant n - s', \tag{3.10}$$

где j — вектор из всех единиц и δ — символ Кронекера. Так как $\beta_{m+s'} \neq 0$, то (3.10) выражает при всех m строку с номером $m + s'$ в B^T как линейную комбинацию j^T и строк с номерами $0, 1, \ldots, m + s' - 1$. Следовательно, теорема доказывается индукцией по m, так как β полностью определяется распределением расстояний кода C.

Замечание. Равенство (3.8) показывает, что все компоненты последовательности $A'(C)$ неотрицательны. Это доказывает теорему 2.6.

Для того чтобы проиллюстрировать приведенные выше результаты, рассмотрим теперь подробно один пример. Пусть C обозначает удлиненный квадратично-вычетный $(48, 2^{24})$-код над $GF(2)$. Хорошо известно, что C — самодуальный линейный код с восемью ненулевыми весами (или расстояниями), а именно 12, 16, 20, 24, 28, 32, 36 и 48 (см. [16]), так что его параметрами являются $d = d' = 12$, $s = s' = 8$. В дополнение к этому нам необходим результат Ассмуса и Мэттсона [2], устанавливающий число 6-множеств, содержащихся в данном числе 12-множеств, которые соответствуют кодовым словам минимального веса. Эта информация достаточна для вычисления распределения весов всех классов смежности кода C, или, что эквивалентно, матрицы

расстояний G относительно C. Как будет ясно из дальнейшего, оказывается, матрица B имеет только 14 различных строк. В табл. 1 приведены первые 11 компонент этих строк, а также их кратности в B (с точностью до множителя 2^{-24}).

Таблица 1

Распределение весов классов смежности удлиненного двоичного квадратично-вычетного кода длины 48 (первые 11 компонент)

$k(e)$ \ i	0	1	2	3	4	5	6	7	8	9	10	кратности
0	1											1
1		1										48
2			1								1 012	1 128
3				1						220		17 296
4					1				44		6 336	194 580
5						1		8		180		1 712 304
6							1		48		777	2 334 960
6							2		42		786	2 814 924
6							3		36		795	916 688
6							4		30		804	350 244
6							5		24		813	25 944
6							6		18		822	4 324
7								9		205		6 658 960
8									54		768	1 745 815
												Сумма $= 2^{24}$

Строка $B(e)$ матрицы характеризуется прежде всего наименьшим целым $k = k(e)$, для которого $B_k(e) \neq 0$, т. е. минимальным расстоянием между e и кодовыми словами. Из теоремы 3.2 с очевидностью следует, что $k(e)$ не превышает $s' = 8$ (общий результат дается теоремой 3.3). Вычисляя при $J = \{12, 16, 20, 24, 28, 32, 36, 48\}$ компоненты β_i полинома (3.9) при $m = 0$, получаем

$$\beta_0 = \beta_1 = \beta_2 = \beta_3 = 1, \quad \beta_4 = {}^5/_{27}, \quad \beta_5 = \beta_6 = \beta_7 = {}^1/_9, \quad \beta_8 = {}^1/_{54}$$

и $\beta_i = 0$ при $i \geqslant 9$. Отсюда, используя (3.10), находим формулы для параметров $B_k(e)$, $k \leqslant 8$:

$$\begin{aligned}
&B_8(e) = 44 && \text{при } k(e) = 4,\\
&B_7(e) = 8 && \text{при } k(e) = 5,\\
&B_8(e) = 54 - 6B_6(e) && \text{при } k(e) = 6,\\
&B_7(e) = 9 && \text{при } k(e) = 7,\\
&B_8(e) = 54 && \text{при } k(e) = 8.
\end{aligned}$$

Следовательно, по теореме 3.2 строка $B(e)$ матрицы B полностью определяется значением $k(e)$, за исключением случая $k(e)=6$. Фактически Ассмус и Мэттсон [2] показали, что $B_6(e)$ принимает только значения 1, 2, ..., 6. Это дает первые 9 столбцов табл. 1, где пустые места считаются нулями. Десятый столбец вычисляется затем с помощью (3.10) при $m=1$:

$$108\beta(x) = 1920P_3(x) - 20P_5(x) + 205P_7(x) - 9P_9(x).$$

Отсюда легко вычислить значения $B_9(e)$, указанные в табл. 1. Другие столбцы матрицы B можно последовательно получить тем же методом.

Наконец, число классов смежности C в G с данным распределением весов среди 14 возможных определяется следующим образом. Для классов смежности, минимальный вес k которых меньше 6, это число просто равно биномиальному коэффициенту $\binom{48}{k}$; для шести классов с $k(e)=6$ его можно найти из результата Ассмуса и Мэттсона [2]; для $k(e)=7$ и $k(e)=8$ оно вычисляется из предыдущих значений и очевидного тождества

$$\sum_{e\in G} B_k(e) = M\binom{n}{k}\lambda^k,$$

справедливого для любого (n, M)-кода.

3.2. Граница внешнего расстояния. Теперь мы можем оправдать название «внешнее расстояние», данное параметру s', и доказать неравенство (1.1).

Теорема 3.3. *Для любого кода C с заданным внешним расстоянием s' каждая точка из G удалена не более чем на s' по крайней мере от одного кодового слова.*

Доказательство. Предположим, что найдется точка e в G, для которой $B_0(e)=B_1(e)=\ldots=B_{s'}(e)=0$, где B — матрица расстояний (3.2) группы G относительно C. Тогда по теореме 3.2 $B_i(e)=0$ при $i=0, 1, \ldots, n$. Полученное противоречие доказывает теорему.

Как мы убедились, для приведенного выше $(48,2^{24})$-кода действительно существуют точки в G, находящиеся на расстоянии s' от кодовых слов. Следует отметить, что в общем случае это не всегда так.

Теорема 3.4. *Для любого (n, M)-кода C с заданным минимальным расстоянием d и внешним расстоянием s'*

$$M_t \leqslant q^n/M \leqslant M_{s'}, \tag{3.11}$$

где $t = [(d-1)/2]$ *и* $M_k = 1 + n\lambda + \ldots + \binom{n}{k}\lambda^k$. *Кроме того, если в* (3.11) *достигается одно из равенств, то достигается также и другое, причем это происходит тогда и только тогда, когда код* C *является совершенным кодом с исправлением* t *ошибок. Необходимое условие существования такого кода состоит в том, что полином Ллойда* $Q_t(x)$ *должен иметь* t *различных нулей в множестве* $\{1, 2, \ldots, n\}$.

Доказательство. Докажем правое неравенство. Для любых $a \in C$, $e \in G$ положим $N(a, e) = 0$, если расстояние между a и e больше s', и $N(a, e) = 1$ в противном случае. По теореме 3.3

$$\sum_{a \in C} N(a, e) \geqslant 1 \quad \text{для любого} \quad e \in G. \tag{3.12}$$

С другой стороны, каждая точка удалена не более чем на s' в точности от $M_{s'}$ точек из G, так что

$$\sum_{e \in G} N(a, e) = M_{s'} \quad \text{для любого} \quad a \in G.$$

Поэтому, суммируя обе части неравенства (3.12) по всем $e \in G$, получаем искомый результат. Кроме того, правое равенство в (3.11) достигается тогда и только тогда, когда достигается равенство в (3.12) для всех $e \in G$, т. е. по определению тогда и только тогда, когда C образует совершенный код с исправлением s' ошибок. Остальная часть доказательства легко следует из теорем 2.3 и 2.6 (левое неравенство в (3.11) является границей Хэмминга); детали опускаются.

Из (3.11) также вытекает, что t не может превышать s'; это согласуется с теоремой 2.2.

4. ЧИСЛО РАССТОЯНИЙ И ОРТОГОНАЛЬНЫЕ СПИСКИ

Настоящий раздел посвящен исследованию дуального расстояния d', числа расстояний s и соотношений между ними. Мы будем пользоваться следующими обозначениями при $k = 0, 1, \ldots, n$:

$$v_k = \binom{n}{k}\lambda^k, \quad M_k = v_0 + v_1 + \ldots + v_k.$$

Основным инструментом исследования будут характеристические матрицы, введенные автором [7]; их определение мы сейчас напомним. Пусть C обозначает (n, M)-код над абелевой группой F порядка q и периода ν. При любом целом k, $0 \leqslant k \leqslant n$, *$k$-я характеристическая матрица* H_k кода C определяется так: строки нумеруются кодовыми словами, столбцы — элементами

веса k в G и элемент (a, h) матрицы H_k есть скалярное произведение $\langle a, h\rangle$ (ср. разд. 2.2), т. е.

$$H_k = [\langle a,\ h\rangle;\ a \in C,\ h \in Y_k]. \tag{4.1}$$

Таким образом, H_k — это $(M \times v_k)$-матрица над полем Γ_v комплексных корней v-й степени из единицы. В частности, H_0 — вектор j из всех единиц. Следующий разультат [7, теорема 5] очень полезен; он легко следует из теоремы 2.4.

Теорема 4.1. *Характеристические матрицы кода C и расстояние Хэмминга между кодовыми словами связаны соотношениями*

$$H_k \tilde{H}_k = [P_k(d_H(a,\ b));\ a,\ b \in C], \tag{4.2}$$

где $P_k(x)$ — полином Кравчука степени k, а $\tilde{H}_k$ — сопряженная транспонированная матрица H_k.

Удобно также ввести в рассмотрение $(M \times M_m)$-матрицу K_m над Γ_v, определенную с помощью характеристических матриц H_k (n, M)-кода:

$$K_m = [H_0,\ H_1,\ \ldots,\ H_m], \qquad 0 \leqslant m \leqslant n. \tag{4.3}$$

4.1. Число расстояний. *Расстояниями кода* называются ненулевые значения, действительно принимаемые расстоянием Хэмминга между кодовыми словами. Параметр s кода, определенный в разд. 2.3, равен числу различных расстояний. Для (n, M)-кода с расстояниями $d_1, d_2, \ldots, d_s$ определим *аннулирующий полином*

$$\alpha(x) = M(1 - x/d_1)(1 - x/d_2) \ldots (1 - x/d_s), \tag{4.4}$$

т. е. полином степени s, равный нулю в точках d_i и удовлетворяющий соотношению $\alpha(0) = M$. Интересно сравнить это определение с (3.9), особенно в случае аддитивных кодов.

Теорема 4.2. *Для любого (n, M)-кода C с s различными расстояниями справедливо матричное равенство*

$$K_s(\alpha_0 I_{v_0} \oplus \alpha_1 I_{v_1} \oplus \ldots \oplus \alpha_s I_{v_s})\,\tilde{K}_s = M I_M, \tag{4.5}$$

где матрица K_s определена в (4.3), *α_i — компоненты аннулирующего полинома в базисе полиномов Кравчука, I_t — единичная матрица порядка t и знак $\oplus$ обозначает прямую сумму. Кроме того, число кодовых слов ограничено величиной*

$$M \leqslant M_s = \sum_{i=0}^{s} \binom{n}{i} \lambda^i. \tag{4.6}$$

Доказательство. Для произвольного полинома $\alpha(x) = \alpha_0 P_0(x) + \ldots + \alpha_m P_m(x)$ степени $m \leqslant n$ по теореме 4.1 получаем

$$\sum_{k=0}^{m} \alpha_k H_k \tilde{H}_k = [\alpha(d_H(a, b)); \ a, \ b \in C]. \tag{4.7}$$

Если в качестве $\alpha(x)$ взять аннулирующий полином (4.4) кода, то правая часть в (4.7) будет равна MI_M, так как элемент (a, b) есть $\alpha(0)$ или $\alpha(d_i)$ при некотором i в зависимости от того, совпадают a и b или нет. Поскольку левые части в (4.5) и (4.7) при $m = s$ одинаковы, это доказывает первое утверждение.

Для упрощения обозначений введем диагональную матрицу Δ порядка M_s:

$$\Delta = \alpha_0 I_{v_0} \oplus \alpha_1 I_{v_1} \oplus \ldots \oplus \alpha_s I_{v_s}. \tag{4.8}$$

Мы доказали, что $K_s \Delta \tilde{K}_s = MI$. Следовательно, матрица $K_s \Delta \tilde{K}_s$ не вырождена и, так как K_s есть $(M \times M_s)$-матрица, можно написать

$$\min(M, M_s) \geqslant \text{ранг}(K_s) \geqslant \text{ранг}(K_s \Delta \tilde{K}_s) = M,$$

откуда вытекает вторая часть (4.6) теоремы. (Из приведенных выше неравенств также следует, что ранг матрицы K_s над Γ_ν принимает максимальное значение, а именно ранг $(K_s) = M$.)

Из неравенства (4.6) естественно возникает вопрос: что можно сказать о кодах с s расстояниями, для которых $M = M_s$? Таким кодам посвящен последний раздел статьи; здесь мы только приведем прямые следствия из теоремы 4.2. Связь с совершенными кодами очевидна. Действительно, применяя теорему 4.2 к дуальному коду C' данного аддитивного (n, M)-кода C, получаем, что $q^n/M = M_{s'}$, где s' — число расстояний кода C' или, что то же, внешнее расстояние кода C. Следовательно, по теореме 3.4 для кода C' граница (4.6) достигается тогда и только тогда, когда дуальный к нему код C является совершенным кодом с исправлением s' ошибок.

Напомним теперь определение *обобщенной матрицы Адамара*, введенное Батсоном [6]. При любом целом ν, $\nu \geqslant 2$, квадратная матрица K порядка M над полем комплексных корней ν-й степени из единицы называется обобщенной матрицей Адамара, если она удовлетворяет равенству $\tilde{K}K = MI$. При $\nu = 2$ это совпадает с классическим определением матрицы Адамара.

Теорема 4.3. *Пусть C обозначает (n, M)-код над абелевой группой периода ν, имеющий s расстояний и состоящий из $M = M_s$ кодовых слов. Тогда*

(1) *квадратная матрица K_s порядка M_s является обобщенной матрицей Адамара над полем комплексных корней ν-й степени из единицы*:

(2) *расстояния кода C являются нулями полинома Ллойда $Q_s(x)$ степени s* (см. (2.5)).

Доказательство. Пусть $d_1, d_2, \ldots, d_s$ — расстояния кода C и $\alpha(x)$ — аннулирующий полином (4.4). Равенство (4.5), где Δ и K_s — невырожденные квадратные матрицы порядка $M = M_s$, эквивалентно равенству

$$\Delta \tilde{K}_s K_s = MI. \tag{4.9}$$

С другой стороны, диагональные элементы матрицы $\tilde{K}_s K_s$, очевидно, равны M. Следовательно, в силу (4.9), должно быть $\Delta = I$ и $\tilde{K}_s K_s = MI$. Это доказывает утверждение (1) теоремы.

Чтобы доказать утверждение (2), заметим, что равенство $\Delta = I$ означает, что $\alpha_0 = \alpha_1 = \ldots = \alpha_s = 1$ в (4.8). Следовательно, аннулирующий полином $\alpha(x)$ равен $P_0(x) + \ldots + P_s(x)$, т. е., в силу определения (2.5), полиному Ллойда $Q_s(x)$. Теорема доказана.

Определение. *Обобщенным кодом Адамара* (ОА-кодом) *порядка s* называется q-ичный (n, M)-код с s расстояниями, для которого в (4.6) достигается равенство.

Это название выбрано для того, чтобы подчеркнуть связь с обобщенными матрицами Адамара (теорема 4.3(1)). Для иллюстрации этого определения исследуем кратко случай ОА-кодов первого порядка ($s = 1$), т. е. эквидистантных q-ичных кодов длины n, состоящих из $M_1 = 1 + n(q-1)$ слов. Полином Ллойда $Q_1(x)$ равен $M_1 - qx$. Следовательно, по теореме 4.3(2) единственное расстояние кода должно быть

$$d_1 = (1 + n(q-1))/q.$$

В двоичном случае ($q = 2$) ОА-коды первого порядка представляют собой хорошо известные эквидистантные $(n, n+1)$-коды Адамара с $d_1 = (n+1)/2$ (см. [3, стр. 324]). Из такого кода можно вывести матрицу Адамара порядка $n+1$, а именно ею будет матрица K_1 (см. теорему 4.3(1)). Обратно, любая матрица Адамара порядка $n+1$ дает двоичный код Адамара длины n.

В общем q-ичном случае единственные известные ОА-коды первого порядка — это регистровые коды максимальной длины (см. [3, стр. 323]). Они представляют собой линейные коды над $GF(q)$, где q — степень простого числа, а параметры их имеют вид

$$n = (q^k - 1)/(q-1), \quad M = q^k, \quad d_1 = q^{k-1}.$$

4.2. Ортогональные списки. Каждому (n, M)-коду над q-ичным алфавитом F поставим в соответствие $(n \times M)$-список, столбцами которого служат M различных n-последовательностей, образованных кодовыми словами. Этот список называется *ортогональным списком силы r и индекса* μ (см., например, [4]), если любые r его строк содержат все q^r упорядоченных r-последовательностей над F ровно по μ раз, при этом очевидно, что $M = \mu q^r$. Такой список обычно обозначается через $[M, n, q, r]$. Следует заметить, что это определение ортогональных списков, построенных из кодов, исключает возможность повторяющихся столбцов.

Прежде чем показать, что максимальную силу списка можно вычислить из распределения расстояний соответствующего кода (теорема 4.5), введем некоторые обозначения. При любом целом m, $0 \leqslant m \leqslant n$, рассмотрим m-последовательность $(\omega_1, \omega_2, \ldots, \omega_m)$ элементов $\omega_i \in F$ и m-последовательность $L = (i_1, i_2, \ldots, i_m)$ различных целых i_k, где $1 \leqslant i_k \leqslant n$. Для данного (n, M)-кода C над F определим $N_L(\omega_1, \ldots, \omega_m)$ как число кодовых слов a, удовлетворяющих условиям

$$a_{i_1} = \omega_1, \quad a_{i_2} = \omega_2, \quad \ldots, \quad a_{i_m} = \omega_m.$$

В соответствии с этими определениями максимальная сила $(M \times n)$-списка, соответствующего коду C, есть наибольшее из неотрицательных целых чисел m, для которых

$$N_L(\omega_1, \omega_2, \ldots, \omega_m) = M/q^m \tag{4.10}$$

при любом выборе компонент w_i и последовательности L.

Пусть $A(C)$ — распределение расстояний (2.10) кода C и $A'(C)$ — его преобразование Мак-Вильямс (2.2). Тогда дуальное расстояние d' кода C, в силу своего определения, должно удовлетворять условиям

$$A'_1(C) = \ldots = A'_{d'-1}(C) = 0, \quad A'_{d'}(C) \neq 0. \tag{4.11}$$

Теорема 4.4. *Пусть $H_0 = j$, H_1, ..., H_n — характеристические матрицы (n, M)-кода C. Тогда дуальным расстоянием d' кода C будет наименьшее из положительных целых чисел k, для которых $j^T H_k \neq 0$.*

Доказательство [7, следствие 7]. Умножая обе части равенства (4.2) слева на j^T и справа на j, сразу получаем

$$\| j^T H_k \|^2 = M A'_k(C),$$

где $A'(C)$ — преобразование Мак-Вильямс распределения расстояний и $\| \ \|$ обозначает эрмитову норму. Таким образом, теорема вытекает из определения (4.11) параметра d'.

Следующая теорема показывает комбинаторное значение параметра d'; отметим, что она сводится к хорошо известному результату для линейных кодов над некоторым полем.

Теорема 4.5. *Для любого* (n, M)*-кода* C *с дуальным расстоянием* d' *максимальная сила соответствующего* $(n \times M)$*-списка равна* $d' - 1$.

Доказательство. Пусть h — произвольный элемент веса k (k — целое, $0 \leqslant k \leqslant n$) в $G = F^n$, т. е. элемент множества Y_k, и L — произвольная m-последовательность (m — целое, $k \leqslant m \leqslant n$) различных целых чисел i_t, $1 \leqslant i_t \leqslant n$, обладающая тем свойством, что каждый индекс i, соответствующий ненулевой компоненте h_i элемента h, совпадает с одним из индексов i_t, т. е.

$$(i \neq i_1, i_2, \ldots, i_m) \Rightarrow (h_i = 0). \tag{4.12}$$

В соответствии с определением N_L и характеристической матрицы H_k [см. (4.1) и (2.6)] h-й элемент матрицы-строки $j^T H_k$ есть

$$(j^T H_k)(h) = \sum_{a \in C} \langle a, h \rangle = \sum_{\omega_i \in F} N_L(\omega_1, \ldots, \omega_m) \chi_{\omega_1}(h_{i_1}) \ldots \chi_{\omega_m}(h_{i_m}). \tag{4.13}$$

Сначала выберем k не превосходящим максимальную силу r и положим $m = k$. В силу (4.10) и хорошо известных свойств групповых характеров имеем

$$(j^T H_k)(h) = Mq^{-k} \sum_{\omega_i \in F} \chi_{\omega_1}(h_{i_1}) \ldots \chi_{\omega_m}(h_{i_m}) = M\delta_{k,0},$$
$$k = 0, 1, \ldots, r,$$

где δ — символ Кронекера. По теореме 4.4 отсюда вытекает, что $d' - 1 \geqslant r$.

Далее, предположим, что $d' - 1 > r$. Тогда в соответствии с теоремой 4.4 должны выполняться равенства

$$j^T H_k = M\delta_{k,0} j^T, \qquad k = 0, 1, \ldots, r + 1.$$

Поэтому для данной m-последовательности L, где $m = r + 1$, можно вывести из (4.13) систему q^m уравнений относительно q^m неизвестных $N_L(\omega_1, \ldots, \omega_m)$, $\omega_i \in F$,

$$\sum_{\omega_i \in F} N_L(\omega_1, \ldots, \omega_m) \chi_{\omega_1}(h_{i_1}) \ldots \chi_{\omega_m}(h_{i_m}) = M\delta_{h,0},$$

где h пробегает элементы группы G, удовлетворяющие (4.12). В силу хорошо известных свойств групповых характеров эта

система допускает единственное решение, а именно (4.10). Итак, ортогональный список должен иметь силу $m = r + 1$. Полученное противоречие показывает, что $d' - 1$ должно быть максимальной силой, и теорема доказана.

Для дальнейшего пользования сформулируем другое определяющее свойство дуального расстояния кода в терминах матриц K_m (см. (4.3)). Оно легко следует из теоремы 4.4 и приводится без доказательства.

Теорема 4.6. *Пусть C — произвольный (n, M)-код и m — наибольшее из целых чисел, $0 \leqslant m < n$, для которых $\check{K}_m K_m = MI$. Тогда дуальное расстояние кода C равно $2m + 2$ или $2m + 1$ в зависимости от того, является или нет матрица $\check{K}_m H_{m+1}$ нулевой.*

Следующая теорема устанавливает нижнюю границу числа слов в коде с данным дуальным расстоянием; в терминах ортогональных списков этот результат принадлежит Рао [19]. Указанную границу можно рассматривать как «дуальную» к границе сферической упаковки. Фактически для аддитивных кодов эти границы получаются одна из другой на основании дуальности таких кодов.

Теорема 4.7. *Для любого q-ичного (n, M)-кода с дуальным расстоянием d' (или, что то же, для любого ортогонального списка $[M, n, q, d' - 1]$ максимальной силы $d' - 1$)*

$$M \geqslant M_{t'}, \quad \text{где} \quad t' = [(d' - 1)/2]. \tag{4.14}$$

Доказательство. Пусть $A(C)$ — распределение расстояний данного кода C и $A'(C)$ — его преобразование Мак-Вильямс. По теоремам 2.1 и 2.6 $A'(C)/M$ есть положительная $(n + 1)$-последовательность. Следовательно, теорема 2.3 будет применима, если A заменить на $A'(C)/M$ и $d(A)$ на d'. Но тогда (4.14) следует из (2.4) и тождества $A'_0(C) + \ldots + A'_n(C) = q^n$ (см. теорему 2.1).

Замечание. С помощью подхода, основанного на линейном программировании [7], т. е. используя положительность распределения расстояний, можно, вообще говоря, улучшить границу Рао: для ортогональных списков $[M, n, q, r]$ справедливо неравенство $M \geqslant q^n/M'$, где M' — «верхняя граница линейного программирования» для числа кодовых слов в q-ичных кодах, имеющих длину n и минимальное расстояние $d \geqslant r + 1$. Это следует из теорем 2.1 и 4.5. Например, $M \geqslant 2^7$ для списков $[M, 13, 2, 4]$, и эта граница действительно достигается. Мы не бу-

дем далее углубляться в существо этого вопроса; нам просто хотелось обратить внимание читателя, интересующегося ортогональными списками, на границы линейного программирования.

Согласно теореме 4.6, $(M \times M_{t'})$-матрица $K_{t'}$, где $t' = [(d'-1)/2]$, удовлетворяет равенству

$$\tilde{K}_{t'} K_{t'} = MI. \tag{4.15}$$

Следовательно, у нее не может быть строк меньше, чем столбцов; это дает другое доказательство теоремы 4.7. Мы отложим рассмотрение кодов, достигающих границы (4.14), до п. 4.3.

Теперь кратко исследуем коды с достижимым максимальным расстоянием, введенные Синглтоном [21]. Учитывая границу Синглтона и применяя теорему 4.5, заключаем, что для любого q-ичного (n, M)-кода с минимальным расстоянием d и дуальным расстоянием d'

$$q^{d'-1} \leqslant M \leqslant q^{n-d+1}. \tag{4.16}$$

Кроме того, точно так же, как в случае линейных кодов (см., например, [8]), если достигается одно из равенств в (4.16), то достигается также и другое; эти результаты формально можно вывести из положительности распределения расстояний [7, теорема 15]. Код называется кодом *с достижимым максимальным расстоянием*, если он достигает границы Синглтона, т. е. если $M = q^{n-d+1}$. Предыдущее рассуждение показывает эквивалентность этого понятия и понятия ортогонального списка $[M, n, q, n-d+1]$ индекса один, которая была неявно отмечена Синглтоном [21].

4.3. Параметры s и d'. Наконец, соберем вместе теоремы 4.2, 4.3 и 4.7 и установим «дуальный» аналог теоремы 3.4.

Теорема 4.8. *Параметры s и d' произвольного (n, M)-кода C удовлетворяют неравенствам*

$$M_{t'} \leqslant M \leqslant M_s, \tag{4.17}$$

где $t' = [(d'-1)/2]$. Кроме того, если достигается одно из равенств в (4.17), то достигается также и другое, причем это происходит тогда и только тогда, когда C — обобщенный код Адамара порядка s. Необходимое условие существования такого кода состоит в том, что полином Ллойда $Q_s(x)$ должен иметь в множестве $\{1, 2, \ldots, n\}$ s различных нулей, а именно расстояний кода.

Доказательство. Отметим сначала, что неравенства (4.17) следуют из теорем 4.2 и 4.7. Это влечет $s \geqslant t'$, что

согласуется с теоремой 2.2, если ее применить к преобразованию Мак-Вильямс распределения расстояний.

Далее, предположим, что $M = M_s$, т. е. C есть OA-код порядка s. По теореме 4.3 квадратная матрица K_s является обобщенной матрицей Адамара, так что (см. теорему 4.6) s не может превышать $t' = [(d' - 1)/2]$. Поэтому должно быть $s = t'$ и $M = M_{t'}$.

Наконец, предположим, что $M = M_{t'}$. Тогда (см. (4.15)) квадратная матрица $K_{t'}$ является обобщенной матрицей Адамара, так что в силу (4.3) можно написать

$$\sum_{k=0}^{t'} H_k \widetilde{H}_k = K_{t'} \widetilde{K}_{t'} = MI.$$

В соответствии с (4.7), где $m = t'$ и $\alpha_0 = \alpha_1 = \ldots \alpha_m = 1$, это очевидно означает, что полином

$$\alpha(x) = \sum_{k=0}^{t'} P_k(x) = Q_{t'}(x)$$

обращается в нуль при $x = d_1, d_2, \ldots, d_s$, где d_i — расстояния кода C. Так как $\alpha(x)$ имеет степень t', а t' не может превышать s, то $t' = s$ и потому $M = M_s$. С помощью этого рассуждения мы также переоткрыли «условие Ллойда» для OA-кодов, приведенное еще в теореме 4.3(2). Доказательство закончено.

Замечание. Пусть $\alpha(x)$ — аннулирующий полином (4.4) некоторого кода. Теорема 4.8 показывает, что компоненты α_0, $\alpha_1, \ldots, \alpha_s$ полинома $\alpha(x)$ в базисе полиномов Кравчука равны 1, когда дуальное расстояние равно $d' = 2s + 1$, т. е. для OA-кодов. В общем случае при $s + 1 \leqslant d' \leqslant 2s + 1$ можно показать, что

$$\alpha_0 = \alpha_1 = \ldots = \alpha_{d'-1-s} = 1.$$

Это следует из теорем 4.2 и 4.6; подробное доказательство мы не приводим.

5. КОМБИНАТОРНЫЕ СВОЙСТВА КОДОВ

В настоящем разделе нас будут интересовать два свойства кода, выражающие некоторую «симметрию»: метрическая инвариантность и существование обобщенных t-блоков. Наше определение блоков из кодов подобно определению Ассмуса и Мэттсона [1]; в двоичном случае эти определения фактически совпадают.

Код C называется *метрически инвариантным*, если число кодовых слов, удаленных на расстояние i от фиксированного кодового слова, зависит только от i и не зависит от выбранного

слова. По определению это число равно i-й компоненте $A_i(C)$ распределения расстояний. Начиная с этого места, будем предполагать, что все коды содержат нулевую n-последовательность над F; это не будет существенным ограничением, так как любой код можно преобразовать в код, обладающий этим свойством, применяя подходящий сдвиг группы $G = F^n$, который не меняет расстояния между кодовыми словами. Таким образом, для кодов, метрически инвариантных, распределение расстояний сводится к классическому *распределению весов*.

Говорят, что элемент $a \in G$ *покрывается* элементом $b \in G$, если каждая ненулевая компонента a_i элемента a равна соответствующей компоненте b_i элемента b; это будем записывать в виде $a \leqslant b$. Пусть, например, $n = 5$, $F = \{0, 1, 2\}$. Тогда $a = (0, 0, 1, 1, 2)$ покрывается элементом $b = (1, 0, 1, 1, 2)$.

Подмножество S группы G будем называть *t-блоком типа* λ ($\lambda = q - 1$) с параметрами $(\mu_t; n, w, t)$, $0 \leqslant t \leqslant w \leqslant n$, $\mu_t \geqslant 1$, если оно удовлетворяет двум условиям: (1) все его элементы имеют один и тот же вес w, (2) каждый элемент веса t в G покрывается одним и тем же числом μ_t элементов из S. Легко убедиться в том, что t-блок, где $t \geqslant 1$, является $(t-1)$-блоком $(\mu_{t-1}; n, w, t-1)$, где

$$(w - t + 1)\,\mu_{t-1} = \lambda\,(n - t + 1)\,\mu_t. \qquad (5.1)$$

В частности, μ_0 есть мощность множества S. В двоичном случае ($q = 2$, $\lambda = 1$) это определение эквивалентно классическому определению t-блоков без повторяющихся блоков. При $\lambda \geqslant 2$ любой t-блок S типа λ переходит в t-блок S^* типа 1 при отображении $F \to Z_2$, заданном соответствием

$$u \to u^* = 0 \text{ или } 1 \text{ в зависимости от того, } u = 0 \text{ или } u \neq 0.$$

В действительности это отображение сохраняет вес и отношение покрытия. Следует отметить, что S^* может иметь повторяющиеся блоки с различными кратностями.

Весами кода, содержащего нулевую n-последовательность, называются значения, принимаемые функцией w_H на ненулевых кодовых словах. Очевидно, что каждый вес является также расстоянием (обратное справедливо для кодов, метрически инвариантных). В этом разделе мы рассмотрим свойства множеств кодовых слов фиксированного веса образовывать t-блоки. Используются два метода; первый основан на теореме 4.5, второй — на теореме 3.2.

5.1. Основной подход. Для данного кода C длины n и элемента e из $G = F^n$ обозначим через $\mu(i, e)$ число кодовых слов веса i, покрывающих e (ясно, что $\mu(i, e) = 0$ при $i < w_H(e)$).

Теорема 5.1. *Пусть C будет q-ичным (n, M)-кодом с дуальным расстоянием d', а e — элемент из G, вес которого $w_H(e) = t$ не превышает $d' - 1$. Тогда при $k = 0, 1, \ldots, d' - 1 - t$ числа $\mu(i, e)$ удовлетворяют равенствам*

$$\sum_{i=t}^{n} \binom{i-t}{k} \mu(i, e) = q^{-(t+k)} M \binom{n-t}{k} \lambda^k. \tag{5.2}$$

Доказательство. Левая часть в (5.2) равна числу пар (a, g), $a \in C$, $g \in G$, где $e \leqslant g \leqslant a$ и $w_H(g) = t + k$. При $t + k \leqslant d' - 1$ из теоремы 4.5 вытекает, что число кодовых слов, покрывающих элемент g веса $t + k$, равно M/q^{t+k}. Так как число таких элементов g, покрывающих данный элемент e веса t, равно $\binom{n-t}{k} \lambda^k$, отсюда получается (5.2).

Следствие 5.2. *В тех же предположениях, что и выше,*

$$\sum_{i=t}^{n} \left(\mu(i, e) - q^{-n} M \binom{n-t}{i-t} \lambda^{i-t} \right) \beta(i) = 0 \tag{5.3}$$

для любого полинома $\beta(x)$ над полем рациональных чисел, степень которого не превышает $d' - 1 - t$.

Доказательство. При $\beta(x) = \binom{x-t}{k}$ формула (5.3) вытекает из (5.2) в силу хорошо известного тождества

$$\sum_{j=0}^{m} \binom{m}{j} \binom{j}{k} \lambda^j = \lambda^k q^{m-k} \binom{m}{k},$$

где $m = n - t$. Таким образом, утверждение доказано, так как полиномы $\binom{x-t}{k}$, $0 \leqslant k \leqslant d' - 1 - t$, образуют базис векторного пространства рациональных полиномов $\beta(x)$, степень которых не превышает $d' - 1 - t$.

Следующая теорема аналогична результату Ассмуса и Мэттсона [1] для линейных кодов.

Теорема 5.3. *Пусть C будет q-ичным кодом с дуальным расстоянием d', а t — такое целое число, $1 \leqslant t \leqslant d'$, что число весов кода C, не меньших t, само не более $d' - t$. Тогда любое множество кодовых слов фиксированного веса, не меньшего t, образует t-блок типа $\lambda = q - 1$.*

Доказательство. Пусть e — элемент веса t из G и J_t — множество весов кода C, не меньших t. Тогда $\mu(i, e) = 0$, если i

не принадлежит J_t. Ранг системы (5.2) при $0 \leqslant k \leqslant d' - 1 - t$ с неизвестными $\mu(i, e)$, $i \in J_t$, равен $|J_t|$. Действительно, квадратная матрица

$$\left[\binom{i-t}{k};\ i \in J_t,\ 0 \leqslant k \leqslant |J_t| - 1\right]$$

не вырождена. Поэтому решение $(\mu(i, e),\ i \in J_t)$ единственно и не зависит от конкретного элемента e веса t. Следовательно, при каждом $i \in J_t$ множество кодовых слов веса i образует t-блок типа λ с $\mu_t = \mu(i, e)$. Теорема доказана.

Теорема 5.4. *Код C, у которого число s расстояний не превышает дуального расстояния d', метрически инвариантен.*

Доказательство. При $a \in C$ определим код $C_a = \{b - a \mid b \in C\}$, полученный «сдвигом» кода C. Ясно, что все коды C_a имеют то же распределение расстояний, что и код C, и, значит, то же дуальное расстояние d'. Рассмотрим равенство (5.2) для кода C_a при $e = 0$. Так как каждый вес кода C_a есть расстояние кода C, то при $k = 0, 1, \ldots, d' - 1$ можно написать

$$\sum_{i \in J} \binom{i}{k} \mu(i,\ 0) = q^{-k} M \binom{n}{k} \lambda^k - \delta_{k,0}, \tag{5.4}$$

где J — множество расстояний кода C. Согласно условию теоремы, мощность s множества J не превышает d'. Следовательно, в силу тех же рассуждений, что и в теореме 5.3, система (5.4) допускает единственное решение $(\mu(i, 0), i \in J)$, не зависящее от a. Так как $\mu(i, 0)$ для кода C_a равно числу кодовых слов кода C, удаленных на расстояние i от данного $a \in C$, это означает, что код C метрически инвариантен.

Пример. Для положительного целого m, где $m \equiv 0 \pmod 4$, рассмотрим двоичный $(m^2 - 1, m^4/2)$-код C, имеющий три $(s = 3)$ расстояния, а именно

$$d_1 = m(m-1)/2, \quad d_2 = m^2/2, \quad d_3 = m(m+1)/2. \tag{5.5}$$

Кердок [10] недавно построил такой код при любом m, являющемся степенью числа 2. В общем случае можно показать [7, теорема 12], что C — максимальный код в классе кодов длины $m^2 - 1$, имеющих только расстояния (5.5), и что его распределение расстояний определяется формулами

$$N_1 = m(m+1)(m^2-2)/4, \quad N_2 = m^2 - 1, \quad N_3 = m(m-1)(m^2-2)/4,$$

где $N_i = A_{di}(C)$. При этом оказывается, что компоненты $A'_1(C), \ldots, A'_4(C)$ преобразования Мак-Вильямс последовательности $A(C)$ равны нулю и $A'_5(C) > 0$; поэтому дуальное рас-

стояние d' кода C равно пяти. Следовательно, по теореме 5.4 код C метрически инвариантен. Кроме того, по теореме 5.3 при $t=2$ кодовые слова веса d_i образуют 2-блок при $i=1, 2, 3$. Так как такие коды действительно существуют при $m=2^r$, $r\geqslant 2$, а именно это коды Кердока, мы указали три бесконечных семейства 2-блоков. Блоки, соответствующие значению d_2, являются дополнениями хорошо известных блоков Адамара, в то время как блоки, соответствующие d_1 и d_3, представляются новыми.

5.2. Дуальный подход. При данном двоичном коде C длины n и точке e из $G=Z_2^n$ обозначим через $B_i^{(k)}(e)$ число кодовых слов веса k, удаленных на расстояние i от e. Очевидно, что $B_i^{(k)}(e)\neq 0$ только при $w_H(e)\geqslant k-i$ и $w_H(e)\equiv k-i \pmod 2$. С другой стороны,

$$\sum_{k=0}^{n} B_i^{(k)}(e)=B_i(e), \tag{5.6}$$

где $B_i(e)$ — число кодовых слов, удаленных на расстояние i от e (см. (3.1)). Теорема 5.6 показывает, как «решать» систему (5.6). Нам понадобится

Лемма 5.5. *Пусть C — двоичный код, e — элемент веса $t\geqslant 1$ в G и E — множество элементов веса $t-1$ в G, покрываемых e. Тогда при любых i и k параметры $B_i^{(k)}$ кода C удовлетворяют соотношению*

$$\sum_{g\in E} B_{i-1}^{(k)}(g)=\tfrac{1}{2}(t+k-i+2)B_{i-2}^{(k)}(e)+\tfrac{1}{2}(t-k+i)B_i^{(k)}(e).$$

Доказательство. Левая часть есть число пар (a, g), где $a\in C$, $g\in E$, $w_H(a)=k$, $d_H(a, g)=i-1$. Пусть a — кодовое слово веса k, удаленное на расстояние j от данного e. Тогда легко видеть, что число элементов g в E, удаленных на расстояние $i-1$ от a, равно $(t-k+i)/2$, когда $j=i$, равно $(t+k-i+2)/2$, когда $j=i-2$, и равно нулю при остальных значениях j. Лемма доказана.

Теорема 5.6. *При заданных целых i и k параметр $B_i^{(k)}(e)$ любого двоичного кода однозначно определяется весом*[1]) *элемента e, распределением весов кода и значениями параметров $B_j(g)$ при $g\leqslant e$.*

---

[1]) Это утверждение доказывается и затем используется при дополнительном ограничении, что вес элемента e не превосходит k. — *Прим. перев.*

Доказательство. Теорема тривиальна при $k = 0$, а также при $e = 0$. Предположим, что она справедлива при $k = 0, 1, \ldots, r-1$, а также при $k = r$ для любого e, вес которого меньше данного числа t, где $t \leqslant r$. Основываясь на этих предположениях индукции, докажем результат при $k = r$ и при любом элементе e веса t. Очевидно, что $B_{r-t}^{(m)}(e) = 0$ при $m > r$, так что (5.6) можно записать в виде

$$B_{r-t}^{(r)}(e) = B_{r-t}(e) - \sum_{m=0}^{r-1} B_{r-t}^{(m)}(e).$$

Это доказывает теорему при $i = r - t$ (индукцией по k). Далее, значения $B_i^{(r)}(e)$ выражаются с помощью леммы 5.5 последовательно при $i = r - t + 2, r - t + 4, \ldots$ через числа $B_j^{(r)}(g)$, где $g \in E$. Поскольку элементы из E имеют вес $t - 1$, теорема вытекает из предположения индукции относительно веса элемента e.

Теперь мы подготовлены к тому, чтобы доказать основной результат этого раздела; интересно сравнить его с теоремами 5.3 и 5.4. Для линейных кодов утверждение о t-блоках первоначально было получено Ассмусом и Мэттсоном [1].

Теорема 5.7. *Пусть C — двоичный код с минимальным расстоянием d и внешним расстоянием s', причем $d \geqslant s'$. Тогда код C метрически инвариантен. Кроме того, кодовые слова фиксированного веса образуют t-блок (типа 1), где $t = d - s'$.*

Доказательство. Пусть a — кодовое слово. Так как s' не превышает d, то

$$B_0(a) = 1, \quad B_i(a) = 0 \quad \text{при} \quad i = 1, \ldots, s' - 1.$$

Поэтому в силу теоремы 3.2 строка $B(a) = (B_0(a), \ldots, B_n(a))$ матрицы расстояний B не зависит от a, и первое утверждение теоремы доказано.

Рассмотрим элемент e в G данного веса $m \leqslant d - s'$. Очевидно, что единственное кодовое слово, которое может находиться на расстоянии менее s' от e, есть нулевое слово; это может произойти, если m само меньше s'. Во всяком случае s'-последовательность $(B_0(e), \ldots, B_{s'-1}(e))$ не зависит от e. Отсюда по теореме 3.2 полная строка $B(e)$ сама не зависит от e, так как $w_H(e) = m$. Следовательно, по теореме 5.6 при заданных i и k параметр $B_i^{(k)}(e)$ принимает постоянное значение при любом e веса $t = d - s'$. С другой стороны, при $k \geqslant t$ параметр $B_{k-t}^{(k)}(e)$ есть число кодовых слов веса k, покрывающих e. Итак, доказано, что если положить

$$\mu_t = B_{k-t}^{(k)}(e) \quad \text{при} \quad w_H(e) = t = d - s',$$

то кодовые слова веса k образуют t-блок $(\mu_t; n, k, t)$.

Пример 1. Рассмотрим удлиненный квадратично-вычетный $(48,2^{24})$-код (см. п. 3.1). Так как минимальное расстояние d равно 12, а внешнее расстояние s' равно 8, то кодовые слова фиксированного веса по теореме 5.7 образуют 4-блок. Кроме того, мы убедились, что строка $B(e)$ матрицы расстояний зависит только от веса e, когда этот вес не более пяти. Поэтому на основании тех же рассуждений, что и в теореме 5.7, кодовые слова фиксированного веса образуют 5-блок. Первоначально это было доказано в [1]. Отметим общий результат относительно кодов, подобных квадратично-вычетному коду длины 48:

Теорема 5.8. *Пусть C — двоичный $(48,2^{24})$-код, расстояния которого принадлежат множеству* $\{12, 16, 20, 24, 28, 32, 36, 48\}$. *Тогда C имеет такое же распределение расстояний, как удлиненный квадратично-вычетный код. Кроме того, код C метрически инвариантен и кодовые слова фиксированного веса образуют 5-блок.*

Доказательство. Единственность распределения расстояний кода C можно доказать с помощью теоремы 12 из [7]. Тогда утверждение о метрической инвариантности и 5-блоках следует из теоремы 5.7 и приведенного выше замечания. Действительно, строка $B(e)$ матрицы расстояний зависит только от веса e при $w_H(e) \leqslant 5$ для любого кода, имеющего то же распределение расстояний, что и удлиненный квадратично-вычетный код.

Пример 2. Пусть C — код Препараты длины $n = 4^r - 1$ с минимальным расстоянием $d = 5$ (см. [18]). Вычисляя распределение расстояний кода C, Гёталс и Сновер [9] явно показали, что внешнее расстояние кода C равно $s' = 3$. В таком случае по теореме 5.7 кодовые слова каждого веса образуют 2-блоки. Это было первоначально установлено Семаковым, Зиновьевым и Зайцевым [20].

Часть теоремы 5.7 можно перенести на недвоичные коды. Доказательство очень похоже, и мы его не приводим.

Теорема 5.9. *Пусть C будет q-ичным кодом с минимальным расстоянием d и внешним расстоянием s', причем $d \geqslant s'$. Тогда код C метрически инвариантен. Кроме того, кодовые слова веса d образуют t-блок типа $\lambda = q - 1$, где $t = d - s'$.*

Причина, по которой t-блоки типа λ в общем случае нельзя выявить для весов, больших d, состоит в том, что лемму 5.5 и теорему 5.6 нельзя перенести на недвоичный случай.

6. ОБОБЩЕННЫЕ КОДЫ АДАМАРА

ОА-код порядка s определен в п. 4.1 как код с s расстояниями, где $1 \leqslant s \leqslant n-1$, для которого число кодовых слов принимает максимальное значение $M = M_s$. Как и в предыдущем разделе, будем предполагать, что все коды содержат нулевую n-последовательность.

Очевидным примером служит двоичный код C длины $n = 2s+1$, словами которого являются 2^{n-1} n-последовательностей, имеющих четный вес. Основные параметры кода C таковы: $d = 2$, $s = (n-1)/2$, $d' = n$, $s' = 1$. Легко показать, что любой двоичный ОА-код порядка s с $n = 2s+1$ совпадает с C, если он содержит нулевое слово.

Как мы убедились в разд. 4, аддитивный ОА-код есть не что иное, как код, дуальный к совершенному аддитивному коду. Поэтому из недавних результатов по совершенным кодам [11, 24] следует, что при $s \geqslant 2$ существуют только два аддитивных ОА-кода, кроме указанного выше класса, а именно коды Голея. Единственность их (в классе аддитивных кодов) доказана Плесс [17].

В этом разделе мы дадим очень сильные необходимые условия для произвольных ОА-кодов (теоремы 6.1 и 6.4); они делают существование новых ОА-кодов при $s \geqslant 2$ маловероятным.

Теорема 6.1. *Пусть C будет q-ичным ОА-кодом порядка s. Тогда*

(1) *дуальное расстояние d' кода C равно $2s+1$ и M_s делится на q^{2s};*

(2) *код C метрически инвариантен и кодовые слова фиксированного веса, не меньшего t, образуют t-блок типа λ при любом $t \leqslant s+1$.*

Доказательство. По теореме 4.8 дуальное расстояние d' должно быть равно $2s+1$ или $2s+2$. В силу теоремы 4.6 при $d' = 2s+2$ было бы $\check{K}_s H_{s+1} = 0$, что невозможно, так как матрица K_s не вырождена. Таким образом, $d' = 2s+1$, откуда следует (см. теорему 4.5), что число кодовых слов $M = M_s$ делится на q^{2s}. Утверждение (2) вытекает из теорем 5.3 и 5.4.

Замечания. (1) Интересно сравнить условия делимости $q^{2s} | M_s$ для ОА-кодов и $M_t | q^n$ для совершенных кодов с исправлением t ошибок.

(2) Можно показать с помощью теорем 4.8 и 6.1 (2), что минимальное расстояние ОА-кода порядка s равно по крайней мере $2s$, за исключением приведенного выше двоичного $(2s+1, 4^s)$-кода с $d = 2$.

В силу теоремы 6.1 распределение расстояний OA-кода сводится к его распределению весов, для которого мы получим теперь явную формулу. Сначала нам понадобятся некоторые свойства полиномов Ллойда (2.5).

Лемма 6.2. *При заданных n и λ полиномы Ллойда $Q_0(x)$, $Q_1(x)$, ..., $Q_{n-1}(x)$ удовлетворяют условиям ортогональности* $(q = \lambda + 1)$:

$$\sum_{k=1}^{n} Q_i(k) Q_j(k) \binom{n-1}{k-1} \lambda^{k-1} = q^{n-1} \binom{n-1}{i} \lambda^i \delta_{i\ j}. \tag{6.1}$$

Доказательство. Пусть $P_k^{(n)}(x)$ обозначает полином Кравчука (2.1) степени k, а $Q_t^{(n)}(x)$ — полином Ллойда (2.5) степени t. Можно показать [13, стр. 110], что

$$Q_k^{(n)}(x) = P_k^{(n-1)}(x-1). \tag{6.2}$$

Поэтому (6.1) сводится к определяющим отношениям полиномов Кравчука (см. [23, стр. 48]), и лемма доказана.

Пусть s — целое число, $1 \leqslant s \leqslant n-1$. По лемме 6.2 и хорошо известному результату относительно ортогональности полиномов полином $Q_s(x)$ имеет s различных действительных нулей $d_1, d_2, \ldots, d_s$, где $1 \leqslant d_k \leqslant n$. Каждому d_k поставим в соответствие *число Кристоффеля* $w_{k,s}$, определенное (см. [23, стр. 61]) формулой

$$w_{k,s}^{-1} = \sum_{j=0}^{s-1} \left[(Q_j(d_k))^2 / q^{n-1} \binom{n-1}{j} \lambda^j \right]. \tag{6.3}$$

Лемма 6.3. *Пусть $\beta(x)$ пробегает полиномы степени не выше $2s-1$. Тогда s-последовательность $(w_{1,s}, w_{2,s}, \ldots, w_{s,s})$ чисел Кристоффеля является единственным решением множества линейных уравнений*

$$\sum_{k=1}^{s} w_{k,s} \beta(d_k) = \sum_{i=1}^{n} \binom{n-1}{i-1} \lambda^{i-1} \beta(i).$$

Доказательство. Это следует из леммы 6.2 в силу хорошо известного свойства чисел Кристоффеля (см. [23, теорема 3.4.1]).

Теорема 6.4. *Пусть $d_1, d_2, \ldots, d_s$ — веса OA-кода порядка s. Тогда числа d_k совпадают с нулями полинома Ллойда*

$Q_s(x)$. *Кроме того, число* N_k *кодовых слов веса* d_k *удовлетворяет равенству*

$$n\lambda M_s/qd_kN_k = \sum_{j=0}^{s-1}\left[(Q_j(d_k))^2/\binom{n-1}{j}\lambda^j\right]. \tag{6.4}$$

Доказательство. Первое утверждение вытекает из теоремы 4.3(2). Далее, используя следствие 5.2 при $t=1$ и теорему 6.1(1), для произвольного полинома $\beta(x)$ степени не выше $2s-1$ находим

$$\sum_{k=1}^{s}\mu(d_k, e)\beta(d_k) = q^{-n}M_s\sum_{i=1}^{n}\binom{n-1}{i-1}\lambda^{i-1}\beta(i),$$

где $\mu(d_k, e)$ — число кодовых слов веса d_k, покрывающих данный элемент $e \in G$ веса 1. Сравнивая это с леммой 6.3, заключаем, что

$$\mu(d_k, e) = q^{-n}M_s w_{k,s}. \tag{6.5}$$

С другой стороны, ясно, что параметры 1-блока, образованного кодовыми словами веса d_k, удовлетворяют равенству

$$d_kN_k = \lambda n\mu(d_k, e) \tag{6.6}$$

(см. (5.1) при $w = d_k$, $t=1$, $\mu_1 = \mu(d_k, e)$, $\mu_0 = N_k$). Искомое равенство (6.4) получается из (6.3), (6.5) и (6.6) с помощью элементарных вычислений.

Чтобы проиллюстрировать теорию, докажем, что, кроме двух хорошо известных двоичных кодов, не существует ОА-кодов третьего порядка над некоторым полем.

Теорема 6.5. *Для алфавитов, порядки которых являются степенями простых чисел, существуют только два неэквивалентных* ОА*-кода третьего порядка*: $(7,2^6)$*-код, образованный всеми двоичными 7-последовательностями четного веса, и* $(23,2^{11})$*-код Голея.*

Доказательство. Пусть C будет q-ичным ОА-кодом третьего порядка длины n. Из теорем 6.1(1) и 6.4 имеем два необходимых условия: у полинома Ллойда $Q_3(x)$ должны быть три различных нуля в множестве $\{1, 2, \ldots, n\}$ и q^6 должно делить M_3. В своем доказательстве несуществования совершенных кодов с исправлением трех ошибок ван Линт [12] показал, что эти условия одновременно выполняться не могут, если q является степенью простого числа, $q \neq 2$.

Таким образом, можно ограничиться двоичным случаем ($q = 2$, $\lambda = 1$). Из (2.1) и (6.2) легко вывести, что при $0 \leqslant s \leqslant n - 1$

$$Q_s(n + 1 - x) = (-1)^s Q_s(x).$$

Поэтому $(n + 1)/2$ — нуль полинома $Q_s(x)$, если s нечетно. Тогда при $s = 3$ находим по теореме 6.4, что $d_2 = (n + 1)/2$ есть один из трех весов. С помощью прямого подсчета из (6.4) выводим формулу для числа N_2 кодовых слов веса d_2:

$$3N_2 = n(n - 2)(n^2 - n + 6)/(3n - 5). \qquad (6.7)$$

С другой стороны, по теореме 6.1(2) эти кодовые слова образуют 4-блок. Параметры μ_i этого блока можно вычислить из (6.7) и (5.1), где $\mu_0 = N_2$; при этом получается, что

$$9(\mu_0 - 8\mu_3) = 3n + 2 + 64/(3n - 5).$$

Так как левая часть есть целое число, то $3n - 5$ должно делить 64. Единственными значениями n ($n \geqslant 7$), удовлетворяющими этому условию, будут $n = 7$ и $n = 23$. Двоичные ОА-коды с этими двумя длинами, как известно, существуют, а именно это коды, упомянутые в теореме. Ясно, что при $n = 7$ указанный код единствен. Из теоремы Сновера [22] об единственности кода Голея вытекает, что ОА-код длины 23 также единствен (сообщено Сновером). Теорема доказана.

БЛАГОДАРНОСТИ

Наибольшую признательность автор выражает Дж. М. Гёталсу за многочисленные беседы и полезные предложения. Автор благодарит также госпожу Ф. Дж. Мак-Вильямс, чьи замечания позволили улучшить разд. 3.

СПИСОК ЛИТЕРАТУРЫ

1. Assmus E. F., Jr., Mattson H. F., Jr., New 5-designs, *J. Comb. Theory*, **6** (1969), 122.
2. Assmus E. F., Jr., Mattson H. F., Jr., Algebraic theory of codes, II, Air Force Cambridge Res. Lab. Final Report, 1970.
3. Берлекэмп Э., Алгебраическая теория кодирования, изд-во «Мир», М., 1971.
4. Bose R. C., Bush K. A., Orthogonal arrays of strength two an three, *Ann. Math. Stat.*, **23** (1952), 508.
5. Bose R. C., Mesner D. M., On linear associative algebras corresponding to association schemes of partially balanced designs, *Ann. Math. Stat.*, **30** (1959), 21.
6. Butson A. T., Generalized Hadamard matrices, *Proc. Amer. Math. Soc.*, **13** (1962), 894.
7. Delsarte P., Bounds for unrestricted codes by linear programming, *Philips Res. Repts.*, **27** (1972), 272.

8. Goethals J. M., A polynomial approach to linear codes, *Philips Res. Repts.*, **24** (1969), 145.
9. Goethals J. M., Snover S. L., Nearly perfect binary codes, *Discrete Math.*, **3** (1972), 65.
10. Kerdock A. M., A class of low-rate nonlinear binary codes, *Inform. and Control*, **20** (1972), 182. (Русский перевод: Кердок А. М., Класс нелинейных двоичных кодов с низкой скоростью передачи, Кибернетический сборник (нов. серия), вып. 10, изд-во «Мир», М., 1973, стр. 33—38.)
11. Lenstra H. W., Jr., Two theorems on perfect codes, *Discrete Math.*, **3** (1972), 125.
12. Van Lint J. H., On the nonexistence of perfect 2- and 3-Hamming-error-correcting codes over $GF(q)$, *Inform. and Control*, **16** (1970), 396.
13. Van Lint J. H., Coding theory, в сб. «Lecture notes in Mathematics», Springer-Verlag, Berlin, 1971.
14. Mac Williams F. J., Ph. D. Dissertation, Harvard University, 1961 (не опубликовано).
15. Mac Williams F. J., A theorem on the distribution of weights in a systematic code, *BSTJ*, **42** (1963), 79.
16. Pless V., Power moment identities on weight distributions in error-correcting codes, *Inform. and Control*, **6** (1963), 147.
17. Pless V., On the uniqueness of the Golay codes, *J. Comb. Theory*, **5** (1968), 215.
18. Preparata F. P., A class of optimum nonlinear double-error-correcting codes, *Inform. and Control*, **13** (1968), 378. (Русский перевод: Препарата Ф. П., Класс оптимальных нелинейных кодов с исправлением двойных ошибок, Кибернетический сборник (нов. серия), вып. 7, изд-во «Мир», М., 1970, стр. 18—42.)
19. Rao C. R., Factorial experiments derivable from combinatorial arrangements of arrays, *Jour. Royal Stat. Soc.*, **9** (1947), 128.
20. Семаков Н. В., Зиновьев В. А., Зайцев Г. В., Равномерно упакованные коды, *Проблемы передачи информации*, **7**, вып. 1, 1971, стр. 38—50.
21. Singleton R. C., Maximum distance q-ary codes, *IEEE Trans. on Inform. Theory*, **10** (1964), 116.
22. Snover S. L., Ph. D. Dissertation, Michigan State University, 1973.
23. Г. Серё, Ортогональные многочлены, ГИФМЛ, М., 1962.
24. Tietäväinen A., On the nonexistence of perfect codes over finite fields, *SIAM J. Appl. Math.*, **24** (1973), 88.

Улучшенный метод частичного перекрытия для умножения, выполняемого в темпе поступления информации [1])

Майкл С. Патерсон, Майкл Дж. Фишер, Альберт Р. Мейер

Приводится нижняя оценка вида $cN \log N$ для временной сложности многоленточной машины Тьюринга, выполняющей умножение в темпе поступления информации N-разрядных бинарных целых чисел. Для более общего класса, включающего некоторые виды машин со случайным доступом, соответствующая оценка есть $cN \log N/\log\log N$. Эти оценки хорошо сравнимы с известными верхними оценками $cN(\log N)^k$, а для отдельных классов верхние и нижние оценки совпадают. Доказательства этого основываются на методе «частичного перекрытия», предложенном Куком и Андерой.

1. ВВЕДЕНИЕ

Актуальная проблема в области сложности вычислений состоит в получении нижних оценок времени вычисления для естественных алгоритмов, выполняемых достаточно мощной машиной. Для обычной машины, задача которой состоит в отображении входной последовательности в выходную, тривиальная нижняя оценка для многих отображений — это число шагов, необходимых для прочтения входной последовательности. Существует много комбинаторных приемов, включающих, например, пересекающиеся последовательности (ср. [6, § 10.4]), которые позволяют получать нетривиальные нижние оценки, но лишь для элементарных машин Тьюринга с одной лентой. Мощная техника диагонализации употребляется только для отображений, позволяющих кодировать вычисления машины.

В этой статье излагается и развивается далее метод «частичного перекрытия», введенный Куком и Андерой [3], который устанавливает для очень большого класса машин нелинейную нижнюю оценку времени, необходимого для умножения бинарных целых чисел. (Аналогичный метод недавно был вновь ис-

---

[1]) Paterson M. S., Fischer M. J., Meyer A. R., An improved overlap argument for on-line multiplication, *SIAM-AMS Proc.*, **7** (1974), 97—111. Большая часть этой работы выполнена в Уорикском университете при частичной поддержке Science Research Council of the United Kingdom. Кроме того, работа финансировалась следующими фондами: the National Science Foundation under research grant GJ-34671 to MIT Project MAC; the Artifical Intelligence Laboratory, an MIT research program sponsored by the Advanced Research Projects Agency, Department of Defense, under Office of Naval Research contract number N00014-70-A-0362-0003.

пользован Андерой [1].) Наш вклад по отношению к [3] состоит, во-первых, в том, что основная линия доказательства несколько короче и проще, во-вторых, нижняя оценка, приводимая в [3], увеличена на множитель $\log\log n$ и, как показано, она справедлива и для среднего и для максимального времени, в-третьих, намечен новый путь исследований, который позволит получить даже более сильный результат для класса многоленточных машин Тьюринга. Мы покажем на подходящих алгоритмах умножения, что в некоторых случаях новые результаты будут оптимальными с точностью до постоянного коэффициента.

Наш главный результат — нижняя оценка для умножения, производимого в темпе поступления информации. Будем говорить, что отображение входной последовательности символов в выходную осуществляется *в темпе поступления информации*, если n-й выходной символ для всех n печатается после того как прочитан n-й и перед тем как будет прочитан $(n+1)$-й входной символ [1]). Для умножения, производимого в темпе поступления информации, множимое и множитель задаются бинарными числами, причем первым идет младший разряд и каждый входной символ кодирует два соответствующих входных разряда. Можно всегда предполагать, что при умножении N-разрядных чисел требуется получить лишь последние из N значащих цифр. (Оставшиеся цифры, если это желательно, можно найти, приписывая нули к аргументам.) Умножение в темпе поступления информации несомненно возможно, хотя прямое его выполнение может занять время, по меньшей мере пропорциональное n в промежутке между чтением $(n-1)$-й и n-й цифр, т. е. время порядка N^2 для N-разрядного произведения. Мы покажем здесь, что минимальное среднее время вычисления для умножения, производимого в темпе поступления информации, ограничено снизу и сверху функциями вида $N(\log N)^k$, где для нижних оценок k приблизительно равно 1, а для верхних k приблизительно 1 или 2 в зависимости от рассматриваемого класса. Точные результаты приводятся в § 6, 7 и 8. За всеми необходимыми сведениями мы отсылаем к работе [3].

---

[1]) Мы используем эту сильную форму определения, которая запрещает очень быстро выдавать выход, из соображений технического удобства. Это ограничение несущественно по двум причинам. Для бинарного умножения i-я цифра произведения не может быть определена, пока не будут прочитаны i-е цифры двух входов, за исключением того случая, когда оба числа четные; следовательно, можно воспользоваться более слабым определением не более чем для четверти всех возможных входов. При этом наши средневременные оценки изменяются только на постоянный множитель. Кроме того, чтобы удовлетворять сильному определению, любую МОА можно модифицировать за счет добавления двухголовочной линейной ленты, служащей выходным буфером, не меняя ее временны́е характеристики. Это нисколько не повлияет на наши нижние оценки (ср. [5]).

2. МАШИННЫЕ МОДЕЛИ

В класс машин, для которого применимы наши доказательства, мы хотим включить не только знакомые многоленточные машины, но также машины Тьюринга с лентами большей размерности и несколько упрощенные варианты «машин со случайным доступом». Мы будем исключать повторяющиеся массивы и другие машины с неограниченным параллелизмом, так как на них возможно производить умножение в «реальное время» [2]. Наши определения в основном совпадают с определениями из [3] с незначительными отличиями, чтобы сделать более простыми понятия и доказательства или для того, чтобы полнее использовать технические возможности доказательства. Предполагается, что читатель знаком с основными определениями и методами теории автоматов [6].

Машина с ограниченной активностью (МОА) имеет детерминированное управляющее устройство с конечным числом состояний, оперирующее с входной лентой, с которой можно только считывать и на которой символы располагаются только в одну строчку, с выходной лентой, на которую можно только писать, причем тоже в одну строчку, и с *памятью*. Память — счетное множество *ячеек*, каждая из которых может содержать бинарную величину. Память доступна и может модифицироваться конечным фиксированным числом подвижных *рабочих головок*, которые характеризуются конечным множеством сдвигов $\varphi_1, \ldots, \varphi_p$. Сдвиг φ_i для каждого i есть отображение множества ячеек в само себя, и головка, расположенная над некоторой ячейкой x, может сдвигаться за один шаг к ячейке $\varphi_i(x)$. Полный шаг МОА описывается следующим образом. В зависимости от состояния конечного управляющего устройства входная лента *может быть* продвинута на один символ и в точности одна рабочая головка *должна быть* «сдвинута» одним из сдвигов. Затем в зависимости от управляющего состояния, нового входного символа (если он существует) и величины, которая хранится в обозреваемой ячейке памяти, возможно запоминание новой величины, определение выходного символа и введение нового управляющего состояния. Таким образом, для каждого данного символа на входной ленте существует единственный шаг, когда этот символ читается, и по определению он не может быть прочитан еще раз. Более того, на каждом шаге читается только один символ из памяти, поэтому мы можем говорить о «символе памяти на шаге s».

Разнообразные ограничения в этом определении, такие, как бинарная память, движение одной головки на каждом шаге и отсутствие зависимости нового шага от ранее запомненной величины, вводятся для того чтобы упростить описание, увели-

чив при этом время умножения не более чем на постоянный множитель по сравнению с более гибкими машинами.

Будем говорить, что вычисление совершается в *реальное время*, если оно совершается в темпе поступления информации и каждый входной символ читается через фиксированное число шагов после предыдущего входа. Отметим, что память не должна быть первоначально «пустой», за исключением тех случаев, когда рассматривается специальный класс «униформных» машин, определяемый ниже.

Легко сконструировать МОА, которая умножает в реальное время. Для этого необходима структура памяти, основанная на бесконечном бинарном дереве, проходимом единственной головкой, которая идет по левой или правой ветви в зависимости от входных цифр. Правильный выход либо уже хранится в ячейке дерева, либо же очевидным образом закодирован в его структуре. Например, эта структура может иметь сдвиг ψ, такой, что для всех x или $\psi(x) = x$, или же $\psi(\psi(x)) = x$ и $\psi(x) \neq x$. Выполняемая альтернатива может быть определена для любой ячейки последовательностью из нескольких шагов.

Для того чтобы обойтись без такой пророческой конструкции, Кук и Андера [3] ввели два класса — определенные ниже *полиномиально ограниченные* и *униформные* машины. Мы добавили также еще два класса.

1) *Полиномиально ограниченные машины.* Структура памяти полиномиально ограничена, если существуют константы c и d, такие, что для всех ячеек x и для всех t число ячеек, которые достижимы из x за t шагов, равно не более чем ct^d. МОА с такой структурой памяти называется *полиномиально ограниченной машиной.*

2) *Униформные машины.* Структура памяти является униформной, если для каждой пары ячеек x, y существует перестановка f, такая, что $f(x) = y$ и для каждого сдвига φ_i этой структуры $f \circ \varphi_i = \varphi_i \circ f$. МОА с первоначально «пустой» (т. е. когда каждая ячейка имеет одинаковую первоначальную величину) униформной структурой называется *униформной машиной.* Читателю рекомендуется работа [3] для более подробного обсуждения этого и других классов.

При подходящем определении машины Тьюринга даже с несколькими головками и лентами большой размерности удовлетворяют *обоим* ограничениям. Наш основной результат справедлив для машин, которые удовлетворяют *по крайней мере одному* ограничению. Описанная выше МОА, умножающая в реальное время, не удовлетворяет ни одному из них.

3) *Одноразмерные многоголовочные многоленточные машины Тьюринга.* Можно доказать более сильный результат, чем

для классов (1) и (2), если ограничиться рассмотрением только одноразмерных, т. е. линейных лент.

4) *Забывающие машины.* Для этого класса мы переносим внимание со структуры памяти, которая может быть произвольной, на форму управления с конечным числом состояний или «программу». Ячейка (единственная) памяти, доступная на каждом шаге, определяет последовательность запоминания для любого вычисления, и это зависит, вообще говоря, от входа. Машина называется *забывающей*, если для входной последовательности данной длины последовательность запоминания фиксирована, т. е. она не зависит от входных символов. Естественно, что состояние управляющего устройства и величины, хранящиеся в памяти, могут (а в общем случае так и бывает) зависеть от входных символов; более точно — это движение головок, которое инвариантно. По отношению к забывающим машинам у нас двойной интерес. Во-первых, это ограничение позволяет получить очень простое доказательство улучшенной нижней оценки, и, во-вторых, оказывается, что почти все алгоритмы, предлагаемые или используемые для умножения, являются забывающими или могут быть сделаны забывающими только путем введения постоянного множителя в оценке времени вычисления.

В § 7 мы дадим ретроспективный обзор других классов машин, для которых применима эта техника доказательства.

3. РЕТРОСПЕКТИВНЫЕ ФУНКЦИИ

Неформально функция, отображающая входную последовательность в выходную, является *ретроспективной*[1]), если выходные величины в любом сегменте очень сильно зависят от входных величин непосредственно предшествующего сегмента и, следовательно, для оценки функции необходимо «вернуться» к предыдущему входному сегменту. Как будет показано, умножение в темпе поступления информации является ретроспективным.

Пусть $K_N = \sum_{2^i < N} 2^{2^i}$. Отсюда следует, что в бинарном представлении K_N в i-м разряде расположена «1» тогда и только тогда, когда i является степенью двойки, причем эти разряды перенумерованы, начиная с нуля, с правого (низкоразрядного) конца. Преимущество такого определения K_N в том, что умножение N-разрядного числа на K_N типично ретроспективно, а основное доказательство проще, чем для двухвходового умножения. В теореме 1 приводится нижняя оценка среднего времени

[1]) Мы сочли возможным не следовать Куку и Андере, давшим весьма сложное определение этих функций.

умножения в темпе поступления информации N-разрядного числа на K_N, следовательно, и для максимального времени при умножении в темпе поступления информации. Теорема 2 утверждает, что такая же оценка справедлива для среднего времени при произвольном умножении в темпе поступления информации.

На рис. 1 показано умножение K_N на N-разрядное число X с результатом Z. Числа представлены обычным образом, справа расположены младшие разряды; R и M — неотрицательные целые, и мы всегда будем считать, что $R = 2^r$ — степень двойки. Величина W представляет собой ту подобласть X, которая содержит разряды $X_{M+R-1} \ldots X_{M+1}X_M$, а Y — подобласть Z, которая содержит разряды $Z_{M+2R-1} \ldots Z_{M+R+1}Z_{M+R}$.

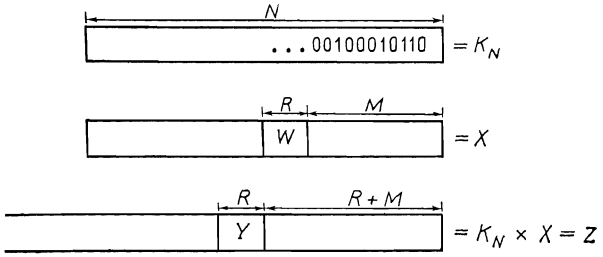


Рис. 1.

Мы иногда будем представлять W и Y как R-разрядные целые из интервала от 0 до $2^R - 1$. Говоря, что W допускает величину i, мы подразумеваем, что бинарное разложение i можно поместить в W-подобласть X. Это изменит Z, так как Z всегда обозначает произведение $K_N \cdot X$, что в свою очередь воздействует на величину Y. Таким образом, можно исследовать зависимость Y от W.

То, каким образом Y будет меняться с изменением W, конечно, зависит от оставшихся разрядов величины X, отличных от разрядов из W, которые мы обозначим через $X \setminus W$. Как число, величина $X \setminus W$ есть значение бинарной последовательности, полученной подстановкой во все разряды W-области X нулей.

Для некоторой фиксированной величины $X \setminus W$ пусть W принимает все возможные значения 0, 1, ..., $2^R - 1$; тогда $X_i = X \setminus W + i \cdot 2^M$, $0 \leqslant i \leqslant 2^R - 1$. Пусть $Z_0, Z_1, \ldots$, и $Y_0, Y_1, \ldots$ будут соответствующими величинами из Z и Y, именно $Z_i = K_N \cdot X_i$, а Y_i есть Y-область Z_i.

Если $i < j$, то $Z_j - Z_i = (X_j - X_i) K_N = (j - i) 2^M K_N$. Так как $K_N = K_{2R} + 2^{2R} \cdot \bar{K}$ для некоторого целого $\bar{K}$, мы имеем

$$Z_j - Z_i \equiv (j - i) 2^M K_{2R} \pmod{2^{M+2R}}.$$

Допустим теперь $Y_i = Y_j$. Тогда $Z_j - Z_i \equiv a2^M \pmod{2^{M+2R}}$ для некоторого целого a, $|a| < 2^R$, и, следовательно, $(j-i)K_{2R} \equiv a \pmod{2^{2R}}$. Так как $R = 2^r$, то

$$2(2^R - 1) \geqslant K_{2R} = 2^{2^r} + 2^{2^{r-1}} + \ldots + 2^{2^0} \geqslant 2^R.$$

В силу правого неравенства, $(j-i)K_{2R} \geqslant 2^R > a$, и, значит, для выполнения равенства мы должны иметь $(j-i)K_{2R} \geqslant a + 2^{2R} > 2^{2R} - 2^R$. Из левого неравенства для K_{2R} следует, что $j - i > \frac{1}{2}2^R$. Поэтому для любого i существует не более одного $j > i$, такого, что $Y_i = Y_j$, и нами доказана

Лемма 1. *Для фиксированных значений M, R, N и $X \setminus W$ каждое значение Y может возникнуть не более чем при двух значениях W.*

4. ЧАСТИЧНОЕ ПЕРЕКРЫТИЕ

Это понятие взято за основу очень изящного комбинаторного доказательства, приведенного в [3]. Недавно такой метод вновь был использован Андерой в работе [1]. Необходимость ввести определение частичного перекрытия появилась при вычислении ретроспективных функций. Очевидный способ, с помощью которого можно найти информацию о предыдущем входном сегменте, состоит в проверке ячеек, которые посещались при чтении этого сегмента. Частичное перекрытие определяется с использованием только определенной ранее в § 2(4) последовательности памяти. Если два последовательных посещения одной и той же ячейки l встречаются на шагах s_1 и s_2 $(s_1 \leqslant s_2)$, то пара (s_1, s_2) называется *перекрывающей парой*, l называется *перекрывающей ячейкой* шага s_2 и величина, которая хранится в l на шаге s_1 и используется на шаге s_2, называется *величиной перекрытия* шага s_2. *Суммарным перекрытием* называется

$$\Omega = |\{(s_1, s_2) \mid (s_1, s_2) - \text{перекрывающая пара}\}|.$$

Очевидно, что суммарное время $T \geqslant \Omega$, так как каждый шаг s является второй компонентой не более чем одной перекрывающей пары, что зависит от того, посещалась ли ранее ячейка, которая используется на шаге s.

Пусть C_1, C_2 — различные смежные временны́е интервалы во время вычисления. Определим *перекрытие* (C_1, C_2) как число перекрывающих пар (s_1, s_2), для которых $s_1 \in C_1$ и $s_2 \in C_2$.

Без потери общности будем предполагать, что $N = 2^n$. Для любых $i = 0, \ldots, n$, называемых *уровнями*, определим $R_i = 2^i$, и если $S = S_{N-1}S_{N-2} \ldots S_1S_0$ — любой список длины N, то разобьем S на соприкасающиеся блоки $S_{i,2^{n-i}-1}, \ldots, S_{i,1}, S_{i,0}$

длины R_i, где $S_{i,j} = S_{(j+1)R_i-1} \dots S_{jR_i}$, $0 \leqslant j \leqslant 2^{n-i} - 1$. Если X — входная последовательность, то временной интервал $C_{i,j}$ начинается, когда читается самая правая цифра из $X_{i,j}$, и продолжается до того момента, пока должно начаться чтение самой правой цифры из $X_{i,j+1}$ (или до тех пор, пока вычисление кончится, если не существует такого $j+1$).

Пусть $t_{i,j}$ — длина интервала $C_{i,j}$. Ясно, что для каждого уровня i суммарное время вычисления есть $T = \sum_j t_{i,j}$. Мы определим $\omega_{i,j}$ как перекрытие $(C_{i,j}, C_{i,j+1})$ для всех подходящих i, j и $\omega_i = \sum_j \omega_{i,2j}$ для любого i.

Лемма 2. *Суммарное перекрытие* $\Omega = \sum_i \omega_i$.

Доказательство. Пусть (s_1, s_2) — перекрывающая пара и i — наименьший уровень, такой, что $s_1, s_2 \in C_{i,j}$ для некоторого j. Интервал $C_{i,j}$ — это соединение двух интервалов $C_{i-1,2j}$ и $C_{i-1,2j+1}$ на следующем нижнем уровне. Выберем i так, что $s_1 \in C_{i-1,2j}$ и $s_2 \in C_{i-1,2j+1}$, поэтому (s_1, s_2) вносит вклад в $\omega_{i-1,2j}$ и, следовательно, в ω_{i-1}. Допустим, он вносит вклад в $\omega_{i',2j'}$. Тогда $i' \leqslant i-1$, так как s_1 и s_2 принадлежат одному и тому же интервалу для каждого уровня выше $i-1$, а $i' \geqslant i-1$, так как если (s_1, s_2) вносит вклад в $\omega_{i',2j'}$, то и s_1, и s_2 находятся в одном интервале $C_{i'+1,j'}$ на уровне $i'+1$. Следовательно, $i' = i-1$, и ясно, что $j' = j$. Отсюда заключаем, что каждая пара перекрытия вносит вклад в точности один раз в одно перекрытие ω_i и, следовательно, в точности один раз в $\sum_i \omega_i$. ■

5. ВЫЧИСЛЕНИЕ С МАЛЫМ ПЕРЕКРЫТИЕМ

Рассмотрим вычисление в темпе поступления информации некоторой машиной $\mathfrak{M}$ при входе X и выходе Z. Как и прежде, M и R — фиксированные числа, W есть подслово X длины R, именно $X_{M+R-1} \dots X_M$, и Y — подслово Z длины R, именно $Z_{M+2R-1} \dots Z_{M+R}$. В оставшейся части этого параграфа $X \setminus W$, M и R фиксированы, и мы будем исследовать изменение величины Y в зависимости от изменения W. В отличие от предыдущих параграфов теперь Z представляет выход машины $\mathfrak{M}$ при входе X.

Задание определенных значений W полностью определяет вычисление машины $\mathfrak{M}$. Пусть s_0 — шаг, который продвигает входную головку на первый символ подслова W, s_1 — шаг, который сдвигает входную головку с последнего символа W, и s_2 — шаг, который читает следующий входной символ, после того как выдано подслово Y. Определим интервал C_W как мно-

жество шагов от s_0 до $s_1 - 1$, C_Y как множество шагов от s_1 до $s_2 - 1$, и пусть $t_W = s_1 - s_0$, $t_Y = s_2 - s_1$ — отрезки времени, ассоциированные соответственно с C_W и C_Y. Величина T — суммарное время вычисления, и мы полагаем $\omega =$ перекрытие (C_W, C_Y). Это понятие иллюстрируется рис. 2.

При изменении W изменяются также ω, t_Y, t_W, T и Y. Пусть $Q(\hat{\omega}, \hat{t}, \hat{T})$ — суммарное число различных значений Y, порождаемых таким W, что $\omega \leqslant \hat{\omega}$, $t_Y \leqslant \hat{t}$ и $T \leqslant \hat{T}$. Если $\mathfrak{M}$ вычисляет ретроспективную функцию, то $Q(\hat{\omega}, \hat{t}, \hat{T})$ должна быть большой и мы будем использовать этот факт для получения ограничений на $\hat{\omega}$, $\hat{t}$ и $\hat{T}$, что в конце концов приведет к нашей нижней оценке для T.

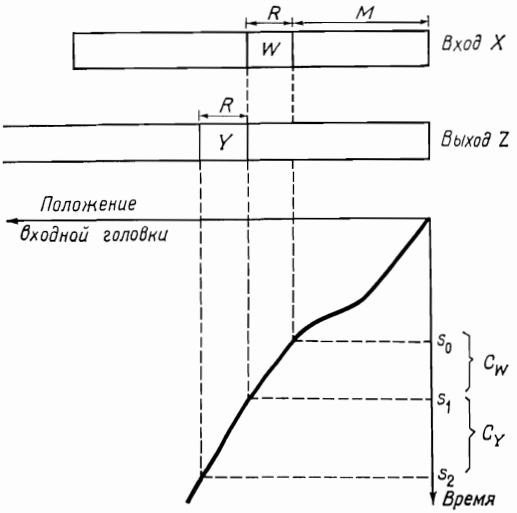


Рис. 2.

Наши верхние оценки для Q зависят от типа машины, но находятся с использованием того же общего метода. Для данной величины W мы наблюдаем вычисление на интервале C_Y и записываем подходящим образом информацию о W, что влияет на вычисление в течение интервала C_Y.

Для каждого класса машин будет записано достаточно информации, чтобы обеспечить правильность следующего.

Условие 1. *Пусть w и w' — два значения W, а y и y' — соответствующие значения Y. Если информация, записанная для w и w' одинакова, то $y = y'$.*

Из условия 1 непосредственно следует, что число различных возможных информационных записей, найденных для значе-

ний W, при которых ω, t_Y и T ограничены соответственно значениями $\hat{\omega}$, $\hat{t}$ и $\hat{T}$, является верхней оценкой для $Q(\hat{\omega}, \hat{t}, \hat{T})$.

Лемма 3. *Существует константа* C, *зависящая только от* $\mathfrak{M}$, *такая, что для* $\hat{T} > 1$ *и* $\hat{\omega} \leqslant \hat{t} \leqslant \hat{T}$

a) $Q(\hat{\omega}, \hat{t}, \hat{T}) \leqslant \hat{T}^C 2^{\hat{\omega}} \binom{\hat{t}}{\hat{\omega}}$, *если машина* $\mathfrak{M}$ — *полиномиально ограниченная или униформная*;

b) $Q(\hat{\omega}, \hat{t}, \hat{T}) \leqslant \hat{T}^C 2^{\hat{\omega}}$, *если* $\mathfrak{M}$ — *одноразмерная многоголовочная многоленточная машина Тьюринга*;

c) $Q(\hat{\omega}, \hat{t}, \hat{T}) \leqslant C 2^{\hat{\omega}}$, *если машина* $\mathfrak{M}$ — *забывающая*.

Доказательство. (a) *Случай* 1. Машина $\mathfrak{M}$ — *полиномиально ограниченная*. Информационная запись содержит состояние устройства управления, позицию каждой головки в момент времени s_1, подмножество $\theta = \{t_1, \ldots, t_{\hat{\omega}}\}$ целых от 0 до $\hat{t} - 1$ и бинарную последовательность v длины $\hat{\omega}$. Величина θ выбирается так, чтобы включать моменты времени, связанные с s_1 из всех W-перекрывающих шагов. Наконец, i-й разряд из v равен символу, на который ссылаются в момент $s_1 + t_i$ и который будет перекрывающей величиной, если $s_1 + t_i$ есть W-перекрывающий шаг. Этим гарантируется выполнение условия 1.

Суммарное число таких записей, очевидно, не более чем $C 2^{\hat{\omega}} \binom{\hat{t}}{\hat{\omega}} H^C$, где H — число возможных позиций для каждой головки на шаге s_1. Так как $\mathfrak{M}$ — полиномиально ограничена, то $H \leqslant \hat{t}_W^d \leqslant T^d \leqslant \hat{T}^d$; получена оценка типа (a).

(a) *Случай* 2. Машина $\mathfrak{M}$ — *униформная*. Этот случай в точности совпадает с первым, за исключением того, что мы не записываем действительную позицию головки в момент s_1, так как число возможных позиций слишком велико. Вместо этого мы записываем для каждой головки h шаг C_Y (если он существует), при котором h впервые посещает ячейку l до шага s_0, вместе с моментом времени первого посещения l (который единственным образом определяет l). Назовем такую ячейку l *заполненной*. (В случае более чем одной головки должно быть записано небольшое количество дополнительной информации для подсчета возможных пересечений головок перед просмотром заполненной ячейки. Более подробно это описано в [3].)

Если головка h никогда не посещает заполненную ячейку в течение интервала C_Y, то, в силу униформности, символы, которые читаются и пишутся этой головкой, могут быть единственным образом определены из перекрывающих шагов и величин без знания положения h в момент s_1. С другой стороны, если h не посещает заполненную ячейку l, то шаг C_Y, при котором l посещается, вместе с ячейкой l обеспечивает всю требуемую

информацию для определения читаемых символов головкой в течение интервала C_Y. Ячейку l можно указать временем ее первого посещения, следовательно, существует не более $t_Y T \leqslant \hat{T}^2$ различных начальных позиций единственной головки h, которые необходимо различать.

(b) $\mathfrak{M}$ — *одноразмерная многоголовочная многоленточная машина Тьюринга.* Это специальный случай полиномиально ограниченной машины, поэтому мы можем записывать состояние и начальные позиции головок как в п. (a), случай 1. Однако те позиции, на которых может встречаться перекрытие, определяются проще, так как ячейки, посещаемые во время интервала C_W каждой из головок, формируют интервал. Следовательно, необходимо указать только его конечные точки. Поскольку на линейной ленте за T шагов данная головка может посетить не более $2T$ ячеек, существует не более $2T^2$ возможных интервалов, приходящихся на головку. Как и прежде, достаточна бинарная последовательность длины ω для того, чтобы записать перекрывающие величины. Следовательно, суммарное число таких записей не более $CH^C = (2\hat{T}^2)^C = 2^{\hat{\omega}}$, где H имеет такое же значение, как в п. (a), случай 1.

(c) Машина $\mathfrak{M}$ — *забывающая.* Позиции головок на каждом шаге независимы от W, следовательно, должны быть записаны только состояние управления на шаге s_1 и значения перекрывающих ячеек; тем самым обеспечивается оценка $C2^{\hat{\omega}}$. ■

Мы закончили наши приготовления для главного доказательства комбинаторной леммы. Можно пояснить идею доказательства следующим образом. Пусть $\mathfrak{M}$ — машина, которая умножает в темпе поступления информации, и пусть $\hat{\omega}$, $\hat{t}$ и $\hat{T}$ — границы для ω, t_Y и T соответственно, такие, что $Q(\hat{\omega}, \hat{t}, \hat{T}) < 2^{3R/4}$. По лемме 1 все, кроме $2 \cdot 2^{3R/4}$ значений W, — бесконечно малые величины; т. е. одна из трех границ превышена. Это дает явное выражение нижней оценки для ω через величины t_Y и T, которая на самом деле утверждает что, если T мало, то суммарное перекрытие Ω велико, а это влечет за собой, что T велико. Следовательно, T в среднем должно быть большим.

Лемма 4. *Пусть C, R, a — положительные константы, такие, что* $\log a > 2(C+3)$. *Если* $0 < \omega \leqslant t$ *и* $\omega + t/a + \log T \leqslant R/(2 \log a)$, *то* $T^C 2^{\omega} \binom{t}{\omega} < 2^{3R/4}$. (Все логарифмы в этой работе берутся по основанию два.)

Доказательство. Для любых $p \geqslant q > 0$ по формуле Стирлинга имеем

$$\binom{p}{q} \leqslant \frac{p^q}{q!} \leqslant \left(\frac{pe}{q}\right)^q.$$

Допустим, что предположение леммы справедливо, тогда $\omega < R/(2\log a)$, $t < aR/(2\log a)$ и $\log T < R/(2\log a)$. Функция

$$T^C 2^{\omega}\binom{t}{\omega} < T^C\binom{2t}{\omega}$$

монотонно увеличивается в зависимости от ω, t и T, так как $\omega \leqslant T$. Следовательно,

$$T^C\binom{2t}{\omega} < 2^{CR/(2\log a)}\binom{aR/\log a}{R/(2\log a)} < \\ < 2^{CR/(2\log a)}(2ae)^{R/(2\log a)} < 2^{3R/4}. \blacksquare$$

6. ОСНОВНЫЕ РЕЗУЛЬТАТЫ И ДОКАЗАТЕЛЬСТВА

Теорема 1. *Существует константа c, такая, что для любой МОА $\mathfrak{M}$, умножающей при всех N в темпе поступления информации N-разрядные числа на K_N, среднее время $T(N)$ для всех чисел длины N удовлетворяет следующим оценкам при достаточно больших значениях N:*

(i) *если машина $\mathfrak{M}$ — полиномиально ограниченная или униформная, то $T(N) > cN\log N/\log\log N$;*

(ii) *если $\mathfrak{M}$ — одноразмерная многоголовочная и многоленточная машина Тьюринга или забывающая, то $T(N) > cN\log N$.*

Доказательство. Вначале допустим, что машина $\mathfrak{M}$ — полиномиально ограниченная или униформная. Можно предположить, что $\log\log N > 2(C+3)$, где C такое же, как и в лемме 3, и определить $a = \log N$. Мы вновь рассмотрим ситуацию, которая изображена на рис. 1 и 2, где $X \setminus W$ фиксировано, а W может изменяться. Применяя леммы 1, 3(а) и 4, получаем, что число различных W, для которых $\omega + t/a + \log T \leqslant R/(2\log a)$, меньше чем $2 \cdot 2^{3R/4}$. Следовательно,

$$\text{среднее}_W(\omega + t/a + \log T) > R/(3\log a) \quad \text{для} \quad R \geqslant 16.$$

Так как это неравенство выполняется для всех значений $X \setminus W$, то

$$\text{среднее}_X(\omega + t/a + \log T) > R/(3\log a),$$

где среднее берется по всем N-разрядным числам. Если предположим, что среднее $_X T < N^2$, то среднее $_X \log T \leqslant \log$ среднее$_X T < 2\log N$, так как геометрическое среднее не превосходит арифметического. Теперь используем равенство: среднее$(A+B)$ = среднее(A) + среднее(B). В силу этого, из неравенства

$$\text{среднее}_X(\omega + t/a) > R/(3\log a) - 2\log N > R/(4\log a)$$

получается, что $R > 24(\log N)(\log\log N)$.

Допустим теперь, что $W = X_{i,2j}$ и $Y = Z_{i,2j+1}$ для некоторых i, j; значит, $\omega = \omega_{i,2j}$, $t = t_{i,2j+1}$ и $R = R_i = 2^i$. Суммированием по j предыдущее неравенство дает

$$\text{среднее}_X \omega_i + \text{среднее}_X \sum_j t_{i,2j+1}/a =$$

$$= \sum_j \text{среднее}_X (\omega_{i,2j} + t_{i,2j+1}/a) > \sum_j R_i/(4 \log a) = N/(8 \log a).$$

Если предположить, что $\text{среднее}_X T < N \log N/(16 \log \log N)$, то, в силу $\sum_j t_{i,2j+1} < T$, имеем

$$\text{среднее}_X \omega_i > N/(8 \log a) - N/(16 \log \log N) = N/(16 \log \log N).$$

Так как это неравенство выполняется при всех i, таких, что

$$i < \log N \quad \text{и} \quad 2^i = R_i > 24 (\log N)(\log \log N),$$

мы заключаем, что

$$\text{среднее}_X T > \text{среднее}_X \Omega = \sum_i \text{среднее}_X \omega_i \geqslant$$

$$\geqslant [\log N - \log (24 (\log N)(\log \log N))]\, N/(16 \log \log N) \geqslant$$

$$\geqslant N \log N/(17 \log \log N)$$

при условии, что N достаточно велико. Таким образом, доказан случай (i).

Доказательство случая (ii) в некотором смысле проще. Можно легко показать, что если $\omega + 2c \log T \leqslant R/2$, то $T^C 2^\omega < 2^{3R/4}$. Аналогично тому как это было сделано в случае (i), можно получить, что среднее $_X(\omega + 2c \log T) > R/3$. Если среднее $_X T < N^2$ и $R_i > 48c \log N$, то среднее $_X \omega_{i,2j} > R_i/4$ и среднее $_X \omega_i > N/8$, следовательно,

$$\text{среднее}_X T > \text{среднее}_X \Omega = \sum_i \text{среднее}_X \omega_i > N (\log N)/9.$$

Это и доказывает случай (ii). Конечно, доказательство только для забывающих машин было бы гораздо проще, так как ω_{ij}, t_{ij}, T и т. д. не зависит от X. ■

Мы сразу же можем расширить это доказательство, удаляя зависимость от K_N. Покажем, что умножение почти всех чисел столь же ретроспективно, как умножение на K_N. В ситуации, изображенной на рис. 1 и в лемме 1, заменим K_N на произвольное N-разрядное число K_N^*.

Лемма 5. *Для любого h, $0 < h < 2^R$, если $Y_i = Y_{i+h}$ для некоторого i, то K_{2R}^* должно иметь одно из не более чем 2^{R+1} возможных значений.*

Доказательство. Как в доказательстве леммы 1, если $Y_t = Y_{i+h}$, то $h \cdot K^*_{2R} \equiv a \pmod{2^{2R}}$ для некоторого a, $|a| < 2^R$.

Пусть $d =$ н. о. д. $(h, 2^{2R})$. Тогда $d|a$, следовательно, $a = kd$, где $|kd| < 2^R$. По определению $d|2^{R-1}$, и верно, что $h < 2^R$. Следовательно,

$$k \in \left\{ -\frac{2^R}{d} + 1, -\frac{2^R}{d} + 2, \ldots, -1, 0, 1, \ldots, \frac{2^R}{d} - 1 \right\},$$

поэтому существует $(2 \cdot 2^R/d) - 1$ таких k.

Из элементарной теории чисел известно, что существует в точности d значений K^*_{2R} в области $0 \leqslant K^*_{2R} < 2^{2R}$, которые удовлетворяют равенству $hK^*_{2R} \equiv kd \pmod{2^{2R}}$. Следовательно, существует не более $d(2 \cdot 2^R/d - 1) < 2^{R+1}$ значений K^*_{2R}, для которых $Y_i = Y_{i+h}$. ■

Из этой леммы сразу же вытекает, что по меньшей мере для половины всех возможных значений K^*_{2R} для всех h, $0 < h \leqslant 2^{R-2}$, и для всех i выполняется $Y_i \neq Y_{i+h}$. Следовательно, для этих значений K^*_{2R} не более четырех различных W выдают одинаковые Y. Поэтому доказательство теоремы 1 останется прежним, за исключением того, что термин «среднее$_X$» заменится термином «среднее$_{K_N}$·среднее$_X$». Таким образом, мы доказали, что справедлива

Теорема 2. *Существует константа c, такая, что для любой МОА $\mathfrak{M}$, умножающей в темпе поступления информации, среднее время $T(N)$ для пар N-разрядных чисел удовлетворяет следующим оценкам при достаточно больших значениях N:*

(i) *если машина $\mathfrak{M}$ — полиномиально ограниченная или униформная, то $T(N) > c\,N \log N/\log\log N$;*

(ii) *если $\mathfrak{M}$ — многоленточная машина Тьюринга или забывающая, то $T(N) > cN \log N$.*

Для теорем 1 и 2 мы не знаем прямого доказательства следования одной теоремы из другой.

7. РАСШИРЕНИЯ

В этом параграфе мы наметим пути, которыми могут быть расширены уже рассмотренные классы, допуская в то же время прежние методы доказательства.

Простую «машину со случайным доступом» можно смоделировать с помощью МОА, структура памяти которой основывается на некоем виде бинарного дерева, так что ячейки «адресуются» двоичными последовательностями и доступны за время, пропорциональное длине адреса. Такая структура, конечно, ско-

рее экспоненциальная, нежели полиномиально ограниченная. Однако напомним, что последнее свойство используется в доказательстве только для того, чтобы позволить определить положения головок перед прочтением нового входного символа. Доказательство проходит, как и прежде, так как древовидная структура используется таким образом, что головки возвращаются к корню перед чтением каждого входа (например, если «случайный доступ» выполняется подпрограммой).

Другой подход к памяти со случайным доступом — это структура, основанная на свободной группе с двумя образующими a, b и с четырьмя сдвигами, которые являются левым умножением на a, a^{-1}, b, b^{-1}. Это может быть достаточно полезной памятью со случайным доступом, и, конечно, она является униформной.

С технической точки зрения интересно, что те же самые оценки легко можно получить, когда полиномиально ограниченный класс расширяется заменой «ct^d» в определении на «$c2^{t^\varepsilon}$» для любого $\varepsilon < 1$. К сожалению, мы не знаем естественного класса машин, для которого можно воспользоваться этим расширением.

Наконец, покажем, что без ослабления доказательств для любого из четырех классов машин можно следующим образом добавлять «оракулы». МОА расширяется за счет бесконечного числа состояний устройства управления, но с тем ограничением, что лишь конечное число из них может читать входную ленту. «Оракул», даже нерекурсивный, можно вызывать такой машиной, чтобы читать последовательность ячеек памяти и помещать результат применения его оракульной функции в некоторую другую последовательность ячеек. Вся процедура займет то же число шагов, что и доступ к этим ячейкам, в силу чего доказательства становятся неэффективными. Простой пример, который подчеркивает важность ограничения на темп работы машины для нашего доказательства, — многоленточная машина Тьюринга с оракулом для выполнения автономного умножения в линейное время. Для нее также справедлива нижняя оценка $cN \log N$.

8. ВЕРХНИЕ ОЦЕНКИ

М. Фишер и Стокмейер в [4] приводят мощную технику установления верхних оценок умножения, производимого в темпе поступления информации. Их конструкция показывает, что для широкой области классов машин, включающих многоленточные машины Тьюринга с оракулом и забывающие машины с временно́й сложностью $T(N)$, где T удовлетворяет неравенству $T(2N) \geqslant 2T(N)$, можно построить машину, работающую в темпе поступления информации, с временно́й сложностью, не большей $cT(N) \log N$.

Некоторое расширение их методов показывает, что умножение в темпе поступления информации N-разрядных чисел, где одно из чисел имеет не более $\log N$ единиц, можно произвести за время $O(N \log N)$. В частности, существует машина Тьюринга для умножения в темпе поступления информации на K_N со сложностью $O(N \log N)$, что совпадает с оценкой теоремы 1(ii).

Общий алгоритм умножения в темпе поступления информации можно найти, применив результат Фишера — Стокмейера к алгоритму умножения в темпе поступления информации Шёнхаге — Штрассена [7], сложность которого на машинах Тьюринга выражается величиной $O(N \log N \cdot \log \log N)$. При подходящем построении и быстром доступе таблицы умножения для $\log N$-разрядных чисел, например так, как в машинах со случайным доступом, автономный алгоритм умножения Шёнхаге — Штрассена можно выполнить за время $O(N \log N)$.

	Многоленточные машины Тьюринга	Оракульные машины Тьюринга	Машины со случайным доступом
Умножение в темпе поступления информации	$N \log^2 N \log \log N$	$\underline{N \log N}$	$N \log^2 N$
Умножение в темпе поступления информации на K_N	$\underline{N \log N}$	$\underline{N \log N}$	$N \log N$
Автономное умножение	$N \log N \log \log N$	$\underline{N}$	$N \log N$

Рис. 3.

На рис. 3 представлены некоторые из верхних оценок, полученные с помощью приведенных выше результатов для трех классов машин. Постоянные множители опущены, и черта под выражением означает, что в предыдущих параграфах приведена нижняя оценка такого же порядка. Первый класс — многоленточные машины Тьюринга с одноразмерными лентами, второй — класс МОА с бесконечным числом состояний при ограничении на входные состояния, приведенном в § 7, третий класс — любой вариант машин со «случайным доступом», описанных в § 7. С помощью униформной структуры, основанной на свободных группах с двумя образующими, легко смоделировать машины Тьюринга и память со «случайным доступом». МОА с бинарной древовидной структурой и ограничением на позиции головок, которые приведены в § 7, образуют достаточно мощный класс, для того чтобы быстро выполнять требуемые алгоритмы, хотя техника программирования в этом случае менее прямолинейна.

9. ЗАКЛЮЧЕНИЕ

В этой статье мы описали мощное комбинаторное доказательство, основанное на понятии «частичного перекрытия», и исследовали расширения и ограничения на его применимость. Методы частичного перекрытия применимы только при работе, производимой в темпе поступления информации, но часто с их помощью можно получить оценки сложности, оптимальные с точностью до постоянного множителя.

Важным объектом будущих исследований служит нахождение нетривиальных нижних оценок вычисления без строгого ограничения на темп поступления информации. Такие результаты даже для забывающих машин или комбинаторных цепей были бы значительным продвижением.

СПИСОК ЛИТЕРАТУРЫ

1. Aanderaa S. O., On k-tape versus $(k+1)$-tape real-time computation, *SIAM-AMS Proc.*, **7**, Amer. Math. Soc., Providence, R. I, 1974, pp. 75—96.
2. Atrubin A. J. A one-dimensional real-time iterative multiplier, *IEEE Trans. Electronic Computers*, **ES-14** (1965), 394—399. [Русский перевод: Атрубин А. Д., Одномерное итеративное умножающее устройство, работающее в реальное время, Киб. сб., 5, н. с., 1968, 81—94.]
3. Cook S. A., Aanderaa S. O., On the minimum computation time of functions, *Trans. Amer. Math. Soc.*, **142** (1969), 291—314, MR 40 # 2459. [Русский перевод: Кук С. А., Аандераа С. О., О минимальном времени вычисления функций, Киб. сб., 8, н. с., 1971, 168—200.]
4. Fischer M. J., Stockmeyer L. J., Fast on-line integer multiplication, Proc. 5th ACM Sumpos. on theory of Computing, 1973, pp. 67—72; *J. Comput. Sys. Sci.*, **9** (1974).
5. Fischer P. C., Meyer A. R., Rosenberg A. L., Real-time simulation of multihead tape units, *J. ACM*, **19** (1972), 590—607.
6. Hopcroft J. E., Ullman J. D., Formal languages and their relation to automata, Addison-Wesley, Reading, Mass. 1969, MR 38 # 5533.
7. Shönhage A., Strassen V., Schnelle Multiplikation grosser Zahlen, *Computing* (*Arch. Elektron. Rechnen*), **7** (1971), 281—292, MR 45 # 1431. [Русский перевод: Шёнхаге А., Штрассен В., Быстрое умножение больших чисел, Киб сб., 10, н. с., 1973, 87—98.]

Степени автоматных преобразований[1])

Герхард Рейна

Department of Mathematics, Lehigh University,
Bethlehem, Pennsylvania 18015

Верхняя полурешетка степеней преобразований конечными автоматами аналогична верхней полурешетке степеней рекурсивной неразрешимости (которая возникает из преобразований машинами Тьюринга). Две бесконечные последовательности из конечного алфавита считаются эквивалентными, если каждую из них можно преобразовать в другую посредством некоторого конечного автомата, возможно, после того, как из каждой последовательности будут вычеркнуты начальные сегменты (не обязательно одинаковой длины). Требуется, чтобы выходная последовательность порождалась с той же скоростью, что и входная, ровно один выходной символ на каждый входной. Если такое преобразование возможно только в одном направлении, то между классами эквивалентности выполняется отношение порядка.

Мы показываем, что это частично упорядоченное множество образует верхнюю полурешетку, обладая единственным минимальным классом, и доказываем, что не существует максимального класса. В процессе доказательства последнего утверждения введено понятие полной последовательности, т. е. последовательности, в которой встречается каждый блок алфавита, и отмечена важность этого понятия. Богатство этого частичного упорядочения проиллюстрировано двумя контрастирующими примерами: мы находим одну часть множества, в которой частичное упорядочение плотно, и в то же время приводим два класса $[x] > [z]$, не содержащие класса строго между ними.

1. ВВЕДЕНИЕ

В этой статье мы изучаем некоторое отношение эквивалентности между бесконечными последовательностями и частичное упорядочение классов эквивалентности. Грубо говоря, две последовательности эквивалентны, если каждую из них можно

---

[1]) Gerhard Rayna, Degrees of finite-state transformability, *Information and Control*, **24** (1974), 144—154.

преобразовать в другую, возможно с фиксированной задержкой, посредством некоторой пары конечных автоматов. Если такое преобразование возможно только в одном направлении, то между классами эквивалентности выполняется отношение порядка.

Конечный автомат «читает» входной символ при каждом «такте» (время дискретно) и сразу же «пишет» выходной символ. Входная последовательность считывается постоянно слева направо, с той же скоростью порождается выходная последовательность.

Очевидно, что преобразование, выполняемое конечным автоматом, представляет собой вычислимый процесс. Следовательно, наше частичное упорядочение согласуется с частичным упорядочением степеней рекурсивной неразрешимости [2], но гораздо тоньше.

В наших доказательствах мы будем пользоваться формулировкой конечного автомата Мили: выходной символ может зависеть как от входного символа, так и от состояния автомата. (См., например, [3] или [1, стр. 7].) Если последовательность x можно преобразовать в последовательность y посредством автомата Мили, то x можно преобразовать в y с единичной задержкой посредством автомата Мура (см. [4] или [1, стр. 12]), и обратно (во втором случае задержка не требуется). Так как наше отношение эквивалентности допускает конечную фиксированную задержку, результаты применимы также и к автоматам Мура.

2. ОПРЕДЕЛЕНИЯ

В формулировке Мили конечный автомат — это множество T, содержащее

конечный входной алфавит Σ,
конечный выходной алфавит Σ',
конечное множество состояний S,
функцию перехода состояний f: $\Sigma \times S \to S$,
выходную функцию χ: $\Sigma \times S \to \Sigma'$.

Если выделено начальное состояние s_0, то пару (T, s_0) называют инициальным автоматом. Часто мы будем обозначать одной буквой T инициальный автомат (а не автомат без выделенного начального состояния), если не будет необходимости явно указывать выделенное состояние.

Говорят, что автомат (T, s_0) преобразует последовательность $\{x_n\}$, $n = 0, 1, 2 \ldots$, символов из Σ в последовательность $\{z_n\}$ символов из Σ', если

$$s_{n+1} = f(x_n, s_n) \quad \text{и} \quad z_n = \chi(x_n, s_n) \quad \text{для} \quad n = 0, 1, 2, \ldots .$$

Говорят также, что входная последовательность $\{x_n\}$ проводит автомат (T, s_0) через последовательность состояний $\{s_{n+1}\}$. Отметим, что начальное состояние s_0 опускается. «Последовательность» всегда будет означать бесконечную последовательность символов, снабженных индексами 0, 1, 2, ... Конечная последовательность будет называться блоком. Если длина блока равна N, то назовем его N-блоком. Говорят, что входной N-блок $\{x_0, x_1, x_2, \ldots, x_{N-1}\}$ проводит автомат (T, s_0) через блок состояний $\{s_1, s_2, \ldots, s_N\}$ и производит выходной блок $\{z_0, z_1, \ldots, z_{N-1}\}$. Состояние s_N будем называть финальным состоянием.

Две последовательности символов $\{x_n\}$, $\{y_n\}$, каждая из конечного алфавита (не обязательно одного и того же), будут считаться эквивалентными тогда и только тогда, когда существуют такие инициальные конечные автоматы (T, s_0) и (T', s_0') и такие неотрицательные целые d и d', что (T, s_0) преобразует $\{x_n\}$ в $\{y_{n-d}\}$, т. е. $\{y_n\}$ задерживается на фиксированное d, и (T', s_0') преобразует $\{y_n\}$ в $\{x_{n-d'}\}$. (В этом определении члены последовательности с отрицательными нижними индексами можно выбрать произвольно.) Легко видеть, что это — отношение эквивалентности. (Напомним, что можно определить композицию двух инициальных конечных автоматов и она тоже будет инициальным конечным автоматом.) Класс эквивалентности, содержащий последовательность x, обозначается $[x]$.

В частности, последовательность эквивалентна самой себе, если она задержана (в нее вставлены начальные символы), продвинута (опущено в начале конечное число символов) или в ней изменено любое конечное число символов. Каждый класс эквивалентности называется «степенью автоматных преобразований».

Если инициальный автомат преобразует последовательность x в последовательность y или в последовательность, эквивалентную y, то пишут $[x] \geqslant [y]$. Легко заметить, что это определяет частичное упорядочение классов эквивалентности. Заметим, что $[x] \geqslant [y]$ тогда и только тогда, когда последовательность x преобразуется каким-нибудь инициальным автоматом в последовательность y, задержанную на некоторое конечное неотрицательное число d.

В следующих разделах мы установим ряд свойств этого частичного упорядочения.

3. ЭЛЕМЕНТАРНЫЕ СВОЙСТВА

Т е о р е м а. *Класс* $[0]$ (*которому принадлежит постоянная последовательность* 0) *состоит из почти периодических последовательностей. Кроме того,* $[x] \geqslant [0]$ *для любой последовательности* x.

Доказательство. Пусть x — почти периодическая последовательность. Тогда легко описать инициальный автомат, который безотносительно к входу всегда выдает на выходе последовательность x. Следовательно, $[0] \geqslant [x]$.

Обратно, еще проще описать инициальный автомат, который для произвольного входа выдает постоянную последовательность 0. Следовательно, $[x] \geqslant [0]$.

Это показывает, что если x почти периодична, то $[0] = [x]$. Второй абзац этого доказательства доказывает второе утверждение теоремы.

Теорема. *Частично упорядоченная система степеней автоматных преобразований является верхней полурешеткой.*

Доказательство. Мы должны показать, что для любых классов эквивалентности $[x]$, $[y]$ существует такой класс $[z]$, что $[x] \leqslant [z]$, $[y] \leqslant [z]$ и $[z] \leqslant [w]$ для любого класса $[w]$, для которого $[x] \leqslant [w]$ и $[y] \leqslant [w]$.

Пусть $\{x_i\} \in [x]$ и $\{y_i\} \in [y]$ — последовательности над алфавитами Σ_1 и Σ_2 соответственно. Рассмотрим последовательность $\{z_i = (x_i, y_i)\}$ над произведением алфавитов $\Sigma_1 \times \Sigma_2$. Тогда можно определить автоматы (только с одним состоянием), преобразующие последовательность $\{z_i\}$ в $\{x_i\}$ и $\{y_i\}$.

С другой стороны, предположим, что $\{w_i\}$ — такая последовательность, что существуют автоматы T_1, T_2, преобразующие ее соответственно в последовательность $\{x_i\}$, задержанную некоторым образом, и в последовательность $\{y_i\}$, также некоторым образом задержанную. Можно предположить, что эти задержки равны. Тогда можно описать автомат, «параллельно» соединяющий T_1 и T_2 и преобразующий $\{w_i\}$ в последовательность $\{(x_i, y_i)\}$ с той же самой задержкой.

Отметим, что такая же конструкция приводится в [2, стр. 382], где последовательность $\delta(a)$ определяется как последовательность кодов вида $2^{\alpha(a)}3^{\beta(a)}$ упорядоченных пар $(\alpha(a), \beta(a))$, где α, β — последовательности целых чисел.

4. КЛАССЫ, РАСПОЛОЖЕННЫЕ НИЖЕ ОПРЕДЕЛЕННОГО КЛАССА

Проанализируем структуру частично упорядоченного множества классов, расположенных ниже ($\leqslant$) класса последовательности

$$x = \{101001000100001 \ldots\}.$$

Единицы в этой последовательности занумерованы, так что можно говорить об i-й единице для $i = 1, 2, 3, \ldots$. За i-й единицей следует в точности i нулей.

Для целей этого анализа назовем последовательность $y = \{y_n\}$ *специальной*, если она совпадает с последовательностью x, за исключением того, что те единицы, чьи номера образуют некоторую периодическую последовательность, заменены нулями. Сама последовательность x считается специальной. Формально последовательность y специальна, если существуют целые числа s и

$$0 \leqslant a_1 < a_1 < \ldots < a_s < M,$$

определяющие y по формулам

$y_n = 1$, если $x_n = 1$ и x_n есть i-я единица,
где $i \equiv a_1$ или a_2 или ... или $a_s \bmod M$;
$y_n = 0$ в противном случае.

Теорема. *Если y — специальная последовательность, а x — только что описанная, то $[y] \leqslant [x]$. Обратно, каждый класс ниже класса $[x]$ содержит в точности одну специальную последовательность.*

Доказательство. Если дана специальная последовательность y, определенная числами s, a_1, ..., a_s, M, то легко описать автомат, считающий входные единицы по модулю M и выдающий в точности те единицы, чьи номера находятся в нужном классе вычетов. Следовательно, $[y] \leqslant [x]$.

Обратно, пусть (T, s_0) — инициальный автомат, преобразующий x в y. Если T находится в произвольном состоянии s_i, то входная почти периодическая последовательность $\{100000\ldots\}$ проводит T через некоторую почти периодическую последовательность состояний S_i. Пусть N_i — длина начальной непериодической части этой последовательности состояний и M_i — период периодической части. Выберем N больше всех N_i и в качестве M возьмем какое-то общее кратное всех M_i. Тогда последовательность состояний S_i и соответствующую выходную последовательность Y_i можно рассматривать как содержащие начальную часть длины $N - 1$ и периодическую часть с периодом M; эти параметры не зависят от начального состояния s_i.

Входная последовательность x состоит из последовательно расположенных $(k+1)$-блоков B_k, образованных единицей и следующими за ней k нулями, $k = 1, 2, \ldots$. Пусть T_k — соответствующий блок последовательности состояний автомата (T, s_0) и C_k — соответствующий блок выходной последовательности y. Блоки T_k и C_k должны быть начальными сегментами одной из вышеописанных почти периодических последовательностей состояний и выходных символов соответственно, а именно S_i и Y_i, если s_i — состояние автомата к началу поступления блока B_k.

В последующих рассуждениях мы будем предполагать всюду, что $k > N + M$, так что по крайней мере одна полная копия периодической части включена во все начальные сегменты почти периодических последовательностей, которые уже упоминались.

Если T_k и $T_{k'}$ — начальные сегменты последовательности состояний S_i и $k \equiv k' (\mathrm{mod}\, M)$, то финальные состояния в T_k и $T_{k'}$ совпадают в силу периодичности. Значит, непосредственно следующие блоки T_{k+1} и $T_{k'+1}$ также являются начальными сегментами некоторой общей последовательности S_j, и, разумеется, $k + 1 \equiv k' + 1 (\mathrm{mod}\, M)$. Отсюда вытекает, что последовательность тех S_i, для которых T_k — начальные сегменты, почти периодическая и, таким образом, есть последовательность тех Y_i, в которых выходные блоки C_k образуют начальные сегменты.

Теперь допустим, что C_k — начальный сегмент последовательности Y_i и блок, состоящий из C_k, после которого идет C_{k+1}, также является начальным сегментом для Y_i. Тогда C_{k+1} называется *невидимым*; в противном случае — *видимым*. Будет ли C_{k+1} видимым, очевидно, зависит только от Y_i и $k (\mathrm{mod}\, M)$, так как $k (\mathrm{mod}\, M)$ определяет ту точку в периодике Y_i, где оканчивается C_k, а также последовательность Y_j, для которой C_{k+1} сам является начальным сегментом. Так как Y_i образуют почти периодическую последовательность, то такова же и последовательность пар $(Y_i, k (\mathrm{mod}\, M))$. Следовательно, свойство видимости (невидимости) почти периодическое.

Если нет видимых блоков, то выходная последовательность почти периодическая, и ничего не надо доказывать. Если есть видимые блоки, то для каждого такого C_{k+1} пусть d_k будет номером первого из всех тех мест в C_{k+1}, где C_k, за которым следует C_{k+1}, отличается от соответствующего места в Y_i. Ясно, что d_k зависит только от пары $(Y_i, k (\mathrm{mod}\, M))$, так что различных значений будет лишь конечное число. Пусть d — их максимум. Тогда легко определить конечный автомат, преобразующий выходную последовательность в специальную, задержанную на d, в которой сохранены только те единицы, которые соответствуют началам видимых блоков. Кроме того, можно определить автомат, преобразующий без задержки эту специальную последовательность в первоначальную выходную последовательность. Следовательно, выходная последовательность эквивалентна специальной последовательности.

Осталось проверить лишь одно утверждение теоремы: никакой класс не содержит различных специальных последовательностей. Если последовательности y и z специальны и различны, то в y бесконечно много вхождений единиц, которых нет в z (или обратно). Число подряд идущих нулей по каждую сторону

от этих единиц неограниченно. Далее, никакой автомат, скажем с k состояниями, не может преобразовать входной блок из $k+1$ нулей в блок, состоящий из k нулей, за которыми идет единица; следовательно, никакой автомат не может восстановить все отсутствующие единицы из z, чтобы получить y. Очевидное уточнение этих рассуждений показывает, что последовательность y с постоянной задержкой также нельзя получить. Следовательно, $[z] \not\geqslant [y]$, и потому $[z] \neq [y]$.

5. НЕ СУЩЕСТВУЕТ МАКСИМАЛЬНЫХ КЛАССОВ

В этом разделе мы покажем, что не существует таких последовательностей y, для которых $[z] \leqslant [y]$ при всех z. Этот результат можно вывести из известного факта, что в частичном упорядочении степеней рекурсивной неразрешимости нет максимального элемента, однако методы приводимого ниже элементарного доказательства интересны сами по себе.

В последующих рассуждениях мы предполагаем, что алфавит фиксирован. Вопрос нахождения такой последовательности z, что $[z] > [y]$, тривиален, если z можно записывать в большем алфавите, чем y. Например, если $y = \{y_i\}$, то z может быть $\{(y_i, w_i)\}$, где $w = \{w_i\}$ — одна из несчетного множества последовательностей из нулей и единиц, для которой $[y] \not> [w]$. (Здесь последовательность z записана в алфавите, мощность которого в два раза превосходит мощность алфавита для y.)

Очевидно, надо предположить, что этот алфавит имеет по меньшей мере два символа.

Будем называть последовательность символов из некоторого алфавита *полной*, если в ней для любого k встречается каждый блок длины k из символов этого алфавита. Отсюда следует, что каждый блок встречается бесконечно часто: например, для произвольного n встретится (nk)-блок, состоящий из n последовательных копий k-блока.

Лемма. *Если последовательность y не полная, то ее класс эквивалентности не максимален.*

Доказательство. Пусть B будет k-блоком, который не встречается в y, а A будет k-блоком, который встречается бесконечно много раз среди «целочисленно расположенных k-блоков», т. е. среди k-блоков, начинающихся с позиций 0, k, $2k$, $3k$, ... в последовательности y. Мы построим такую последовательность z, что $[z] \geqslant [y]$, но $[z] \not\leqslant [y]$.

Последовательность z будет образована из y заменой целочисленно расположенных копий блока A копиями блока B. Легко описать автомат, преобразующий z в последовательность

y, задержанную на k: начиная вновь работу после просмотра каждого из целочисленно расположенных входных k-блоков, он должен воспроизводить его неизменным, если это не блок B, и выдавать A в противном случае. Это доказывает, что $[z] \geqslant [y]$.

Покажем, как выбрать для замены копии блока A, чтобы убедиться, что $[z] \not\leqslant [y]$. Мы должны построить z так, чтобы ни для одного инициального автомата T (над специальным алфавитом) и ни для одной задержки d вход y не давал на выходе последовательность z, задержанную на d. Однако множество пар (T, d) счетно. Построим z по шагам следующим образом. Для $i = 1, 2, 3, \ldots$ пусть (T, d) будет i-й такой парой. Заменим i-ю из целочисленно расположенных копий блока A на B, если это необходимо, чтобы гарантировать, что часть последовательности z после этого k-блока, задержанная на d, отличается от начального сегмента выхода инициального автомата T с входной последовательностью y.

Доказав, что единственно возможные максимальные классы — это те, которые содержат полные последовательности, покажем, что никакой из таких классов не максимален. Поэтому максимальных классов не существует.

Теорема. *Если класс содержит полную последовательность, то он не максимален.*

Доказательство. (В этом доказательстве удобно считать, что алфавит содержит по меньшей мере три символа, скажем 0, 1, 2. Если это не так, заменим слово «символ» в доказательстве на «пара последовательных символов». Так как алфавит имеет по меньшей мере два символа, он имеет не менее четырех и, значит, не менее трех пар символов.)

Легко определить инициальный автомат, заменяющий символ, который следует за каждой максимальной цепочкой последовательных копий символа 2, символом, который следовал за предыдущей такой цепочкой; при первой замене вставляется произвольно выбранный символ, скажем 0. (Если алфавит имеет K символов, можно определить инициальный автомат с $2K$ состояниями, которые соответствуют парам (B, X), где B — «истина» или «ложь» в зависимости от того, был последний символ 2 или не 2, а X — символ, который следовал после предыдущей максимальной цепочки двоек. Начальным является состояние (ложь, 0).) Для любой входной последовательности x пусть Dx будет выходом этого инициального автомата. Покажем, что если последовательность x полна, то $[Dx] \not\geqslant [x]$, откуда $[Dx] < [x]$, так как очевидно, что $[Dx] \leqslant [x]$.

Пусть y — произвольная последовательность. Тогда $[y] = [Dx]$ для подходящим образом выбранной последовательно-

сти x, а именно последовательности, полученной заменой символа, который следует за каждой максимальной цепочкой из двоек, символом, расположенным за следующей такой цепочкой (или, скажем, нулем, если в этой последовательности нет больше двоек). (В самом деле, последовательности y и Dx согласуются, за исключением, может быть, символа, следующего за первой такой цепочкой.) Кроме того, если последовательность y полна, то, конечно, полна и x. Следовательно, если дана какая-нибудь полная последовательность y, то

$$[y] = [Dx] \leqslant [x].$$

Покажем теперь, что $[Dx] < [x]$. Для этого нам понадобится лемма, которую мы докажем ниже.

Лемма. *Для любого конечного автомата T и любого неотрицательного целого d существует блок B, длина которого больше d и такой, что независимо от того, в каком состоянии автомат T начинает работу, его выход, если на вход подана последовательность DB, отличается от B, задержанного на d, по меньшей мере в одной из позиций, где они оба определены.*

С помощью этой леммы легко закончить доказательство. Пусть x — любая полная последовательность. Тогда каждый блок B встречается в ней бесконечно часто. Пусть (T, s_0) — любой инициальный автомат и d — произвольная задержка. Тогда выход автомата (T, s_0), которому на вход подана Dx, отличается от x, задержанной на d, бесконечно часто, в частности по меньшей мере один раз в каждом вхождении блока B, соответствующем по лемме этой паре (T, d). Следовательно, $[Dx] \not\geqslant [x]$.

Доказательство леммы. Пусть даны автомат T и положительное целое число d. Занумеруем состояния автомата T: $s_1, \ldots, s_n$. Построим блок B как последовательность n копий блоков $X = 2000 \ldots 0$ и $Y = 2100 \ldots 0$, каждый длины $d + 2$: независимо от того, начинается B с X или Y, DB будет начинаться с X (из-за произвольности выбора символа 0, что было сделано для того, чтобы заменить символ после первой цепочки из двоек). Предположим, что T начинает работу в состоянии s_1, и рассмотрим последний выходной символ для входного блока X. Если это 0, то пусть B начинается с копии блока Y, в противном случае — с копии блока X.

Для завершения индуктивного построения допустим, что в B мы уже определили первые k подблоков X и Y, рассматривая начальные состояния от s_1 до s_k. DB начинается с X, за которым идет уже определенная часть блока B; поэтому первые

$(k+1)(d+2)$ символов уже известны. Теперь допустим, что T начинает работать в состоянии s_{k+1} с этим уже известным блоком DB на входе. Рассмотрим последний символ выхода. Если это 0, то пусть следующая часть блока B будет Y, в противном случае X. Продолжаем так, пока не будет использовано последнее состояние s_n (поэтому B имеет длину $n(d+2)$).

Если блок B определен таким способом и T начинает работать в состоянии s_k с DB на входе, то выход отличается от B, задержанного на d, по меньшей мере в $k(d+2)$ позициях.

6. АТОМ

В разд. 4 мы проанализировали некоторое подмножество частично упорядоченного множества и заметили, что в нем между любыми двумя классами $[x] > [z]$ находится такой класс $[y]$, что $[x] > [y] > [z]$. Здесь мы покажем, что в общем случае это неверно. В частности, построим последовательность (над алфавитом 0,1), класс эквивалентности которой лишь атомарно больше, чем наименьший класс, т. е. $[x] > [0]$ и не существует такого класса $[y]$, что $[x] > [y] > [0]$.

Определим последовательность блоков I_i, где каждый блок I_i — собственный начальный сегмент следующего блока I_{i+1}. Последовательность x будет определена как предел этой последовательности блоков, т. е. как единственная последовательность, для которой каждый блок I_i является начальным сегментом.

Блоки I_i вместе с некоторыми блоками A_i, B_i будут определены индукцией по $i = 1, 2, 3, \ldots$. Детали построения даны в конце раздела. Сейчас мы сформулируем ряд свойств, которыми обладают эти блоки.

(1) A_1 начинается с 0, B_1 — с 1, и они имеют одинаковую длину.

(2) Каждый блок A_{i+1}, B_{i+1} представляет собой последовательность копий меньших блоков A_i и/или B_i, начинающуюся соответственно с A_i и B_i. Они имеют одинаковую длину, большую i. Ни тот, ни другой не являются частью никакой периодической последовательности периода i.

(3) I_{i+1} состоит из I_i и следующими за ним копиями блока A_i и/или B_i.

(4) Перенумеруем инициальные автоматы (T, s_0). Если вход I_i переводит i-й автомат (T, s_0) в финальное состояние s, то, начиная работать в состоянии s, T возвращается в финальное состояние s как под действием входного блока A_i, так и под действием входного блока B_i.

Последовательность $x = \lim I_i$ не является почти периодической. Допустим обратное, т. е. что она имеет конечный период i. Последовательность x можно рассматривать как начальный блок I_{i+1}, за которым идет последовательность копий блоков A_{i+1} и B_{i+1}. Но по свойству 2 ни A_{i+1}, ни B_{i+1} не составляют части никакой периодической последовательности периода i.

Допустим, что у нас есть такая последовательность y, что $[x] \geqslant [y]$. Заменяя, если необходимо, y другим членом из класса эквивалентности $[y]$, можно предположить, что некоторый инициальный автомат (T, s_0) выдает последовательность y, если на его вход подана последовательность x. Пусть i будет номером автомата (T, s_0) в нашей нумерации инициальных автоматов.

Последовательность x можно рассматривать как содержащую начальный блок I_i, за которым идет последовательность копий блоков A_i и B_i одинаковой длины, скажем N. Как только начинается один из этих блоков A_i или B_i, T находится в одном и том же состоянии, скажем s. Следовательно, выходная последовательность y, если уже пройден начальный сегмент I_i входа, содержит последовательность копий только двух блоков, каждый длины N, скажем C (ответ на входной блок A_i, когда T начинает работать в состоянии s) и D (аналогично ответ на B_i).

Очевидно, что $[x] \geqslant [y] \geqslant [0]$. Надо показать, что одно из этих неравенств на самом деле является равенством.

Существуют две возможности. Если C и D совпадают, то последовательность y почти периодическая, поэтому $[y] = [0]$. Если C и D не совпадают, то можно описать инициальный автомат, заменяющий C на A_i и D на B_i с задержкой не более чем на N. Таким образом, x и y эквивалентны: $[x] = [y]$.

Доказательство будет закончено, когда мы опишем конструкцию блоков, использованных в нем. Определим I_{i+1}, A_{i+1} и B_{i+1} по индукции. Начнем с определения I_0 как пустого блока (длины нуль), A_0 — как символа 0 и B_0 — как символа 1.

Пусть (T, s_0) будет $(i+1)$-м инициальным автоматом. Допустим, что $(A|B)^*$ обозначает множество конечных непустых блоков, которые являются последовательностями копий блоков A_i и/или B_i. Для любого состояния s автомата T пусть $\langle s \rangle$ будет множеством состояний, которые являются финальными состояниями автомата (T, s) для входных блоков из множества $(A|B)^*$.

Пусть s_1 — финальное состояние автомата (T, s_0) для входного блока I_i. Выберем состояние $s_2 \in \langle s_1 \rangle$ так, чтобы мощность множества $\langle s_2 \rangle$ была минимальна. Тогда $s_2 \in \langle s_2 \rangle$. Пусть I_{i+1} состоит из I_i и следующего за ним такого блока из множества $(A|B)^*$, что s_2 — финальное состояние автомата (T, s_0) для входа I_{i+1}.

По предположению индукции A_i и B_i имеют длину по меньшей мере i. Они отличаются первыми символами. Значит, если A_iA_i — часть периодической последовательности периода i, то A_iB_i нет. Пусть A_iX_i — одна из этих последовательностей и она непериодическая. Пусть s_A будет финальным состоянием автомата (T, s_2) для входного блока A_iX_i. Очевидно, что $s_A \in \langle s_2\rangle \subseteq \langle s_1\rangle$, поэтому $\langle s_A\rangle = \langle s_2\rangle$ в силу минимальности $\langle s_2\rangle$. Следовательно, $s_2 \in \langle s_A\rangle$. Пусть A'_{i+1} — элемент множества $(A|B)^*$, который начинается с A_iX_i и для которого финальное состояние автомата (T, s_2) есть s_2.

Так как A_iX_i не составляет части никакой периодической последовательности периода i, то и последовательность A'_{i+1} также обладает этим свойством. Ее длина больше $2i$, а значит, больше $i + 1$.

Аналогично пусть B'_{i+1} — элемент множества $(A|B)^*$, начинающийся с B_i, не составляющий части никакой периодической последовательности периода i, и для него s_2 — финальное состояние автомата (T, s_2).

Пусть A'_{i+1} имеет длину k, а B'_{i+1} — длину l. Обозначим через m наименьшее общее кратное чисел k и l, и пусть A_{i+1} состоит из m/k копий блоков A'_{i+1}, а B_{i+1} — из m/l копий блока B'_{i+1}. Тогда A_{i+1}, B_{i+1} имеют одинаковое строение и обладают свойством «от s_2 к s_2», которым обладали вставленные блоки; более того, они имеют одинаковую длину.

Так как A_1 начинается с 0, а B_1 — с 1, то по индукции так же начинаются соответственно A_{i+1} и B_{i+1}.

Это завершает проверку требуемых свойств.

СПИСОК ЛИТЕРАТУРЫ

1. Gill A., Introduction to the theory of finite-state machines, McGraw-Hill, New York, 1962. (Русский перевод: Гилл А., Введение в теорию конечных автоматов, изд-во «Наука», М., 1966.)
2. Kleene S. C., Post E. L., The upper semi-lattice of degrees of recursive unsolvability, *Ann. Math.*, **59** (1954), 379.
3. Mealy G. H., A method for synthesizing sequential circuits, *Bell System Tech. J.*, **34** (1955), 1045.
4. Moore E. F., Gedanken-experiments on sequential machines, «Automata Studies», *Ann. Math. Stud.*, **34** (1956), 129—153. (Русский перевод: Мур Э. Ф., Умозрительные эксперименты с последовательностными машинами, в сб. «Автоматы», ИЛ, М., 1956, стр. 179—210.)

Набросок математической теории вычислений[1])

Д. Скотт

Побудительный мотив попытки сформулировать математическую теорию вычислений кроется в необходимости снабдить *математической семантикой* языки программирования высокого уровня. В этом контексте слово «математическая» должно контрастировать с таким термином, как «операциональная». Так, математическим содержанием *процедуры* следует считать *функцию* от элементов того типа данных, какой связан с входными переменными, принимающую значения элементов того типа данных, какой связан с выходом. С другой стороны, операциональное понимание всегда будет предусматривать некий *след* полной *истории* вычисления, вытекающей из той последовательности действий, которая предписывается заданным определением процедуры, и будет также включать явный финитный выбор *представлений* данных в той или иной форме, близкой в конечном счете к двоичным образам. Существенно то, что функции не зависят от способов их вычисления и, значит, «проще», чем явно порожденные шаг за шагом последовательности операций над представлениями. Когда даются точные определения операциональной семантики, всегда нужно сделать более или менее произвольные выборы схем для каталогизации промежуточных результатов и связей между фазами вычисления (ср. формальные определения таких языков, как ПЛ/1 и АЛГОЛ 68). Истинное же «понимание» программы в значительной степени не зависит от этих выборов. Математическая семантика имеет тенденцию избегать таких иррелевантностей и больше подходит к изучению вопросов типа *эквивалентности* программ.

Способствовать выработке более «абстрактного» и «ясного» подхода к семантике — очень похвально, но, претендуя на в какой-то степени хороший замысел, нельзя полностью игнорировать операциональные аспекты. Причина очевидна: в конце концов, программа должна прогоняться на машине — машине, которая не обладает преимуществами человеческого «абстрактного» понимания и должна оперировать только конечными конфигурациями. Поэтому математическая семантика, будучи пер-

[1]) Scott D., Outline of a mathematical theory of computation, Proc. Princeton Conf. on Information Sci., 1971.

вой главной частью полного и строгого определения языка программирования, должна естественно приводить к некоему операциональному моделированию абстрактных сущностей, что, если оно сделано правильно, устанавливает практическую ценность языка и необходимо для полного представления.

Будем рассуждать пока только о функциях. Ясно, что о данной математически определенной функции мы можем знать, что она *вычислима*, но не знать, *как* она практически вычисляется — точно так же, как можно знать, что бесконечный ряд сходится, не имея ни малейшего представления о том, как найти его сумму. Даже тогда, когда данное абстрактное определение функции достаточно для полного ее задания, мы не можем сказать, что *знаем* функцию, пока не обнаружен алгоритм ее вычисления. (Причем даже тогда, когда наше знание в некоторой степени «косвенное» или «потенциальное».) Вывод таков: *настоящая* теория вычислений должна не только давать абстракции (выявлять вычислимость), но также и указывать их «физические» реализации (как вычислять их).

Новым в нашей теории является как раз выдвижение этих *абстракций*, тогда как способы их реализации, техника воплощения уже в течение ряда лет известны, и это продемонстрировано в компиляторах большой сложности. Конечно, новые понятия могут потребовать новых (или подсказать новые) методов воплощения. Но это еще предстоит увидеть. Отметим, однако, один существенный момент: до тех пор пока под рукой нет общепринятого математического определения языка, кто может сказать, что предложенное воплощение *корректно*? Теперь часто полагают, что смысл данного языка заключен в каком-либо одном конкретном компиляторе для него. Но эта мысль не верна: «один и тот же» язык может иметь много «различных» компиляторов. Человек, написавший один из этих компиляторов, имел (можно надеяться) ясное понимание языка (оно руководило им), и цель математической семантики — сделать это понимание «видимым». «Видимость» эта должна достигаться абстрагированием центральных идей в математические сущности, которыми можно затем «манипулировать» обычным математическим образом. Если бы подход, ориентированный на компиляторы (даже на компиляторы, «прокручиваемые» на «абстрактной» машине), был ясным и прозрачным — что на самом деле не так, — то все равно было бы интересно выведенные абстракции связать в теорию, использующую стандартную математическую практику.

Эта желаемая математическая теория требует, как нам кажется, некоторых новых структуральных понятий, нового взгляда на природу *типов данных* и *функций* (отображений) из одного типа данных в другой тип. Более того, рассуждая

о понятиях «высшего уровня» в языках программирования (таких, как, например, понятие процедуры), мы скоро приходим к тому, что пространство функций также должно рассматриваться как имеющее тип данных. Так как функция (скажем, отображение целых чисел в целые) есть, вообще говоря, сама по себе бесконечный объект, возникает необходимость в некоторой идее конечного *приближения* (аппроксимации) — так же, как мы поступаем в случае вещественных чисел. Здесь мы уже сталкиваемся с такими операционально определенными понятиями функции, которые, кажется, *не имеют* математических прообразов. В частности, не ново разрешать в языках программирования *неограниченные* процедуры, которые могут использовать любые процедуры в качестве своих аргументов и могут производить в качестве значений неограниченные процедуры. Математически это равносильно допущению функции, определенной на *всех* допустимых функциях как аргументах (некоторого рода суперфункционал) и применимой даже к *самой себе* как к аргументу. В настоящее время нет математической теории функций, позволяющей столь свободное обращение с понятием функции, если, конечно, оставаться в рамках непротиворечивости. Главным *математическим* нововведением настоящего исследования является создание собственной математической теории функций, которая служит указанной выше цели (непротиворечиво!) и которая может быть использована как *математический* проект, дающий «корректный» подход к семантике.

Следует сразу же особо подчеркнуть, что проблема *самоприменимости* возникает при интерпретации языков программирования, притом более критически, чем при рассмотрении (в каком-то смысле непрактичных) неограниченных процедур. Проблема касается прослеживания *побочных эффектов* и запоминания команд. Прежде всего, что такое память? Физически мы имеем несколько замечательных ответов, но математически они сводятся к тому, что память есть просто *назначение* (функция), которое связывает *содержания* с *местами* (адресами). Точнее, (текущее) состояние памяти, назовем его σ, математически есть функция

$$\sigma\colon L \to V,$$

которая каждому месту $l \in L$ (где L — множество всех мест) назначает (текущее) содержание $\sigma(l) \in V$ (где V — множество всех допустимых значений для содержаний). Пусть Σ — множество всех состояний. Что такое *побочный эффект*? Очевидно, *изменение* состояния. Что такое *команда*? Это требование побочного эффекта. Более математически, команда есть функция

$$\gamma\colon \Sigma \to \Sigma,$$

преобразующая (старые) состояния в (новые) состояния.

Вопрос: может ли команда запоминаться? Ответ: конечно. Это мы делаем *операционально* все время. Вопрос: оправданно ли это математически? Ответ: посмотрим. Пусть σ — текущее состояние памяти, а $l \in L$ — место, где запомнена команда. Тогда $\sigma(l)$ — команда, т. е.

$$\sigma(l)\colon\ \Sigma \to \Sigma.$$

Следовательно, $\sigma(l)(\sigma)$ вполне определено. Но что это такое? Здесь мы имеем лишь незначительный шаг в сторону от проблемы самоприменимости $p(p)$ для «неограниченных» процедур, и это как раз трудно оправдать математически. Конечно, при операциональном подходе мы не запоминаем команду саму по себе как функцию, мы запоминаем ее как «кодовое слово» или как «кусок текста», что для команд устанавливается недвусмысленно. Но чтобы «вытянуть» формальное описание того, как это работает, — особенно в случае сложных команд, зависящих от многих параметров, — мы должны войти в большинство «проклятых» вопросов семантики языков программирования, и не видно действительно концептуально удовлетворительного способа выпутаться из них.

Переходя к более специальным вещам, спросим: как точно определить, что такое *тип данных*? Упрощая дело, можно отождествить тип данных с множеством D всех объектов этого типа. Но это слишком сильное упрощение: ведь объекты обладают структурой и находятся в некоторых отношениях друг к другу, так что тип есть нечто большее, чем множество. Это структурирование не надо смешивать с идеей *структур данных* (списки, деревья, графы и т. п.); они появляются позже. Род структуры, который мы здесь имеем в виду, более примитивен и общ и связан с основным смыслом *приближения*. Пусть x, $y \in D$ — два элемента типа данных. Тогда идея заключается не в том, чтобы думать о них как о совершенно *разделенных* сущностях из-за того, что они могут быть *различными*. Вместо этого y может быть *лучшей* версией, чем x, в попытке что-то выразить. Давайте запишем отношение

$$x \sqsubseteq y,$$

интуитивно означающее, что y *совместно с* x и (возможно) *точнее* x. Такое интуитивно понимаемое отношение существует на большинстве типов данных естественным образом, и один из тезисов настоящей статьи есть то, что тип данных следует *всегда* снабжать таким отношением. Чтобы «подогнать» некоторые стандартные идеи, возможно, потребуется придать нашим мыслям определенное направление, но, право, стоит потратить усилия для единой трактовки различных типов.

Итак, в дальнейших рассуждениях принимается, что типы (по меньшей мере) структурированы отношениями $\sqsubseteq$. Что абстрактно можно сказать о каждом таком отношении? В соответствии с интуитивным пониманием следует считать, что отношение $\sqsubseteq$ *рефлексивно*, *транзитивно* и *антисимметрично*. Мы переводим это в аксиому.

Аксиома 1. Тип данных есть *частично упорядоченное множество.*

Конечно, это немного: частично упорядоченные множества весьма общи, но все же это некоторый шаг вперед. Следующий шаг связан с отображениями.

Предположим, что D и D' — два типа данных (с соответствующими частичными порядками $\sqsubseteq$ и $\sqsubseteq'$). Пусть $f\colon D \to D'$ — какое-либо разумное отображение одного типа в другой. Следует ли говорить вообще что-нибудь о свойствах отображений? Да, следует. Пусть $x, y \in D$ и $x \sqsubseteq y$. Если бы f была функцией, заданной с помощью программы, то функция была бы чувствительна к точности своих аргументов (входов) в следующем смысле: когда *вход точнее*, тогда и *выход точнее*. Запись

$$x \sqsubseteq y \quad \textit{влечет} \quad f(x) \sqsubseteq' f(y)$$

означает, что по отношению к частичному порядку функция f монотонна. Сформулируем это в виде аксиомы.

Аксиома 2. Отображения между типами данных *монотонны.*

Заметим, что такое условие легко обобщается на функции нескольких переменных, даже если переменные смешанных типов.

В численных вычислениях аксиома 2 иногда отвергается, но это связано с неверным употреблением слова «*точность*». Верно, что известны хорошие асимптотические алгоритмы, дающие *лучшие* ответы при *более грубой* точности, но их следует рассматривать как функции *двух* переменных: обычные входные данные вместе с параметром, указывающим степень точности, или, лучше сказать, число «членов разложения». Может случиться, что, взяв больше членов, мы только разрушим уже полученное хорошее приближение, но заметим, что теперь вход и число членов разложения предполагаются *совершенно* известными. Понятие точности, которое мы связываем с отношением $\sqsubseteq$, несколько иное и *не* зависит от этого предположения. Может быть, уместнее говорить об *информации*; так что $x \sqsubseteq y$ означает, что x и y приближают одну и ту же сущность, но y дает *больше*

информации о ней. Это означает, что мы хотим допустить «неполные» сущности, такие, как x, содержащие только «частичную» информацию. (Этот способ в численных вычислениях называется *интервальным анализом*, но мы не имеем здесь возможности входить в детали.) Допущение частичности для аргументов и значений функций приводит к хорошему эффекту, что наши функции становятся также *частичными*, ибо даже если аргументы полные, то значения могут быть частичными. Это необходимо при рассмотрении алгоритмически заданных функций, так как для некоторых комбинаций аргументов алгоритм может «не сходиться». Тогда, исходя из этой точки зрения, не может быть численной функции вида, совместимого с аксиомой 2, которая отображает «частичные» числа в показатели степени точности. Но это не недостаток, так как можно увидеть, проверяя детали метода, что имеется достаточно много монотонных функций.

Теория, основанная на аксиомах 1 и 2, была бы слишком абстрактной, хотя и небессодержательной. Для тех приложений, которые мы имеем в виду, нужны более специальные свойства приближений. Предположим, задана последовательность приближений

$$x_0 \sqsubseteq x_1 \sqsubseteq \ldots \sqsubseteq x_n \sqsubseteq x_{n+1} \sqsubseteq \ldots .$$

Тогда разумно предположить, что x_n стремится к *пределу*. Обозначим этот предел через y и запишем

$$y = \bigsqcup_{n=0}^{\infty} x_n,$$

так как предел в смысле частичного порядка естественно берется как наименьшая верхняя грань (н. в. г.). Если представить себе члены последовательности, как дающие все больше и больше информации, тогда предел имеет вид «объединения» отдельных составляющих. Фактически ради математической простоты мы предположим, что *всякое* подмножество типа данных имеет наименьшую верхнюю грань, и выражается как

Аксиома 3. Тип данных есть *полная решетка* по своему частичному порядку.

Как хорошо известно, существование произвольных наименьших граней *влечет* существование наибольших нижних граней (н. н. г.) — вот почему мы говорим, что имеем полную решетку.

Последнее предположение требует некоторого разъяснения. Для $x, y \in D$ возможно, что либо они являются приближениями одной и той же «полной» сущности, либо они в какой-то степени

«несовместны». В обоих случаях мы предполагаем, что они имеют н. в. г., или *объединение* $x \sqcup y \in D$. Хуже того, мы предполагаем, что *всё* D имеет н. в. г., которую мы обозначаем $\top$. Это $\top$ есть «верх» решетки, наибольший элемент частичного порядка $\sqsubseteq$. Его не надо рассматривать как «полный» элемент, но как «сверхопределенный элемент». Следовательно, рассматривая равенство

$$x \sqcup y = \top,$$

мы можем понимать его интуитивно как то, что x и y *несовместны*. Следует отличать это от гораздо более слабого отношения быть *несравнимыми*, что просто записывается как $x \not\sqsubseteq y$ и $y \not\sqsubseteq x$.

Н. в. г. *пустого* подмножества нашего типа данных D есть элемент $\bot \in D$. Это есть «низ» решетки, наименьший элемент частичного порядка. Его можно рассматривать как самый «недоопределенный» элемент. Н. в. г. множества всех нижних граней для подмножества $X \subseteq D$ есть н. н. г. $\sqcap X \in D$. Для $x, y \in D$ мы имеем пересечение $x \sqcap y = \sqcap \{x, y\} \in D$. Равенство

$$x \sqcap y = \bot$$

можно понимать так, что x и y *не связаны* в том смысле, что они не имеют перекрывающихся частей информации.

После того как мы предположили, что тип данных допускает пределы, мы должны пересмотреть наш взгляд на функции. Если функция *вычислима* в каком-либо интуитивном смысле, то для извлечения «конечного» объема информации из какого-либо значения функции мы должны запросить лишь «конечный» объем информации об аргументах. Наше понятие информации скорее *качественное*, чем *количественное*. Однако еще возможно выразить это фундаментальное ограничение на функции: а именно, функции должны *сохранять пределы*.

Аксиома 4. Отображения между типами данных *непрерывны*.

Во всей общности мы можем это точно выразить в терминах направленных множеств. Подмножество $X \subseteq D$ *направлено*, если каждое конечное подмножество X имеет по крайней мере одну верхнюю грань в X. (Заметим, что направленное множество не пусто.) Функция $f\colon D \to D'$ *непрерывна* в том и только том случае, если для любого направленного множества $X \subseteq D$ мы имеем

$$f(\sqcup X) = \sqcup \{f(x)\colon x \in X\}.$$

Не все н. в. г. следует рассматривать как *пределы*, а только н. в. г. направленных множеств. Можно привести много приме-

ров, показывающих, что неразумно требовать от функций сохранения *произвольных* н. в. г. Кроме того, совершенно неразумно предполагать, что функция сохраняет н. н. г.: нельзя ожидать никакой «гладкости» при убывании информации. Заметим, что понятие непрерывности легко распространить на функции многих переменных. На самом деле оказывается, что для непрерывности функции по всем переменным *одновременно* достаточно потребовать ее непрерывности по каждой переменной *в отдельности*.

Для первоначального общего наброска теории сделано достаточно много. Однако все это слишком еще абстрактно, ибо хотя некоторые существенные *свойства* вычислимых функций и были выделены, все же возможность «физической» *реализации* не предполагалась нами ни в какой форме. Это необходимо восполнить. Задача заключается в концентрации внимания в точности на тех типах данных, элементы которых могут быть приближены «конечными конфигурациями», представимыми в машинах. Значит, нужно уточнить понятие «конечного объема информации».

Для решения этой проблемы попытаемся применить топологический подход. Во всяком случае, наше предыдущее упоминание *пределов* и *непрерывности* должно подсказывать существование топологических идей в основании нашей теории. В самом деле, всякий тип данных D, удовлетворяющий аксиомам 1 и 3, можно рассматривать как топологическое пространство. Чтобы задать на множестве топологию, нужно сказать, какие подмножества являются *открытыми*. В случае типа данных D для того, чтобы подмножество $U \subseteq D$ было открытым, должны выполняться следующие два условия:

(U_1) *если* $x \in U$ *и* $x \sqsubseteq y$, *то* $y \in U$,

(U_2) *если* $X \subseteq D$ *направлено и* $\bigsqcup X \in U$, *то* $X \cap U \neq \varnothing$.

Легко проверить, что D становится, таким образом, топологическим пространством и что $f\colon D \to D'$ непрерывна в смысле *сохранения пределов* тогда и только тогда, когда она непрерывна в *топологическом* смысле. Для $x, y \in D$ мы пишем

$$x \prec y,$$

понимая под этим то, что y принадлежит топологической *внутренности* верхнего сечения, определяемого x, т. е. множества

$$\{x' \in D\colon \quad x \sqsubseteq x'\}.$$

Это отношение не является иррефлексивным, как может показаться сначала, так как в некоторых типах данных могут существовать *изолированные* элементы x, для которых тогда

$x \prec x$. Мы также пишем

$$x \preccurlyeq y$$

для обозначения того, что наибольшая верхняя грань топологической внутренности верхнего сечения, определяемого x, есть $\sqsubseteq y$. Таким образом, в списке

$$x \prec y, \quad x \preccurlyeq y \quad \text{и} \quad x \sqsubseteq y$$

каждое последующее отношение слабее предыдущего.

Топологические пространства, подобные пространству вещественных чисел (топология которых несколько отличается от наших пространств типов данных), подсказывают нам рассмотреть возможность иметь *плотное* множество в нашем пространстве в том смысле, что все другие элементы пространства получаются как пределы направленных подмножеств плотного множества. Мы называем такое подмножество *базисом*. Собственно, определение таково: подмножество $E \subseteq D$ есть базис, если оно удовлетворяет следующим двум условиям:

(E_1) *если* $e, e' \in E$, *то* $e \sqcup e' \in E$, *и*

(E_2) *для всех* $x \in D$ *мы имеем* $x = \bigsqcup \{e \in E\colon e \preccurlyeq x\}$.

Существование базиса приводит к нескольким следствиям. Например, если существует базис, операция пересечения $x \sqcap y$ непрерывна, в противном случае она не обязана быть непрерывной.

Условия (E_1) и (E_2) все еще недостаточны для того, чтобы сделать типы данных «физическими». Нам нужно более сильное предположение.

Аксиома 5. Тип данных имеет *эффективно заданный* базис.

То есть множество E должно быть «известно». Для данных $e, e' \in E$ мы должны знать, как решить, какое из отношений

$$e \prec e', \quad e \preccurlyeq e', \quad e \sqsubseteq e'$$

истинно, а какое *ложно*. Это однозначно приводит к требованию счетности или конечности для множества E и, возможно, к требованию иметь эффективную нумерацию

$$E = \{e_0, e_1, e_2, \ldots, e_n, \ldots\},$$

в терминах которой указанные выше операции и отношения *рекурсивны*. Чтобы сделать тип данных *действительно* физическим, нам следовало бы считать все это *в высокой степени* вычислимым, но мы здесь не будем входить в подробности: интуитивная идея эффективно заданного базиса не вызовет осо-

бых сомнений, так как в конкретных примерах всегда ясно, что́ нужно уточнить. Заметим, что благодаря аксиоме 5 топологии типов данных *сепарабельны* не только в том смысле, что существуют счетные плотные подмножества (базисы), но и в смысле, что множества

$$\{x \in D: \quad e \prec x\}$$

для всех $e \in E$ образуют счетный базис для топологии в D.

Наиболее важным следствием предположения об эффективно заданном базисе является возможность выяснить, что означает вычислимость *элемента*. Пусть $x \in D$ и E — базис. Тогда x *вычислим* (относительно этого базиса), если существует эффективно заданная последовательность

$$\{e'_0, e'_1, e'_2, \ldots, e'_n, \ldots\} \subseteq E,$$

такая, что $e'_n \sqsubseteq e'_{n+1}$ для каждого n и

$$x = \bigsqcup_{n=0}^{\infty} e'_n.$$

Иными словами, мы должны иметь возможность давать эффективно все более и более хорошие приближения к x, сходящиеся к x в пределе. Это существенное понятие, так как для типа данных D допустимо *несчетное* число элементов, в то время как число вычислимых элементов счетно. Заметим также, что, вообще говоря, может быть много последовательностей, сходящихся к элементу x, так что знание того, что x вычислимо, еще не означает, что известен также и «лучший» способ его вычисления.

На этом мы заканчиваем обсуждение оснований теории. Возможно, кто-нибудь заметит, что аксиома 3 влечет аксиому 1, а аксиома 4 — аксиому 2, но аксиомы были даны в том порядке, в каком возникали естественно. Кратко подытожим все сказанное: типы данных являются полными решетками с эффективно заданными базисами и все допустимые отображения непрерывны. Теперь нужно взглянуть на конструкцию *полезных* типов данных, удовлетворяющих аксиомам, не забывая, что структура решетки есть только наиболее примитивная структура на типе данных и что наиболее «интересная» структура накладывается различного рода непрерывными функциями специально для данного типа.

В первую очередь все *конечные* решетки удовлетворяют аксиомам, и непрерывность для них никакой роли не играет. Разумеется, для «практических» приложений достаточно конечных решеток, но многие понятия легче находят свое выражение через бесконечные структуры. Имеются два особых типа данных N и R для *целых* и *вещественных чисел*. Как решетка, N имеет

элементы 0, 1, 2, ..., n, ... (которые попарно несравнимы относительно $\sqsubseteq$) и еще два элемента $\bot$ и $\top$, расположенные соответственно «выше» и «ниже» всех других элементов. (Здесь была бы полезна картинка.) В этом случае базисом решетки является она сама. Для R элементами служат *замкнутые интервалы* $[\underline{x}, \bar{x}]$ обычных вещественных чисел ($\underline{x} \leqslant \bar{x}$) плюс два элемента $\bot$ и $\top$. Частичный порядок для интервалов определяется так:

$[\underline{x}, \bar{x}] \sqsubseteq [\underline{y}, \bar{y}]$ тогда и только тогда, когда $\underline{x} \leqslant \underline{y} \leqslant \bar{y} \leqslant \bar{x}$.

«Полные» вещественные числа есть одноточечные интервалы $[\underline{x}, \bar{x}]$ с $\underline{x} = \bar{x}$. „Приближенное“ число имеет $\underline{x} < \bar{x}$. Базис состоит из интервалов с *рациональными* концами плюс элемент $\bot$. Есть очень тесная связь между непрерывными функциями на решетке R и обычной теорией непрерывных точечных функций. В обеих решетках арифметические операции представляют собой непрерывные функции. Особенно интересно рассмотреть деление в R.

Пусть D и D' — два каких-либо типа данных. Имеются три особенно важные конструкции

$$(D \times D'), \quad (D + D'), \quad (D \to D')$$

для получения новых, «структурированных» типов данных из начальных типов. (Декартово) *произведение* $D \times D'$ имеет в качестве элементов *пары* (x, x'), где $x \in D$ и $x' \in D'$, и мы определяем:

$(x, x') \sqsubseteq (y, y')$ *тогда и только тогда, когда* $x \sqsubseteq y$ *и* $x' \sqsubseteq' y'$.

Сумма $D + D'$ определяется как «раздельное» объединение D и D', исключая отождествления $\bot = \bot'$ и $\top = \top'$. (Опять был бы полезен рисунок.) *Пространство функций* $(D \to D')$ имеет в качестве элементов все непрерывные функции из D в D', для которых мы определяем:

$f \sqsubseteq g$ *тогда и только тогда, когда* $f(x) \sqsubseteq' g(x)$ *для всех* $x \in D$.

Можно проверить, что эффективно заданные базисы для D и D' определяют базисы в этих конструкциях. Это легко сделать для произведения и суммы и несколько труднее для пространства функций.

Суммы и произведения могут очевидным образом быть обобщены на большее количество слагаемых и сомножителей и даже на бесконечное их число. Например, D^n можно рассматривать как множество всех n-ок из элементов множества D, частично упорядоченное очевидным координатным образом.

Тогда положим

$$D^* = D^0 + D^1 + D^2 + \ldots + D^n + \ldots,$$

что представляет тип всех конечных *списков* из элементов типа D. Можно также прийти к спискам, составленным из списков, которые составлены из списков, которые ... . Если правильно это сделать, то кажется разумным, что получится решетка D^∞, такая, что

$$D^\infty = D + (D^\infty)^*,$$

т. е. каждый элемент решетки D^∞ есть *либо* элемент данного D, либо список из других элементов из D^∞. Это очень напоминает обычное определение списков. Однако необходимо принять во внимание следующее рассуждение.

Хорошо известна теорема о том, что всякая непрерывная (даже только монотонная) функция, отображающая полную решетку *в себя*, имеет *неподвижную точку*. Применяя это замечание к D^∞, мы видим, что для данного $a \in D$ выражение (a, x) определяет непрерывную функцию, отображающую D^∞ в себя. Рассмотрим неподвижную точку

$$x = (a, x).$$

Таким образом, x есть список, первый элемент которого есть a, а второй элемент есть ...? Вторым элементом является сам список x! Значит, x имеет вид бесконечного списка:

$$x = (a, (a, (a, \ldots))).$$

Это не вписывается в обычные идеи о данных типа списков, но разве это плохо? *Нет*, ибо можно показать, что тип D^∞ действительно существует. Он содержит все обычные конечные списки, а также много очень интересных и полезных *пределов* последовательностей из конечных списков. Можно сказать, что D^∞ есть топологическое *пополнение* пространства конечных списков, и мы можем вообще не использовать предельных точек, если нам не видно их какое-либо преимущество.

Процесс «пополнения» пространств весьма общ, и полное приложение этого метода нам пока не ясно. Второй пример касается *пространств функций*. Для данного D положим $D_0 = D$ и

$$D_{n+1} = (D_n \to D_n).$$

Пространства D_n суть «высшие типы» пространств функций над функциями над функциями над ... . Оказывается, существует естественный способ изоморфного вложения каждого D_n в следующее за ним пространство D_{n+1}. Используя эти вложе-

ния, можно перейти к *предельному пространству* D_∞, которое содержит начальный тип данных D и для которого

$$D_\infty = (D_\infty \to D_\infty).$$

(Строго говоря, к этому удивительному равенству нужно отнестись осторожно: оно верно только с точностью до изоморфизма, но для наших целей этого достаточно.) Это пространство дает решение проблемы самоприменимости, так как каждый элемент D_∞ можно рассматривать как (непрерывную!) функцию из D_∞ в D_∞. И обратно, *всякая* непрерывная функция корректно представляется некоторым элементом D_∞. Мы имеем, таким образом, первую «математически» определенную модель так называемого λ-*исчисления* Карри — Чёрча.

Читателю следует заметить, что наша абстрактно представленная теория вычислимых элементов в решетках с эффективно заданными базисами применяется к этим пространствам *функций*. Так что мы знаем, какие функции вычислимы. Более того, мы «знаем», что такое вычислимые элементы пространства D_∞ функций «бесконечного» типа. Ясно, что становится интересным исчисление операторов, порождающих вычислимые функции, и это возвращает нас к семантике языков программирования. В самом деле, естественный путь определения вычислимых функций лежит в контексте подходящего языка программирования. Язык программирования представляет собой и λ-исчисление: это чистый язык «неограниченных» процедур. И это один из многих возможных языков программирования.

В заключение дадим набросок решения проблемы запоминания команд, упомянутой в начале статьи. Пусть L — пространство *мест* (конечное или, если угодно, $L = N$, так что места обозначены целыми числами). Пространство V *значений* должно быть построено методами, которые мы наметили выше. Предположим, что V уже построено. Тогда пространство *состояний памяти* Σ определяется как

$$\Sigma = (L \to V).$$

Пространство *команд* Γ определяется как

$$\Gamma = (\Sigma \to \Sigma).$$

Пространство P *процедур* — с одним параметром и с *побочными эффектами* — определяется как

$$P = (V \to (\Sigma \to V \times \Sigma)),$$

т. е. процедура есть функция, которая при заданном значении ее аргумента и заданном состоянии памяти производит «вычисленное» значение вместе с необходимым изменением состояния памяти. Каковы должны быть эти значения? Они могли бы быть

числами (из N или R), или *местами*, или *списками*, могли бы быть *командами* или *процедурами*. (Даже для обеспечения этого набора возможных значений нам понадобился достаточно углубленный язык программирования.) Мы должны, следовательно, написать

$$V = N + R + L + V^* + \Gamma + P.$$

Если теперь подставить определения Σ, Γ и P в это равенство, то получим лишь несколько более сложное «определение», чем те, которые получились для D^∞ и D_∞. Такое пространство действительно существует математически, и оно снабжает нас значениями для выражений языка программирования того типа, который раньше мы понимали «операционально». Нам следует начать понимать эти языки математически, ибо имеются все необходимые инструменты для этого.

Идею использования монотонных функций в теории рекурсии автор заимствовал у Ричарда Платека (докторская диссертация, Станфорд, 1965). Понятие непрерывности появлялось в работах Лакомба, Нероуда, Клини и Крайзеля и было использовано Платеком, но теория не была достаточно далеко продвинута. Мысль о том, что это можно было бы связать с языками программирования, возникла у автора во время совместной работы с де Баккером по схемам программ в Амстердаме весной и летом 1969 года. Идея абстрактных решеток с базисами является новой и значительно обобщает специальные структуры, которыми пользовались вышеуказанные авторы. Замысел нахождения математической семантики для языков программирования выдвигался Кристофером Стрейчи, и именно благодаря его настойчивости и содействию за осенний семестр 1969 г. в Оксфорде автор смог выработать связную теорию из разрозненных идей. Технические статьи по λ-исчислению и решеткам с базисами будут скоро опубликованы с совместной работой со Стрейчи по семантике различных языков.

СПИСОК ЛИТЕРАТУРЫ [1])

1. Scott D., Continuous lattices, Toposes, algebraic geometry, and logic, Lecture notes in math., **274** (1972).
2. Scott D., The lattice of flow diagrams. Symp. on semantics of algorithmic languages, Lecture notes in math., **181** (1971).

[1]) Добавлен переводчиком. Указаны почти все работы, развивающие идеи переведенной статьи Скотта и опубликованные в доступных изданиях. (Много работ появилось в форме технических и научных отчетов.) Переводчик благодарен В. Ю. Сазонову, помогавшему в составлении этого списка литературы.

3. Scott D., Lattice-theoretical models for various type-free calculi, Proc. IV Intern. Congress on logic, methodology and phylosophy sci., Bucurest (1972).
4. Scott D., Strachey C., Towards a mathematical semantics for computer languages, Proc. symp. on computers and automata, Polytecnical Institute of Brooklyn, N. Y. (1971).
5. De Bakker J. W., de Roever W. P., A calculus for recursive program schemata, Automata, languages and programming, Proc. Symp., IRIA (1973).
6. Manna Z., Vuillemin J., Fixpoint approach to the theory of computation, Communs. ACM conf., **15**, N 7 (1972).
7. Milner R., Implementation and application of Scott's logic for computable functions, Proc. ACM conf., N. Y. (1972).
8. Milner R., Weyhrauch R., Proving compiler correctness in a mathematical logic, *Machine Intelligence*, **7** (1972).
9. Ершов Ю. Л., Вычислимые функционалы конечных типов, *Алгебра и логика*, **11**, № 4 (1972).
10. Ершов Ю. Л., Теория A-пространств, *Алгебра и логика*, **12**, № 4 (1973).
11. Сазонов В. Ю., Последовательно и параллельно вычислимые функционалы, *Сиб. мат. журнал*, **17**, № 3 (1976).
12. Трахтенброт Б. А., Рекурсивные схемы и вычислимые функционалы, Math. foundation of computer science, Lecture notes in computer science (1976).
13. Трахтенброт М. Б., Об интерпретированных функциях в схемах программ, Сб. «Системное и теоретическое программирование», Изд-во ВЦ СО АН, Новосибирск (1973).
14. Транхтенброт М. Б., О представлении последовательных и параллельных функций, Сб. «Вычислительная математика и программирование», Изд-во ВЦ СО АН, Новосибирск (1974).

О классах схем программ[1])

Роберт Л. Констейбл, Дейвид Грис

В статье определяются следующие классы схем программ:

$\mathbf{P}$ — класс схем с конечным числом простых переменных;
$\mathbf{P}_A$ — класс схем с простыми и индексированными переменными (массивами);
$\mathbf{P}_{Ae}$ — класс схем из $\mathbf{P}_A$, допускающих применение предиката равенства к индексированным переменным;
$\mathbf{P}_R$ — класс схем, допускающих рекурсивные определения;
$\mathbf{P}_L$ — класс схем с переменными-метками;
$\mathbf{P}_m$ — класс схем с переменными-маркерами;
$\mathbf{P}_{fds}$ — класс схем с магазинами.

Кроме того, можно рассматривать, например, $\mathbf{P}_{Am}$ — класс схем с массивами и переменными-маркерами и $\mathbf{P}_{AL}$ — класс схем с массивами и переменными-метками.

Утверждается, что $\mathbf{P}_A$, $\mathbf{P}_R$ и $\mathbf{P}_L$ точно представляют механизмы индексирования, рекурсии и переходов по переменным-меткам, имеющиеся во многих «реальных» языках программирования.

В статье показано, что

$$\mathbf{P} \subset \mathbf{P}_R \subset \mathbf{P}_A \equiv \mathbf{P}_{AL} \equiv \mathbf{P}_{Am} \equiv \mathbf{P}_{pdsm} \equiv \mathbf{P}_{Ae}.$$

При этом включения $\mathbf{P} \subset \mathbf{P}_R \subset \mathbf{P}_A$ и эквивалентности $\mathbf{P}_{AL} \equiv \mathbf{P}_{Am} \equiv \mathbf{P}_{pdsm} \equiv \mathbf{P}_{Ae}$ эффективны. Например, если дана схема программ из $\mathbf{P}_{Am}$, то по ней можно эффективно построить эквивалентную схему из $\mathbf{P}_{AL}$. Показано, однако, что хотя для любой схемы из $\mathbf{P}_{AL}$ существует эквивалентная ей схема из $\mathbf{P}_A$, но ее, вообще говоря, нельзя построить эффективно!

Высказывается также предположение, что $\mathbf{P}_A$, $\mathbf{P}_{AL}$ и эквивалентные им классы «универсальны».

1. ВВЕДЕНИЕ

Основной результат логики и теории вычислений заключается в том, что различные на первый взгляд вычислительные системы (например, машины Тьюринга, рекурсивные опреде-

---

[1]) Robert L. Constable, David Gries, On classes of program schemata, TR 71—105, Cornell University, August 1971.

ления) эквивалентны в своих возможностях вычислять функции. Все они универсальны, поскольку допускают вычисление в точности множества эффективных функций.

Патерсон и Хьюитт [5] и Стронг [7] обнаружили, что структурные различия между этими системами выявляются на уровне схем, т. е. при вычислении функционалов.

В [5] демонстрируется следующая иерархия схем: класс схем программ ($\mathbf{P}$) содержится в классе рекурсивных схем ($\mathbf{P}_R$), который содержится в классе параллельных рекурсивных схем ($\mathbf{P}_p$). Эти результаты показывают, что механизм рекурсивного вычисления функций «мощнее» механизма вычисления машинами с конечным числом ячеек. Там же поднят вопрос о том, существует ли хорошо определяемое понятие *универсальной схемы вычисления.*

Мы хотим точнее установить, как соотносятся между собой различные классы схем, а также предложить *естественную* модель схемы вычислений, которая, быть может, окажется универсальной.

Нахождение универсальных схем вычислений, если они существуют, должно происходить аналогично нахождению универсальных вычислительных систем. А именно, предлагается несколько кандидатов и устанавливается их эквивалентность (ведь для того, чтобы начать исследование, требуются кандидаты). В качестве одного из таких кандидатов мы предлагаем схемы с массивами и маркерами. В [5] Патерсон предлагает схемы с магазинами, а Стронг [7] — свои эффективные функционалы. Мы устанавливаем соотношение между нашими схемами с массивами и схемами с магазинной памятью.

Наши «доказательства» эквивалентности или включения одного класса схем в другой заключаются обычно в том, что показывается, как по данной схеме S из одного класса построить эквивалентную ей схему S' из другого класса. Конструкции, как правило, очень просты и очевидны. Формальных доказательств эквивалентности мы не даем.

Материал в статье расположен следующим образом. В разд. 2 рассматривается класс $\mathbf{P}$ схем программ и определяется, что мы понимаем под «эквивалентностью» двух схем. В разд. 3 вводятся рекурсивные схемы $\mathbf{P}_R$, и они сравниваются с обычнами рекурсивными функциями, используемыми в математике. Описывается макрорасширение, которое заключается в подстановке вместо вызова функции соответствующего ей тела, что дает эквивалентную схему. В разд. 4 определяется несколько классов схем: $\mathbf{P}_A$, $\mathbf{P}_{Ae}$, $\mathbf{P}_m$, $\mathbf{P}_L$ и $\mathbf{P}_{pds}$. В разд. 5 показывается, что $\mathbf{P}_R \subset \mathbf{P}_A$. В разд. 6 рассматриваются схемы классов $\mathbf{P}_{pds}$ и $\mathbf{P}_{pdsm}$; показывается, что в любой схеме достаточно иметь максимум два маркера и два магазина. В разд. 7 и 8 показывается,

что $\mathbf{P}_{Am}$, $\mathbf{P}_{AL}$, $\mathbf{P}_{pdsm}$ и $\mathbf{P}_{Ae}$ все эквивалентны и, наконец, что $\mathbf{P}_A \equiv \mathbf{P}_{Am}$. Раздел 9 посвящен изучению схем, допускающих многомерные массивы.

2. ОСНОВНЫЕ ПОНЯТИЯ СХЕМ, КОНЦЕПЦИЯ ЭКВИВАЛЕНТНОСТИ

(2.1) Определение. Схема из множества ***P*** *схем программ* — это предложение в грамматике G[⟨программа⟩], определяемой ниже; при этом

v, w обозначают *простые переменные* из множества $V = \{V_1, V_2, \ldots\}$;
h, f обозначают операции, или *базисные* (всюду определенные) *функции* из множества $F = \{F_1, F_2, \ldots\}$. Каждая функция F_i имеет ранг RF_i (равный числу аргументов у F_i);
p, q обозначают (всюду определенные) *предикаты* из множества $P = \{P_1, P_2, \ldots\}$. Каждый предикат P_i имеет ранг (количество аргументов) RP_i;
l обозначает *метку* из множества $L = \{L_1, L_2, \ldots\}$.

Грамматика G[⟨программа⟩] содержит правила [1])

⟨программа⟩ ::= (v{, v}): ⟨тело⟩
⟨тело⟩ ::= ⟨S-список⟩; [l:] СТОП(v)
⟨S-список⟩ ::= [l:]⟨S⟩ {; [l:]⟨S⟩}
⟨S⟩ ::= „пусто“
| $v \leftarrow m$
| $v \leftarrow f(v_1, \ldots, v_{Rf})$
| **если** $p(v_1, \ldots, v_{Rp})$ **то** [l:]⟨S⟩ **иначе** [l:]⟨S⟩
| СТОП(v)
| **на** l
| **начало** ⟨S-список⟩ **конец**
| **пока** $p(v_1, \ldots, v_{Rp})$ **цикл** ⟨S-список⟩ **конец**

Непосредственно перед ⟨телом⟩ указаны простые переменные, значения которых служат входом для ⟨программы⟩; они должны быть попарно различны. Кроме того, любая метка l, использованная в операторе **на** l, должна также метить некоторый оператор, например l: ⟨S⟩. Каждой меткой l можно метить только один оператор.

---

[1]) Мы пользуемся БНФ со следующими дополнительными обозначениями: {x} означает 0, 1, 2 или более вхождений x, [x] означает 0 или 1 вхождение x.

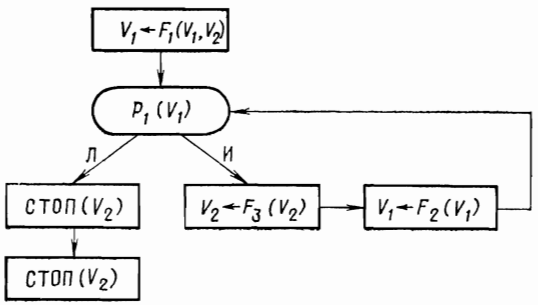
(2.2) Пример. Конструкция

(V_1, V_2): $V_1 \leftarrow F_1(V_1, V_2)$;
L_1: **если** $P_1(V_1)$
то начало $V_2 \leftarrow F_3(V_2)$; $V_1 \leftarrow F_2(V_1)$;
на L_1 **конец**
иначе СТОП(V_2);
СТОП(V_2)

является схемой программ.

В оставшейся части раздела излагается наше представление о том, что такое схема программ. Мы описываем, кроме того, несколько других ограничений, налагаемых на схемы и их семантику.

Патерсон [4] ввел схемы в виде диаграммы. Мы решили определить их формально в терминах алголоподобных операторов исключительно для того, чтобы уменьшить число диаграмм в статье. Иногда, впрочем, мы для ясности будем пользоваться диаграммами. Например, схеме программ, приведенной в примере, эквивалентна диаграмма



Схемы программ могут выражать алгоритмы для вычислений над любой областью D (например, $D = \mathbb{N} = \{0, 1, 2, \ldots\}$ или $D =$ {множество всех двоичных деревьев}). Операции над D — это отображения $\mathscr{F}(\ )$: $D^{Rf} \to D$, а предикаты — это $\mathscr{P}(\ )$: $D^{Rp} \to$ {И, Л}. Таким образом, параметры, необходимые для спецификации конкретного языка программирования, — это область D, подмножество операций $\mathscr{F}(D) = \{f(\ )\colon D^{Rf} \to D\}$ и подмножество предикатов $\mathscr{P}(D) = \{p(\ )\colon D^{Rp} \to$ {И, Л}$\}$.

Предположим, что задана интерпретация области D, предикатов P_i и функций F_i, которые могут использоваться в схемах. Тем самым получен язык программирования, а каждая схема программ становится программой в этом языке.

Например, если положить $D = \mathbb{N}$, $f_1(x) = x + 1$ и $f_2(x) = x - 1$, то получим язык G_3, универсальный для всех эффективных функций над $\mathbb{N}$. Другими словами, в нем выразимы алгоритмы для всех вычислимых функций над $\mathbb{N}$.

Чтобы выполнить программу, нужно переменным $v\{\ , v\}$, перечисленным перед началом ⟨программы⟩, приписать n_3 входных значений, а затем выполнить ⟨тело⟩. Способ выполнения операторов любого из указанных типов ясен каждому программисту, и мы больше не будем об этом говорить. Часто мы пользуемся фразой «выполнить схему» как сокращением для «выполнить схему при некоторой интерпретации множеств D, $\mathscr{F}$, $\mathscr{P}$ и при входных значениях...».

Итак, каждую конкретную схему программ можно рассматривать как отображение n_1 функций, n_2 предикатов и n_3 входных значений в D:

$$\mathscr{F}(D)^{n_1} \times \mathscr{P}(D)^{n_2} \times D^{n_3} \to D$$

Такие отображения называются *функционалами* над D. Например, схема программ (2.2) вычисляет функционал

$$\mathscr{F}(D)^3 \times \mathscr{P}(D)^1 \times D^2 \to D$$

(2.3) Определение. $\mathbb{F}(\mathbf{P}, D)$ — это множество всех функционалов над D, вычислимых схемами программ из класса **P**.

Абстрагирование от программы к схеме программ может привести к лучшему пониманию программистских средств, поскольку оно позволяет полностью отделить вычисляемые значения от структуры данных и механизма управления, применяемых в программе. Например, мы определяем схемы, использующие массивы индексированных переменных; в этих схемах механизм индексирования полностью отделен от вычисляемых значений. А именно, ни один индекс нельзя взять в качестве аргумента базисной функции и ни одно базисное значение нельзя взять в качестве индекса.

Это позволяет сравнить класс $\mathbf{P}_A$ схем с массивами и класс $\mathbf{P}_R$ рекурсивных схем; оказывается, класс схем $\mathbf{P}_A$ мощнее!

Отметим, что оператор

пока $p(\ldots)$ **цикл** ⟨S-список⟩ **конец**

равносилен оператору

l: **если** $p(\ldots)$
то начало ⟨S-список⟩; **на** l **конец**
иначе

Будем предполагать, что все операторы **пока** переведены в эту форму.

Заметим, что все наши схемы имеют единственное выходное значение — значение переменной v из оператора СТОП(v), завершающего выполнение схемы. Это искусственное ограничение; мы ввели его только для краткости и ясности изложения.

Предполагается, что все функции и предикаты *всюду определены.* Это значит, что они дают определенное значение из D, коль скоро значения аргументов принадлежат D. Другие исследователи, работающие в этой области, допускают частичные функции и предикаты.

Если допускать частичные функции и предикаты, часть наших доказательств можно упростить, для других же доказательств это не так. Кроме того, если допускаются частичные базисные функции или предикаты, то никакую форму *параллелизма* нельзя моделировать без *специальных средств параллелизма.*

Как бы то ни было, нам кажется, что свойство *всюду определенности* полезнее и желательнее. В любой «реальной» системе программирования базисные функции всюду определены. В качестве результата они могут выдавать неопределенное значение, но это лучше, чем бесконечное зацикливание из-за неправильных входных данных.

Если далее мы намереваемся охарактеризовать поведение схем, в которых аргументами функций служат подпрограммы (возможно, тоже частичные), то реально это можно сделать, допустив, что аргументами схем могут быть другие схемы. Это непосредственно приводит к теории схем программ произвольного конечного типа. Возможно, в следующих сообщениях мы подробнее обсудим схемы этих типов.

Цель данной статьи — сравнить несколько различных классов схем и выяснить, эквивалентны ли они. При этом наша концепция эквивалентности включает и такие схемы, которые при некоторых интерпретациях выполняются «плохо». Например, схема

$$(V): \quad \text{СТОП}(W)$$

всегда выполняется неправильно, поскольку выходной переменной W не присваивается никакого значения! Тем не менее и такие схемы нужно принимать в расчет.

Мы предполагаем, что каждая область D содержит неопределенное значение Ω. В любой схеме каждая переменная имеет перед началом выполнения (при некоторой интерпретации) значение Ω. Можно также предположить, что есть простая переменная ОМЕГА, которой в ходе выполнения не присваивается никакого значения; следовательно, она всегда имеет значение Ω.

Это предположение поможет нам еще в одном случае. В дальнейшем мы определим классы схем с метками, массивами и т. д. Для того чтобы разделить, насколько это воз-

можно, различные понятия, мы хотим запретить меткам и маркерам быть элементами области D. Например, если в одном из классов схем допускаются операторы вида $v \leftarrow f(m, w)$, где m — специальный маркер, а f — базисная функция, то ясно, что в классе, не допускающем использование маркеров, не может быть эквивалентной схемы. Поэтому мы принимаем следующее соглашение:

(2.4) Если в качестве аргумента базисного предиката, функции или оператора СТОП нужно взять значение, не принадлежащее D, то вместо этого аргумента подставляется неопределенное значение Ω.

Опять-таки такое соглашение для теории несущественно (можно было бы поступать, как в «реальных» языках программирования), но оно приводит к более ясным результатам.

(2.5) Определение. Две схемы программ S_1 и S_2 с входными данными $x_1, \ldots, x_{n_1}$, функциями $f_1, \ldots, f_{n_2}$, предикатами $p_1, \ldots, p_{n_3}$ и выходом y *эквивалентны над* D тогда и только тогда, когда они вычисляют один и тот же функционал над D, т. е.

$$S_1[x_1, \ldots, x_{n_1}, f_1, \ldots, f_{n_2}, p_1, \ldots, p_{n_3}] = \\ = S_2[x_1, \ldots, x_{n_1}, f_1, \ldots, f_{n_2}, p_1, \ldots, p_{n_3}]$$

для всех $x_1, \ldots, x_{n_1}$ из D. Две схемы *эквивалентны*, если они эквивалентны над любой областью D.

3. РЕКУРСИВНЫЕ СХЕМЫ И МАКРОРАСШИРЕНИЯ

Рекурсивное определение дает один из основных способов задания функции в математике. Основную структуру рекурсивного определения можно специфицировать на схемном уровне посредством понятия рекурсивной схемы.

(3.1) Определение. (*Рекурсивное*) *определение функции* f ранга Rf имеет вид $f(x_1, x_2, \ldots, x_{Rf}) = e_0$, где e_0 — выражение. Выражение e (или e_i) имеет один из трех типов:

1) x_i	(i-й параметр; $1 \leqslant i \leqslant n$),
2) $h(e_1, e_2, \ldots, e_{Rh})$,	где h — (всюду определенная) функция,
3) **если** $p(e_1, e_2, \ldots, e_{Rp})$ **то** e_i **иначе** e_j,	где p — (всюду определенный) предикат.

(3.2) Определение. *Рекурсивная схема* состоит из конечного числа определений функций $f_1, f_2, \ldots, f_n$, а также начального

вызова $f_i(v_1, \ldots, v_{Rf_i})$ одной из функций f_i, аргументами которой служат значения $v_1, \ldots, v_{Rf_i}$.

Класс рекурсивных схем обозначим через **R**. Отметим, что рекурсивная схема реализует отображение

$$\mathscr{F}(D)^{n_1} \times \mathscr{P}(D)^{n_2} \times D^{n_3} \to D$$

при некоторых n_1, n_2, n_3. Таким образом, и рекурсивные схемы, и схемы программ осуществляют вычисления над одним и тем же множеством областей. Естественно поэтому спросить, вычисляют ли они один и тот же класс функционалов над всеми областями D. Точнее: верно ли, что $\mathbb{F}(\mathbf{R}, D) = \mathbb{F}(\mathbf{P}, D)$ для всех D? Патерсон и Хьюитт [5] показали, что справедлива

(3.3) Теорема. *Существует такая область* D, *что* $\mathbb{F}(\mathbf{P}, D) \subset \subset \mathbb{F}(\mathbf{R}, D)$.

Фактически, доказав этот результат для случая свободной области $D = \{F_i, P_i \mid i \in \mathbb{N}\}^*$, они установили его справедливость и для любой бесконечной области D. Из этой теоремы следует, что механизм рекурсивного вычисления «мощнее» механизма вычисления программами, содержащими только простые переменные. Другими словами, языки программирования, использующие конечное число ячеек, но допускающие рекурсию, мощнее языков без рекурсии.

Так как $\mathbb{F}(\mathbf{P}, D)$ — подмножество класса $\mathbb{F}(\mathbf{R}, D)$, удобно ввести понятие класса рекурсивных схем программ $\mathbf{P}_R$ так, чтобы $\mathbb{F}(\mathbf{R}, D) \equiv \mathbb{F}(\mathbf{P}_R, D)$.

(3.4) Определение. *Рекурсивная схема программ* из класса $\mathbf{P}_R$ — это предложение в грамматике G[⟨рекурсивная программа⟩], приведенной ниже (в нее входят описанные в (2.1) правила для ⟨программы⟩ и т. д.).

⟨рекурсивная программа⟩ : : = ⟨программа⟩ {⟨определение функции⟩}
⟨определение функции⟩ : : = $f(v_1, \ldots, v_{Rf})$: ⟨тело⟩

Таким образом, рекурсивная схема программ состоит из обычной схемы программ и последовательности определений функций. Определяемые функции называются рекурсивно определенными в противовес базисным функциям, которые не определяются. «Выполнение» осуществляется, как и раньше, за исключением того, что когда вызывается рекурсивно определенная функция, происходит следующее:

1) Значения аргументов вызова присваиваются соответствующим переменным (формальным параметрам) $v_1, \ldots$

..., v_{Rf}, а остальные переменные, используемые функцией, полагаются равными Ω.

2) ⟨тело⟩ функции исполняется до тех пор, пока не будет выполнен оператор СТОП(v). Тогда значение функции считается равным значению переменной v, а выполнение схемы возобновляется в точке вызова.

Предполагается, что переменные и метки, участвующие в некотором определении функции, отличны от тех, которые участвуют в ⟨программе⟩ и других определениях функций. Предполагается также, что они различны в разных обращениях к функции.

(3.5) Пример.

```
(V, W) :    V ← F1(V, W); W ← F2(V, W); СТОП(V).
F2(X, Y):   L:  если P(X) то
                  начало X ← F3(X); Y ← F2(X, Y);
                         на L
                  конец
                иначе СТОП(Y);
                СТОП(Y)
```

Впервые эквивалентность классов $\mathbf{P}_R$ и $\mathbf{R}$ доказал Маккарти [3]. Покажем, как транслировать рекурсивную схему в рекурсивную схему программ (которая интуитивно имеет эквивалентный смысл).

(3.6) Преобразование $S \in \mathbf{R}$ в $S' \in \mathbf{P}_R$. Начальный вызов $f(v_1, \ldots, v_{Rf})$ преобразуется в ⟨программу⟩

$(V_1, \ldots, V_{Rf})$: ВЫВОД $\leftarrow f(V_1, \ldots, V_{Rf})$; СТОП(ВЫВОД)

Определение функции f_i транслируется в

$$f_i(V_1, V_2, \ldots, V_{Rf_i}): V \leftarrow e_0; \text{ СТОП}(V)$$

Внутри выражения e_0 все x_j заменены на V_j. Далее для каждого определения функции итерируются следующие два шага (до тех пор, пока каждый из них не станет неприменимым):

а) если есть присваивание $v \leftarrow f(e_1, \ldots, e_{Rf})$, где по крайней мере одно из выражений e_i не является переменной, то порождаются новые переменные $V_1, \ldots, V_{Rf}$, а само присваивание заменяется на

```
начало V1 ← e1; ...; VRf ← eRf;
       v ← f(V1, ..., VRf)
конец
```

б) если есть присваивание

$$v \leftarrow \textbf{если } p(e_1, \ldots, e_{Rp}) \textbf{ то } e_i \textbf{ иначе } e_j$$

то порождаются новые переменные $V_1, \ldots, V_{Rp}$, а само присваивание заменяется на

начало $V_1 \leftarrow e_1; \ldots; V_{Rp} \leftarrow e_{Rp};$

если $p(V_1, \ldots, V_{Rp})$ **то** $v \leftarrow e_i$ **иначе** $v \leftarrow e_j$

конец

Конкретный вызов функции в «реальной» программе можно рассматривать также как макровызов. Под этим мы понимаем, что перед выполнением программы вызов функции заменяется на ⟨тело⟩ определения этой функции, причем осуществляется такое подходящее переименование переменных и инициализация параметров и результирующих переменных, чтобы получилась эквивалентная программа. Во время ее выполнения вычисление функции осуществляется без перехода на ее ⟨тело⟩. Этот процесс можно провести и для схем.

(3.7) Макрорасширение. Если дана схема из $\mathbf{P}_R$ с вызовом $v \leftarrow f(w_1, \ldots, w_{Rf})$, где f определяется некоторым ⟨определением функции⟩, то мы *раскрываем* вызов и получаем новую, эквивалентную схему из $\mathbf{P}_R$. Это делается так:

а) Копируется ⟨тело⟩ ⟨определения функции⟩ f. Пусть в теле используются переменные $V_1, \ldots, V_n$ и метки $L_1, \ldots, L_m$. Тогда создаются новые переменные и метки $V_1^*, \ldots, V_n^*, L_1^*, \ldots, L_m^*$ и внутри копии ⟨тела⟩ все V_i (L_j) заменяются на V_i^* (L_j^*).

б) Создается новая метка L. Каждый оператор СТОП (V_i^*) внутри копии ⟨тела⟩ заменяется на

$$\textbf{начало } v \leftarrow V_i^*; \textbf{ на } L \textbf{ конец}$$

(напомним, что первоначально был вызов $v \leftarrow f(w_1, \ldots, w_{Rf})$).

в) Предположим, что формальными параметрами функции f служат $V_1, V_2, \ldots, V_{Rf}$. Тогда вызов $v \leftarrow f(w_1, \ldots, w_{Rf})$ заменяется на

начало $V_1^* \leftarrow w_1; \ldots; V_{Rf}^* \leftarrow w_{Rf};$
(копия ⟨тела⟩, полученная в пункте (б));
L:

конец

(3.8) Пример. Рассмотрим схему S:

$$\begin{array}{rl} (W): & V \leftarrow W; \\ & V \leftarrow F(V, W); \\ & \text{СТОП}(V) \\ F(X, Y): & X \leftarrow Y; \ \text{СТОП}(X \end{array}$$

Раскрытие вызова $V \leftarrow F(V, W)$ дает

$$\begin{array}{rl} (W): & V \leftarrow W; \\ & \textbf{начало } X^* \leftarrow V; \ Y^* \leftarrow W; \\ & \qquad X^* \leftarrow Y^*; \ \textbf{начало } V \leftarrow X^*; \ \textbf{на } L \ \textbf{конец} \\ L: & \textbf{конец}; \\ & \text{СТОП}(V); \\ F(X, Y): & X \leftarrow Y; \ \text{СТОП}(X) \end{array}$$

В дальнейшем при доказательстве включения $\mathbb{F}(\mathbf{P}_R, D) \subset \subset \mathbb{F}(\mathbf{P}_A, D)$ мы должны будем суметь раскрыть возможные вызовы рекурсивно определенных функций, имеющие первый уровень. Вызов $f(\ldots)$ имеет первый уровень, если во время выполнения схемы (при некоторой интерпретации) он *не* вызывается выполнением другого вызова f.

(3.9) Раскрытие всевозможных вызовов первого уровня. Опишем теперь, как по схеме S построить схему S', в которой все возможные вызовы первого уровня раскрыты.

а) Внутри главной ⟨программы⟩ все присваивания $v \leftarrow \leftarrow f(\ldots)$ (где функция f не базисная) переписываются в виде

$$v \leftarrow f(\ldots)\{\ \}$$

Аналогично внутри определения каждой функции g все присваивания $v \leftarrow f(\ldots)$ (где функция f не базисная) переписываются в виде

$$v \leftarrow f(\ldots)\{g\}$$

{ } означает пустой список и указывает, что, какова бы ни была определяемая функция, данный вызов не вызывается ни при каком ее выполнении. $\{g\}$ означает, что данный вызов заведомо возникнет при выполнении функции g. Продолжая построение, получаем в общем случае, что в скобках стоит список тех функций, при выполнении которых возникает данный вызов.

б) Сначала для главной схемы, а затем для каждого определения функции итерируется, пока это возможно, следующий шаг.

Выбирается такое присваивание

$$v \leftarrow f(w_1, \ldots, w_{Rf})\{f_j, \ldots, f_k\}$$

что f не содержится в списке $\{f_j, \ldots, f_k\}$. Это означает, что данный вызов не возникает при другом выполнении функции f, а потому он вызов первого уровня. Этот вызов раскрывается,

как описано в (3.7). К каждому списку, поставленному в соответствие стоящему в раскрытом теле функции f вызову небазисной функции, добавляется список $\{f, f_j, \ldots, f_k\}$.

в) Все списки $\{\ldots\}$ удаляются из вызовов.

Очевидно, что если описанный процесс заканчивается, то в результате получается эквивалентная схема программ. Кроме того, будут раскрыты все вызовы первого уровня, поскольку, пока есть хотя бы один такой вызов, можно применять шаг (б), который и раскроет его.

Отметим, что за счет каждого макрорасширения исчезает один вызов и добавляется несколько новых (стоявших в теле функции, вызываемой раскрываемым вызовом). При этом число элементов списка, приписанного каждому из новых вызовов, по крайней мере на единицу больше соответствующего числа для раскрываемого вызова. Если учесть еще тот факт, что число элементов списка, поставленного в соответствие вызову первого уровня, меньше числа ⟨определений функций⟩, то мы увидим, что шаг (б) в конце концов заканчивается.

(3.10) Следствие. *Пусть каждый вызов некоторой схемы из* $\mathbf{P}_R$ *нерекурсивен при любой интерпретации и любом выполнении. Тогда в* $\mathbf{P}$ *существует эквивалентная ей схема.*

Доказательство. Раскроем все вызовы первого уровня. Если после этого в главной ⟨программе⟩ остались какие-нибудь вызовы, они заведомо никогда не будут выполнены (это не вызовы первого уровня, а значит, они рекурсивны). Каждый из таких вызовов заменяется пустым оператором. После этого можно убрать все ⟨определения функций⟩, что дает схему в $\mathbf{P}$.

В дальнейшем при сравнении $\mathbf{P}_R$ с другими классами схем мы будем пользоваться интересной взаимосвязью между рекурсивными вызовами и предикатами, используемыми в схеме.

(3.11) Теорема. *Пусть при выполнении схемы из* $\mathbf{P}_R$ *функция* f *вызывается рекурсивно (т. е. она вызывается второй, третий, ..., n-й раз, прежде чем завершено выполнение первого вызова). Тогда, прежде чем последнее (n-е) выполнение функции* f *приведет к выполнению оператора* СТОП(v), *некоторый предикат* P_i *должен быть вычислен на двух таких аргументах* a_1 *и* a_2, *что* $P_i(a_1) \neq P_i(a_2)$.

Доказательство. Пометим все операторы схемы различными метками. По ходу выполнения схемы будем выписывать последовательность меток операторов в том порядке, в каком мы *приступаем* к их выполнению. При этом мы начинаем

с первого оператора в первом вызове f, а кончаем n-м вызовом f:

$$L_1, L_2, \ldots, L_m$$

В другой список записываются метки операторов, выполняемых при n-м (последнем) выполнении f (оно заканчивается выполнением оператора СТОП(v)):

$$L'_1, L'_2, \ldots, L'_q$$

Понятно, что $L_1 = L'_1$, поскольку обе последовательности начинаются с метки первого оператора ⟨тела⟩ f. Если $q \leqslant m$, то $L'_q \neq L_q$, так как L'_q метит оператор СТОП(v), а L_q метит заведомо другой оператор. Если $q > m$, то $L_m \neq L'_m$, так как L_m метит вызов функции f, а L'_m не может быть меткой такого вызова. В любом случае получается, что последовательности, имеющие одинаковые начала, в каком-то месте различаются. Пусть i $(1 < i \leqslant q, m)$ — первый номер, для которого $L_i \neq L'_i$, $L_{i-1} = L'_{i-1}$. Это может быть лишь в том случае, когда L_{i-1} метит оператор вида

если $p(\ldots)$ **то** S_1 **иначе** S_2

а вычисления предиката p дают разные значения. Теорема доказана.

(3.12) Следствие. *Пусть в схеме из* $\mathbf{P}_R$ *раскрыты (по* (3.9)*) все вызовы первого уровня. Тогда при любой интерпретации для выполнения полученной схемы справедливо следующее: прежде чем будет выполнен оператор* СТОП(v) ⟨тела⟩ *какой-нибудь небазисной функции, некоторый предикат* P_i *должен быть вычислен на двух таких аргументах* a_1 *и* a_2, *что* $P_i(a_1) \neq P_i(a_2)$.

Доказательство. Если раскрыть все вызовы первого уровня, то в полученной схеме останутся только рекурсивные вызовы исходной схемы, так что можно применить теорему (3.11).

4. ДРУГИЕ КЛАССЫ СХЕМ

Введем теперь другие классы схем программ, а именно класс $\mathbf{P}_A$ схем с массивами индексированных переменных, класс $\mathbf{P}_{A^e}$ схем с массивами и предикатом равенства, применяемым к индексированным переменным, класс $\mathbf{P}_L$ схем, допускающих метки l в качестве значений, которые можно присваивать переменным, класс $\mathbf{P}_m$ схем, допускающих конечное число «маркеров» в качестве значений переменных, и класс $\mathbf{P}_{pds}$ схем, которые могут использовать магазинную память.

$\mathbf{P}_A$ и $\mathbf{P}_L$ интересны тем, что они содержат средства, имеющиеся в современных языках программирования высокого уровня (например, ПЛ/I). Что касается магазинной памяти, то

эта структура данных, пожалуй, одна из излюбленных для теоретиков. Поэтому, рассматривая классы $\mathbf{P}_{pds}$ и $\mathbf{P}_A$, мы хотим сравнить мнения теоретиков и программистов об универсальных схемах.

(4.1) Определение. $\mathbf{P}_A$ определяется как класс схем программ $\mathbf{P}$, допускающих наряду с простыми переменными v индексированные переменные $a[w]$. Входные переменные обязательно являются простыми.

Массив (одномерный) B — это последовательность $B_0, B_1, \ldots$ простых переменных [1]). В качестве индекса можно взять любое значение из $\mathbb{N} = \{0, 1, 2, \ldots\}$. Для того чтобы в ходе выполнения порождать новые индексированные переменные, вводятся новые операторы

$$\langle S\rangle ::= v \leftarrow 0 \mid v \leftarrow w + 1$$

В результате выполнения оператора $v \leftarrow 0$ переменной v присваивается значение 0. Функция $+1$ осуществляет переход к следующему элементу; если v имеет значение $i \in \mathbb{N}$, то w в результате выполнения оператора $w \leftarrow v + 1$ примет значение $i + 1$.

Предполагается, что при любой интерпретации $\mathbb{N}$ не пересекается с областью D (это мы скоро обсудим). Наконец, считается, что если значение переменной v не принадлежит $\mathbb{N}$, то $v + 1$ имеет значение 1.

Если значение v есть $i \in \mathbb{N}$, то $B[v]$ — простая переменная B_i. Если значение v не принадлежит $\mathbb{N}$, то $B[v]$ — это B_0.

Отметим, что поскольку $\mathbb{N}$ и D не пересекаются, то там, где мы хотели бы взять в качестве аргумента $i \in \mathbb{N}$ (в предикате, функции или операторе СТОП), используется значение Ω. Это ограничение необходимо для того, чтобы исключить бессмысленное сравнение схем с массивами и схем без массивов. Оно не случайно, а служит для того, чтобы отделить способ задания памяти от других вычислительных механизмов.

Читатель может утверждать, что мы описываем механизм индексирования не так, как это делается в конкретных языках программирования. Для этого у него есть три причины:

1) значения, которые берутся в качестве индексов, не могут быть входными данными программы, поскольку они не принадлежат D;

---

[1]) В разд. 9 рассматриваются и многомерные массивы. Будет показано, что их применение не приводит к увеличению мощности класса $\mathbf{P}_A$.

2) функции типа $+1$ обычно допускаются как «базисные» операции;

3) значения, которые берутся в качестве индексов, не могут быть ни аргументами, ни результатами базисных функций и предикатов.

На это можно ответить следующим образом. Мы пытаемся изолировать различные концепции языков программирования, чтобы дать их описание настолько «чистым», насколько возможно. Далее, мы хотим сравнить вычислительную мощность различных классов схем программ. Если в одном классе допускаются вычисления с использованием функции $+1$ (при некоторых ограничениях), а в другом нет, то такое сравнение становится бессмысленным. Однако если функция $+1$ используется *только* для организации памяти и *не может* использоваться в собственно вычислении, то сравнение может иметь смысл.

Если бы некоторая интерпретация схемы программ задавала функцию F с теми же свойствами, что и у $+1$, то F и $+1$ можно было бы отождествить в этой интерпретации. Аналогично если бы область D содержала подмножество $\tilde{\mathbb{N}}$, изоморфное $\mathbb{N}$, то $\tilde{\mathbb{N}}$ и $\mathbb{N}$ можно было бы отождествить в этой интерпретации.

Строго говоря, операторы типа

(1) $B[V+3] \leftarrow W$; (2) $W \leftarrow 2$; (3) $B[5] \leftarrow W$

не являются правильными для схем класса $\mathbf{P}_A$. Мы допускаем их, так как они делают схему проще, а также потому, что их легко транслировать в правильные операторы класса $\mathbf{P}_A$. Например, операторы (1) и (2) транслируются в

начало $V \leftarrow V+1$; $V \leftarrow V+1$; $V \leftarrow V+1$; $B[V] \leftarrow W$ **конец**

и

начало $W \leftarrow 0$; $W \leftarrow W+1$; $W \leftarrow W+1$ **конец**

Отметим тем не менее, что, когда говорится «дана некоторая схема S из $\mathbf{P}_A$» (или $\mathbf{P}_{Am}$ и т. д), *всегда имеется в виду, что схема строго содержится в данном классе.*

Аналогично можно доказать, что использование функции $\dot{-}1$ определяемой соотношением

$$\textbf{(4.2)} \quad w \dot{-} 1 = \begin{cases} i-1, & \text{если } w \text{ имеет значение } i \in \mathbb{N},\ i \neq 0, \\ 0, & \text{если } w \text{ имеет значение } 0, \\ 0, & \text{если значение } w \text{ не принадлежит } \mathbb{N}, \end{cases}$$

не увеличивает мощности схем.

(4.3) Теорема. *Пусть S — схема из $\mathbf{P}_A$ с той оговоркой, что в ней используется функция $\dot{-}1$. Тогда S можно транслировать в эквивалентную схему S' из $\mathbf{P}_A$.*

Доказательство. S' моделирует функцию $\dot{-}1$ при помощи нового массива DOWN. Таким образом, DOWN [0] = 0, а DOWN [i], когда это необходимо, содержит значение $i \dot{-} 1$ для $i > 0$. S транслируется в S' так:

(1) в начале схемы S помещается оператор DOWN [0] $\leftarrow 0$;
(2) каждый оператор $v \leftarrow w + 1$ в S заменяется на

начало DOWN [$w + 1$] $\leftarrow w$; $v \leftarrow w + 1$ **конец**

(3) каждый оператор $v \leftarrow w \dot{-} 1$ в S заменяется на

$$v \leftarrow \text{DOWN}[w]$$

Теорема доказана.

(4.4) Определение. Схема класса $\mathbf{P}_{Ae}$ — это схема из $\mathbf{P}_A$, в которой допускаются операторы типа

$$\langle S\rangle ::= \textbf{если } v \ominus w \textbf{ то } [l:]\langle S\rangle \textbf{ иначе } [l:]\langle S\rangle$$

Предикат $v \ominus w$ определяется формулой

$$v \ominus w = \begin{cases} \text{И, если } v \text{ и } w \text{ оба не принадлежат } \mathbb{N} \setminus \{0\}, \\ \text{И, если } v, w \text{ принадлежат } \mathbb{N} \text{ и совпадают,} \\ \text{Л в остальных случаях.} \end{cases}$$

Таким образом, $v \ominus w$ тогда и только тогда, когда $A[v]$ и $A[w]$ — одна и та же простая переменная.

(4.5) Определение. Схемой из класса $\mathbf{P}_L$ схем, допускающих переменные-метки, называется схема класса $\mathbf{P}$ с двумя дополнительными операторами

$$\langle S\rangle ::= v \leftarrow l \mid \textbf{на } v$$

Если в схеме есть оператор $v \leftarrow l$, то метка l обязана метить некоторый оператор. В результате выполнения оператора $v \leftarrow l$ переменной v присваивается метка l. Если v имеет значение l, то переход **на** v передает управление на оператор, помеченный l; если же v имеет любое другое значение, то **на** v равносильно пустому оператору. Предполагается, что при любой интерпретации множество меток L не пересекается с областью D.

(4.6) Определение. Пусть s обозначает *магазин* из класса магазинов $PD = \{PD_1, PD_2, \ldots\}$. $\mathbf{P}_{pds}$ определяется как класс

схем **P**, расширенный за счет добавления двух операторов

$$\langle S\rangle ::= \langle \text{pushdown}\rangle \mid \langle \text{popup}\rangle$$
$$\langle \text{popup}\rangle ::= v \leftarrow s$$
$$\langle \text{pushdown}\rangle ::= s \leftarrow v$$

В результате выполнения оператора ⟨pushdown⟩ содержимое v помещается в *верхушку магазина* s. В результате выполнения оператора ⟨popup⟩ содержимое верхушки помещается в v, после чего верхушкой становится следующий элемент магазина. Если в момент выполнения оператора $v \leftarrow s$ магазин s пуст, то v и s не меняются. Каждый магазин первоначально пуст.

Особо отметим эффект выполнения оператора ⟨popup⟩, когда магазин пуст. Иногда принимается соглашение, что в этом случае происходит «поломка» или зацикливание. Этот подход можно было бы применять и в данной работе.

(4.7) Определение. Обозначим через m элемент множества маркеров $M = \{M_1, M_2, \ldots\}$. $\mathbf{P}_m$ определяется как класс схем **P**, расширенный за счет добавления операторов

$$\langle S\rangle ::= v \leftarrow m \mid \textbf{если } v = m \textbf{ то } [l:]\langle S\rangle \textbf{ иначе } [l:]\langle S\rangle$$

Предполагается, что при любой интерпретации M не пересекается с областью D. Отметим, что в любой схеме из $\mathbf{P}_m$ может использоваться лишь конечное число маркеров.

Коль скоро заданы классы схем **P**, $\mathbf{P}_R$, $\mathbf{P}_A$, $\mathbf{P}_{Ae}$, $\mathbf{P}_L$, $\mathbf{P}_m$ и $\mathbf{P}_{pds}$, можно естественным образом порождать новые классы. Так, например, $\mathbf{P}_{AL} \equiv \mathbf{P}_{LA}$ — это класс схем программ с массивами и переменными метками. При этом, конечно, в тех случаях, когда взаимодействуют различные механизмы, на интерпретации налагаются очевидные ограничения. Например, в классе $\mathbf{P}_{AL}$ оператор **на** v равносилен пустому оператору, когда значение v принадлежит $\mathbb{N}$. В то же время $v \ominus 0$ истинно, когда значением v является некоторая метка.

Для нас наибольший интерес представляют классы $\mathbf{P}_A$, $\mathbf{P}_{Ae}$, $\mathbf{P}_{AL}$, $\mathbf{P}_{Am}$ и $\mathbf{P}_{pdsm}$. Мы покажем, что они имеют одинаковые вычислительные мощности. Однако эквивалентность между $\mathbf{P}_A$ и остальными классами неэффективна. Так, например, если дана схема из $\mathbf{P}_{Am}$, то эквивалентная ей схема в $\mathbf{P}_A$ существует, но ее не всегда можно эффективно построить.

5. СООТНОШЕНИЕ МЕЖДУ $\mathbf{P}_A$ И $\mathbf{P}_R$

Сначала покажем, как по данной схеме из $\mathbf{P}_R$ построить эквивалентную ей схему из $\mathbf{P}_{AL}$. Затем покажем, как избавиться от переменных-меток, чтобы получить эквивалентную схему из

$\mathbf{P}_A$. Таким образом, массивы индексированных переменных — средство по крайней мере той же мощности, что и рекурсивные процедуры.

Схема из $\mathbf{P}_{AL}$, эквивалентная данной схеме из $\mathbf{P}_R$, использует массив A в качестве стека для хранения параметров, локальных переменных и информации о возвратах функций, вызовы которых выполняются в текущий момент. Эта информация, необходимая для выполнения конкретного вызова функции, размещается в расположенных подряд элементах массива A; множество этих элементов называется *областью данных*. В типичных реализациях рекурсивных процедур алголоподобных языков память соответствующей области данных освобождается при завершении выполнения вызова функции и ее можно использовать для других целей. Мы не заботимся ни о какой эффективности и не будем освобождать эту память.

Глобальная переменная ТОР будет всегда содержать значение i индекса последнего (в текущий момент) элемента массива. Другая переменная, АЗ (активная зона), содержит индекс j начала области данных функции, вызов которой непосредственно выполняется в текущий момент (*активной* функции). Если ее область данных занимает $k+1$ ячейку, то она состоит из элементов A [АЗ], A [АЗ + 1], ..., A [АЗ + k].

Рассмотрим функцию f. Предположим, что она использует переменные $V_1, V_2, \ldots, V_p$, где $V_1, \ldots, V_{Rf}$ — ее формальные параметры. Тогда необходимая для выполнения вызова f область данных занимает $p+2$ ячейки и имеет следующий формат:

Текущее значение переменной АЗ
V_1
. . .
V_p
Метка возврата

Метка возврата — это метка оператора, к которому нужно вернуть управление после завершения выполнения данного вызова.

Вызов f в рекурсивной схеме заменяется сложным оператором, который 1) размещает $p+2$ ячейки под область данных, 2) запоминает индекс АЗ начала текущей области данных, 3) помещает метку возврата и значения фактических парамет-

ров в новую область данных, 4) приписывает неопределенное значение Ω всем остальным используемым функцией переменным; 5) присваивает глобальной переменной АЗ индекс начала новой области данных и 6) передает управление на первый оператор ⟨тела⟩ f для начала его выполнения.

В ⟨теле⟩ каждой функции вхождение V_i заменяется на $A[\text{АЗ} + i]$, поскольку используемые функцией переменные хранятся сейчас в области данных, определяемой содержимым АЗ. Каждый оператор СТОП(V_i) исходного определения функции заменяется на

начало $RV \leftarrow A[\text{АЗ} + i]$; **на** $A[\text{АЗ} + (p + 1)]$ **конец**

где RV — глобальная переменная. Таким образом, значение функции, полученное в результате ее выполнения, запоминается в RV, а управление передается оператору, определяемому меткой возврата, т. е. непосредственно следующему за вызовом функции. В этой точке возврата восстанавливается область данных предыдущего вызова, а значение RV присваивается подходящей переменной:

$$\text{АЗ} \leftarrow A[\text{АЗ}];\ v \leftarrow RV$$

Этого эскиза доказательства должно быть достаточно, чтобы убедить читателя; подробно конструкция приводится в приложении.

(5.1) Теорема. *По данной схеме из* $\mathbf{P}_R$ *можно эффективно построить эквивалентную схему из* $\mathbf{P}_{AL}$.

(5.2) Пример. Пусть дана схема $S \in \mathbf{P}_R$:

```
(X, Y):   если P(X) то Y ← f(X) иначе;
          СТОП(Y);
  f(X):   если Q(X) то V ← X
          иначе начало V ← g(X); V ← f(V) конец;
          СТОП(V)
```

Тогда схема S' такова:

```
(X, Y):   ТОР ← 0; АЗ ← 0;
          если P(X) то
            начало ТОР ← ТОР + 1;                          ┐
              A[ТОР] ← АЗ; A[ТОР + 3] ← RL1;               │
              A[ТОР + 1] ← X;                              │
              A[ТОР + 2] ← ОМЕГА;                          │ Y ← f(X)
              АЗ ← ТОР; ТОР ← ТОР + 3;                     │
              на Lf;                                       │
  RL1:      АЗ ← A[АЗ]; Y ← RV                           ┘
              конец
            иначе;
          СТОП(Y);
```

```
Lf:   если Q(A[АЗ + 1]) то                                  ⎤
          A[АЗ + 2] ← A[АЗ + 1]                              ⎦ V ← X
      иначе начало A[АЗ+2] ← g(A[АЗ + 1]);                   ⎤ V ← g(X)

          начало ТОР ← ТОР + 1;                              ⎫
            A[ТОР] ← АЗ;                                     ⎪
            A[ТОР + 3] ← RL2;                                ⎪
            A[ТОР + 1] ← A[АЗ + 1];                          ⎪
            A[ТОР + 2] ← ОМЕГА;                              ⎪ V ← f(V)
            АЗ ← ТОР; ТОР ← ТОР + 3;                         ⎪
            на Lf;                                           ⎪
            RL2:    АЗ ← A[АЗ];                              ⎪
            A[АЗ + 2] ← RV                                   ⎪
          конец                                              ⎭

      конец;
   начало RV ← A[АЗ + 2]; на A[АЗ + 3]                       ⎤ СТОП(V)
      конец                                                  ⎦
```

Покажем теперь, как по любой схеме из $\mathbf{P}_R$ построить эквивалентную схему из $\mathbf{P}_A$; таким образом, $\mathbb{F}(\mathbf{P}_R, D) \subseteq \mathbb{F}(\mathbf{P}_A, D) \subseteq \mathbb{F}(\mathbf{P}_{AL}, D)$. Это построение основано на следующей теореме:

(5.3) Теорема. *Пусть* $S \in \mathbf{P}_{AL}$, *и известно, что существует базисный предикат* P *ранга* n *и такие значения простых переменных* $RT_1, \ldots, RT_n$ *и* $RF_1, \ldots, RF_n$, *что* $P(RT_1, \ldots, RT_n) = $ И, *а* $P(RF_1, \ldots, RF_n) = $ Л. *Тогда можно построить эквивалентную схему* S' *из* $\mathbf{P}_A$. S' *называется* P*-имитатором схемы* S.

Построение. Без ограничения общности будем считать, что P имеет ранг 1, а простые переменные RT и RF, для которых $P(RT) = $ И, $P(RF) = $ Л, не меняют своих значений rt и rf во время выполнения схемы S.

Предположим, что в операторах типа $v \leftarrow l$ используются метки $L_1, L_2, \ldots, L_k$. Тогда каждой простой переменной v схемы S соответствуют переменные $v, v^0, v^1, \ldots, v^k$ схемы S', а каждому массиву A — массивы $A, A^0, A^1, \ldots, A^k$. Если во время выполнения схемы S переменная v принимает значение из $D \cup \mathbb{N}$, то в S' она принимает то же значение, в то время как $v^0 = rt$. Если же в S $v = L_i$, то в S' $v = \Omega$, $v^i = rt$, $v^j = rf$ для $0 \leqslant j \leqslant k$, $j \neq i$.

Теперь понятно, как преобразовать схему из $\mathbf{P}_{AL}$ в схему из $\mathbf{P}_A$:

а) каждое присваивание $v \leftarrow w$ заменяется на

начало $v \leftarrow w$; $v^0 \leftarrow w^0, \ldots, v^k \leftarrow w^k$ **конец**

б) каждое присваивание $v \leftarrow L_i$ заменяется на

начало $v \leftarrow$ ОМЕГА; $v^0 \leftarrow RF$; ...; $v^k \leftarrow RF$; $v^i \leftarrow RT$ **конец**

в) каждое присваивание $v \leftarrow f(\ldots)$ заменяется на

начало $v \leftarrow f(\ldots)$; $v^0 \leftarrow RT$ **конец**

г) каждый оператор **на** v заменяется на

если $P(v^0)$
то
иначе если $P(v^1)$ **то на** L_1
иначе если $P(v^2)$ **то на** L_2
иначе ...
иначе если $P(v^k)$ **то на** L_k
иначе

Отметим, что оператор **на** v моделируется корректно и тогда, когда значение v не является меткой — в этом случае управление передается на следующий оператор. Заметим также, что если переменной v ни разу не присваивалось значение, то $v = v^0 = \ldots = v^k = \Omega$. Моделирование снова корректно независимо от того, $P(\Omega) =$ И или $P(\Omega) =$ Л. Построение завершено.

(5.4) Теорема. *Пусть* $S \in \mathbf{P}_{Am}$. *Пусть известно, что существует базисный предикат* P *ранга* n *и такие значения простых переменных* $RT_1, \ldots, RT_n$ *и* $RF_1, \ldots, RF_n$, *что* $P(RT_1, \ldots, RT_n) =$ И, $P(RF_1, \ldots, RF_n) =$ Л. *Тогда можно построить эквивалентную схему* S' *из* $\mathbf{P}_A$. S' *называется* P*-имитатором схемы* S.

Доказательство предоставляется читателю.

Проблема, конечно, заключается в нахождении некоторого предиката P, который не является ни тождественно истинным, ни тождественно ложным (если такой вообще существует), а также соответствующих аргументов. Определим теперь по данной схеме S ЛОКАТОР, который это осуществляет.

(5.5) Определение. Пусть S — схема из любого класса схем. ЛОКАТОР для S — это схема S', обладающая следующими свойствами:

1) Множества входных переменных схем S и S' совпадают.

2) Если выполнение схемы S завершается с результатом v, прежде чем некоторый предикат дважды вычисляется с разными результатами, то так же работает и S'.

3) Предположим, что во время выполнения схемы S предикат P_i вычислен на аргументах a_1, a_2 и $P_i(a_1) =$ И, $P_i(a_2) =$ Л.

Тогда соответствующее выполнение схемы S' осуществляет такие действия:

а) аргументы a_1 запоминаются в ячейках $RT_1, \ldots, RT_{RP_i}$;

б) аргументы a_2 запоминаются в ячейках $RF_1, \ldots, RF_{RP_i}$;

в) управление передается на оператор

НАЧАЛОi: СТОП(ОМЕГА)

где НАЧАЛОi — новая метка.

Заметим, что если выполнен оператор, помеченный меткой НАЧАЛОi, то в результате получается Ω. Сам по себе ЛОКАТОР бессмыслен, но, объединяя его с P_i-имитатором схемы S, можно получить схему, эквивалентную S.

(5.6) Теорема. *Пусть S из $\mathbf{P}_{AL}$ (или $\mathbf{P}_{Am}$) имеет* ЛОКАТОР S' *в $\mathbf{P}_A$. Тогда в $\mathbf{P}_A$ существует схема, эквивалентная S.*

Доказательство. Пусть ЛОКАТОР из $\mathbf{P}_A$ — это

$$(v_1, \ldots, v_m)\text{: } \langle S\text{-список}\rangle$$

Предположим, что в S используются n предикатов $P_1, \ldots, P_n$, и пусть для $i = 1, \ldots, n$ P_i-имитатором из $\mathbf{P}_A$ для схемы S является

$$(v_1, v_2, \ldots, v_m)\text{: } \langle S\text{-список}\rangle_i$$

Тогда ясно, что схема

$(v_1, \ldots, v_m)$: $W_1 \leftarrow v_1$; $\ldots$; $W_m \leftarrow v_m$; $\langle S$-список$\rangle$;

НАЧАЛО1: $v_1 \leftarrow W_1$; $\ldots$; $v_m \leftarrow W_m$; $\langle S$-список$\rangle_1$;

.

.

.

НАЧАЛОn: $v_1 \leftarrow W_1$; $\ldots$; $v_m \leftarrow W_m$; $\langle S$-список$\rangle_n$

(в $\langle S$-списке$\rangle$ ЛОКАТОРа все операторы НАЧАЛОi: СТОП(ОМЕГА) заменены пустыми операторами) эквивалентна S.

Здесь $W_1, \ldots, W_m$ — новые переменные. Напомним, что ЛОКАТОР находит предикат P_i, принимающий разные значения, присваивает значения переменным RT_j, RF_j, $j = 1, \ldots, RP_i$ и передает управление на метку НАЧАЛОi. В каждом имитаторе используются глобальные переменные RT_j, RF_j. Теорема доказана.

По схеме $S \in \mathbf{P}_R$ можно построить эквивалентную схему $S' \in \mathbf{P}_{AL}$. Покажем, как для S' (или, что то же самое, для S) построить ЛОКАТОР, принадлежащий $\mathbf{P}_A$.

(5.7) Построение принадлежащего классу $\mathbf{P}_A$ ЛОКАТОРа для схемы $S' \in \mathbf{P}_{AL}$, полученной по $S \in \mathbf{P}_R$. Без ограничения общности будем считать, что все предикаты имеют ранг 1.

1) Берется копия S, и в ней раскрываются все вызовы первого уровня (конструкция (3.9)).

2) Результат этапа 1 транслируется в $\mathbf{P}_{AL}$. В полученной схеме операторами вида **на** v будут *только* возвраты функций, **на** $A[\text{АЗ} + (p+1)]$, с помощью которых заменялись операторы СТОП(v) в исходных определениях функций. Ясно, что результат этого этапа эквивалентен и S, и S'.

3) Каждый оператор вида

если $P_i(v)$ **то** S_1 **иначе** S_2

заменяется на

если $P_i(v)$
то если $P_i(RT)$ **то** S_1
иначе начало $RF \leftarrow RT$; $RT \leftarrow v$;
на НАЧАЛОi
конец
иначе если $P_i(RT)$ **то начало** $RF \leftarrow v$; **на** НАЧАЛОi
конец
иначе S_2

4) каждый оператор вида $v \leftarrow l$ или **на** v заменяется пустым оператором.

5) В конце схемы добавляются операторы

НАЧАЛОi: СТОП(ОМЕГА)

Результат описанной трансляции и есть ЛОКАТОР. Все операторы в нем допустимы в $\mathbf{P}_A$, поскольку все переменные-метки уничтожены на этапе 4. Осталось показать, что эта схема работает так, как требуется. Во-первых, если во время выполнения вычисляется такой предикат P_i, что $P_i(\ldots) \neq P_i(RT)$, то мы присваиваем соответствующие значения переменным RT и RF и переходим на НАЧАЛОi. Таким образом, если некоторый предикат вычисляется дважды с разными результатами, то выполнение схемы прекращается. Во-вторых, отметим, что все вызовы первого уровня раскрыты. Поэтому, в силу следствия (3.11), прежде чем внутри расширенного тела функции в схеме из $\mathbf{P}_R$ будет выполнен *хотя бы один* оператор СТОП, некоторый предикат будет вычислен дважды с разными результатами. Следовательно, какова бы ни была функция, еще до возврата из ее тела ЛОКАТОР передаст управление на НАЧАЛОi. Поэтому возвраты не нужны, и их можно опустить. Это означает, что этап 4 не влияет на вычисления схемы.

(5.8) Теорема. $\mathbb{F}(\mathbf{P}_R, D) \subseteq \mathbb{F}(\mathbf{P}_A, D) \subseteq \mathbb{F}(\mathbf{P}_{AL}, D)$.

Доказательство. В силу (5.1), можно построить схему $S' \in \mathbf{P}_{AL}$, эквивалентную схеме $S \in \mathbf{P}_R$. Далее, для S' существует ЛОКАТОР в $\mathbf{P}_A$ (5.7). Осталось применить теорему (5.6).

Построение схемы из $\mathbf{P}_A$ по схеме из $\mathbf{P}_R$ было эффективным. Сейчас мы покажем, что в $\mathbf{P}_A$ существует схема, не эквивалентная никакой схеме из $\mathbf{P}_R$; это означает, что массивы — более мощное средство, чем рекурсия!

В [5] Патерсон посредством нового оператора /*OR*/ ввел некий тип параллелизма:

(5.9) Определение. Значение функции $P(x)/OR/Q(x)$
а) истинно, когда хотя бы одно из $P(x)$, $Q(x)$ истинно;
б) ложно, когда оба $P(x)$ и $Q(x)$ ложны;
в) не определено, когда оба $P(x)$ и $Q(x)$ не определены, либо одно ложно, а другое не определено.

(5.10) Пример. (X): **если** $Q(X)$ **то** $V \leftarrow F_1(X)$ **иначе** $V \leftarrow X$;
СТОП(V)
$Q(X)$: **если** $P(X)$ **то истина**
иначе $V_1 \leftarrow Q(L(X))/OR/Q(R(X))$
СТОП(V_1)

В работе [5] показано, что эта схема не транслируема в $\mathbf{P}_R$.

(5.11) Теорема. $\mathbb{F}(\mathbf{P}_R, D) \subset \mathbb{F}(\mathbf{P}_A, D) \subset \mathbb{F}(\mathbf{P}_{AL}, D)$.

Доказательство. Докажем, что схема (5.10) транслируема в $\mathbf{P}_A$. Будем писать $LR(x)$, $LL(x)$, $RL(x)$ и т. д. вместо $L(R(x))$, $L(L(x))$, $R(L(x))$ и т. д. Схема (5.10) останавливается тогда и только тогда, когда существует такая цепочка $\alpha \in \{L, R\}^*$, что $Q(\alpha(x)) = $ И. Идея доказательства состоит в том, чтобы использовать массив для реализации поиска «ширины один» в двоичном дереве с корнем x (от каждой вершины x ответвляются $L(x)$ и $R(x)$).

Искомая схема из P_A такова:

```
(X):    TOP ← 0; CURRENT ← 0; A[TOP] ← X;
AGAIN:  если P(A[CURRENT]) то начало V ← F1(X); СТОП(V)
                                конец
        иначе начало TOP ← TOP + 1;
                     A[TOP] ← L(A[CURRENT]);
                     TOP ← TOP + 1;
                     A[TOP] ← R(A[CURRENT]);
                     CURRENT ← CURRENT + 1;
                     на AGAIN
               конец
```

6. СООТНОШЕНИЕ МЕЖДУ $\mathbf{P}_{pds}$ И $\mathbf{P}_R$

Обозначим через $\mathbf{P}_{(n, m)}$ подкласс схем класса $\mathbf{P}_{pdsm}$, использующих n магазинов и m маркеров. В этом разделе мы опишем конструкции, показывающие, что

$$\mathbb{F}(\mathbf{P}_{(n, m)}, D) \equiv \mathbb{F}(\mathbf{P}_{(2, n+m+1)}, D) \equiv \mathbb{F}(\mathbf{P}_{(2, 2)}, D) \text{ для } n, m \geqslant 2,$$
$$\mathbb{F}(\mathbf{P}_R, D) \subseteq \mathbb{F}(\mathbf{P}_{(1, 0)}, D).$$

Аналогичные результаты относительно использования массивов $\mathbf{P}_A$ приведены в разд. 9.

(6.1) Лемма. $\mathbb{F}(\mathbf{P}_{n, m}, D) \subseteq \mathbb{F}(\mathbf{P}_{(2, n+m+1)}, D)$.

Набросок доказательства. Пусть в $S \in \mathbf{P}_{(n, m)}$ используются магазины $PD_1, \ldots, PD_n$ и маркеры $M_1, \ldots, M_m$. Введем новые маркеры $A_0, A_1, \ldots, A_n$.

Вхождение магазина PD_i в S заменяется в соответствующей схеме $S' \in \mathbf{P}_{(2, n+m+1)}$ вхождением ее основного магазина PD.

Оператор $\langle$pushdown$\rangle$ $PD_i \leftarrow v$ схемы S заменяется в S' на

начало $PD \leftarrow v$; $PD \leftarrow A_i$ **конец**

Таким образом, A_i в PD указывает, что заслано значение верхушки магазина PD_i. Маркер A_0 помещается в низ магазина PD и указывает, когда он становится пустым.

Оператор $\langle$popup$\rangle$ $v \leftarrow PD_i$ моделируется в S' составным оператором, который в поисках маркера A_i осуществляет просмотр PD сверху вниз. Значение, хранящееся в PD непосредственно ниже A_i, и есть то, что нужно заслать в v. Этот просмотр требует второго, дополнительного магазина для сохранения содержимого PD. После считывания значения в ячейку v содержимое основного магазина PD восстанавливается с помощью дополнительного магазина. Подробности см. в приложении.

(6.2) Лемма. $\mathbb{F}(\mathbf{P}_{(2, m)}, D) \subseteq \mathbb{F}(\mathbf{P}_{(2, 2)}, D)$.

Эскиз доказательства. Маркеры, используемые в $(2, m)$-схеме S, можно двоично кодировать и декодировать, применяя конечное число операторов схемы программ. Пусть, например, $M_1, \ldots, M_m$ $(m \geqslant 2)$ — используемые в S маркеры. Тогда эквивалентная схема $S' \in \mathbf{P}_{(2, 2)}$ использует маркеры M_0 и M, а маркер M_i из S представляется в виде

$$\underbrace{M_0 M \ldots M}_{i-1 \text{ раз}} M \underbrace{M_0 M \ldots M}_{m-i \text{ раз}}.$$

Подробности см. в приложении.

(6.3) Теорема. $\mathbb{F}(\mathbf{P}_{(n,m)}, D) \equiv \mathbb{F}(\mathbf{P}_{(2,2)}, D)$ *для* $n, m \geqslant 2$.

Доказательство. Утверждение следует из лемм (6.1) и (6.2).

(6.4) Теорема. $\mathbb{F}(\mathbf{P}_R, D) \subseteq \mathbb{F}(\mathbf{P}_{(1,0)}, D)$.

Набросок доказательства. Рассуждаем так же, как в разд. 5 при доказательстве включения $\mathbb{F}(\mathbf{P}_R, D) \subseteq \subseteq \mathbb{F}(\mathbf{P}_A, D)$. В одном магазине (скажем, PD) хранятся области данных всех активных в текущий момент функций. Как и раньше, можно записать ЛОКАТОР и P_i-имитатор; на этот раз они будут из класса $\mathbf{P}_{(1,0)}$. Единственный серьезный вопрос состоит в том, действительно ли при выполнении вызова функции имеется доступ ко всем переменным из области данных. Этого можно добиться с помощью конечного числа простых переменных схемы, поскольку каждая область данных имеет известные конечные размеры. Подробности см. в приложении.

7. ЭКВИВАЛЕНТНОСТЬ КЛАССОВ $\mathbf{P}_{AL}$, $\mathbf{P}_{Am}$, $\mathbf{P}_{pdsm}$ И $\mathbf{P}_{Ae}$

Сначала докажем эквивалентность $\mathbf{P}_{AL}$ и $\mathbf{P}_{Am}$. Затем в леммах (7.2) и (7.3) установим эквивалентность $\mathbf{P}_{Am}$ и $\mathbf{P}_{pdsm}$. Можно также показать эквивалентность классов $\mathbf{P}_{pdsm}$ и $\mathbf{P}_{pdsL}$, но здесь мы этого делать не будем.

Мы считаем, вообще говоря, что класс $\mathbf{P}_{Am}$ наиболее предпочтителен из классов схем. Алгоритмы, использующие магазинную память, часто выглядят очень громоздкими, в то время как массивы — более естественное средство, с помощью которого можно довольно просто моделировать магазинную память. Класс $\mathbf{P}_m$ кажется нам предпочтительнее класса $\mathbf{P}_L$. Вообще, чем меньше операторов перехода участвуют в программе, тем она понятней, наглядней, легче читается, легче отлаживается, а особенности употребления меток как переменных приводят лишь к увеличению количества разветвлений.

(7.1) Теорема. $\mathbb{F}(\mathbf{P}_{AL}, D) \equiv \mathbb{F}(\mathbf{P}_{Am}, D)$.

Доказательство. Покажем, что по данной схеме $S \in \in \mathbf{P}_{AL}$ можно построить эквивалентную схему $S' \in \boldsymbol{P}_{Am}$, и обратно.

Пусть в схеме S имеется n меток $L_1, \ldots, L_n$. Тогда в схеме S' будут использоваться соответственно маркеры $M_1, \ldots, M_n$. Каждый оператор $v \leftarrow L_i$ в S заменим оператором $v \leftarrow M_i$.

Каждый оператор **на** v в S заменим на

начало если $v = M_1$ **то на** L_1
иначе если $v = M_2$ **то на** L_2
иначе ...
если $v = M_n$ **то на** L_n
иначе
конец

В результате получим, очевидно, эквивалентную схему $S' \in \mathbf{P}_{Am}$.

Пусть теперь дана схема $S' \in \mathbf{P}_{Am}$. Предположим, что в ней используются маркеры $M_1, M_2, \ldots, M_n$. Будем рассматривать их как метки и дополним схему последовательностью операторов

M_1: **на** v_1;
M_2: **на** v_2;
. . . . . .
M_n: **на** v_n;

где v_i — новые простые переменные. Для всякого оператора

если $v = M_t$ **то** S_1 **иначе** S_2

порождаются три новые метки LT, LF и DONE, а сам оператор заменяется на

начало $v_1 \leftarrow LF$; $v_2 \leftarrow LF$; ... $v_n \leftarrow LF$; $v_t \leftarrow LT$;
на v;
LF: S_2; **на** DONE;
LT: S_1;
DONE:
конец

Мы видим, что при выполнении последовательности операторов схема S_1 выполняется тогда и только тогда, когда $v = M_t$; в противном случае выполняется S_2. Это и есть требуемое свойство.

(7.2) Лемма. $\mathbb{F}(\mathbf{P}_{pdsm}, D) \subseteq \mathbb{F}(\mathbf{P}_{Am}, D)$.

Доказательство. По данной схеме $S \in \mathbf{P}_{pdsm}$ построим эквивалентную ей схему $S' \in \mathbf{P}_{Am}$. Каждый магазин PD схемы S моделируется некоторым массивом A схемы S'. Счетчик CA указывает, где находится верхушка массива A. Каждый элемент магазина PD представляется двумя последовательными

ячейками массива A, имеющими формат

Фактическое содержимое элемента магазина PD
Маркер

Вторая ячейка указывает, пуст ли магазин PD. Для этой цели в схеме класса $\mathbf{P}_{Am}$ применяются два дополнительных маркера M_0 и M_1. Пустой магазин представляется так, как показано на рис. 7.1, *а*, тогда как магазин PD на рис. 7.1, *б* можно было бы описывать массивом A на рис. 7.1, *в*.

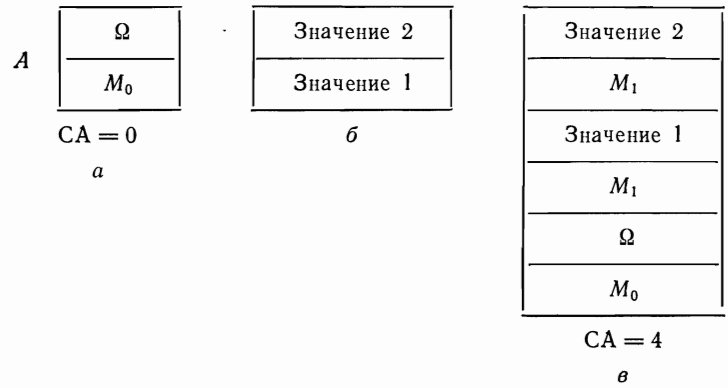


Рис. 7.1. Моделирование магазинов массивом: *а* — пустой магазин, *б* — PD с двумя элементами, *в* — соответствующий массив A.

Нетрудно понять, как построить схему S', эквивалентную S. Для каждого магазина PD, используемого в S, нужно выполнить три шага:

1) Создать новое имя массива A и простую переменную CA и в начало схемы S поместить операторы, осуществляющие загрузку начальных значений для массива:

$$\text{CA} \leftarrow 0;\ A[0] \leftarrow M_0;\ A[1] \leftarrow \text{ОМЕГА};$$

2) Каждый оператор $\text{PD} \leftarrow v$ заменить на

начало $\text{CA} \leftarrow \text{CA} + 2;$
$A[\text{CA}] \leftarrow M_1;$
$A[\text{CA} + 1] \leftarrow v$
конец

3) Каждый оператор $v \leftarrow PD$ заменить на

начало если $A[\mathrm{CA}] = M_1$
то начало $v \leftarrow A[\mathrm{CA} + 1]$
$\mathrm{CA} \leftarrow \mathrm{CA} \dot{-} 2$
конец
иначе
конец

(В силу теоремы (4.3) можно взять функцию $\dot{-}1$, а стало быть, и $\dot{-}2$.)

(7.3) Лемма. $\mathbb{F}(\mathbf{P}_{Am}, D) \subseteq \mathbb{F}(\mathbf{P}_{pdsm}, D)$.

Доказательство. По данной схеме $S \in \mathbf{P}_{Am}$ построим эквивалентную схему $S' \in \mathbf{P}_{pdsm}$. Каждое значение из D и всякий маркер M, используемые в S, будут представлять сами себя в схеме S'. Пусть $*$ — новый маркер. Тогда каждое значение $i \in \mathbb{N}$, вычисляемое в схеме S, в схеме S' будет представляться последовательностью из $i + 1$ маркеров $*$. Поскольку любая переменная может принимать значение i (*исключая* переменные, которые используются как аргументы базисных функций или предикатов), каждая переменная схемы S должна быть представлена в схеме S' посредством магазина! (Если переменная используется как аргумент, то вместо нее можно взять либо Ω, либо просто одну звездочку.)

Наша задача по преобразованию схемы S в S' будет решаться в четыре этапа. Сначала построим схему S_1, эквивалентную схеме S. В схеме S_1 все аргументы функций, предикатов и операторов СТОП являются новыми простыми переменными, которые (в соответствии со способом их введения) нет нужды рассматривать как магазинные. На втором шаге строим схему S_2, эквивалентную схеме S_1, исключительно для упрощения перехода от простых переменных к магазинам. Например, оператор $A[v] \leftarrow A[w]$ заменяется на

начало $v_0 \leftarrow A[w]$; $A[v] \leftarrow v_0$ **конец**

где v_0 — новая простая переменная.

На третьем шаге рассматриваем простые переменные как магазинные и показываем, как изменить для этого каждый оператор. Например, мы должны преобразовать оператор $v \leftarrow w(v, w$ — простые переменные) в составной оператор, который копирует содержимое магазина w в магазин v.

Наконец, мы показываем, как представить массивы магазинами и преобразовать те два типа операторов, в которых используются массивы, а именно $A[w] \leftarrow v_1$ и $w \leftarrow A[v_2]$ (v_1, v_2 — простые переменные).

Каждый массив A представляется в общем случае некоторым магазином РА. В РА фактические значения массива A разделяются запятыми. Таким образом, массив A, изображенный на рис. 7.2, *а*, представляется магазином РА на рис. 7.2, *б*. Для

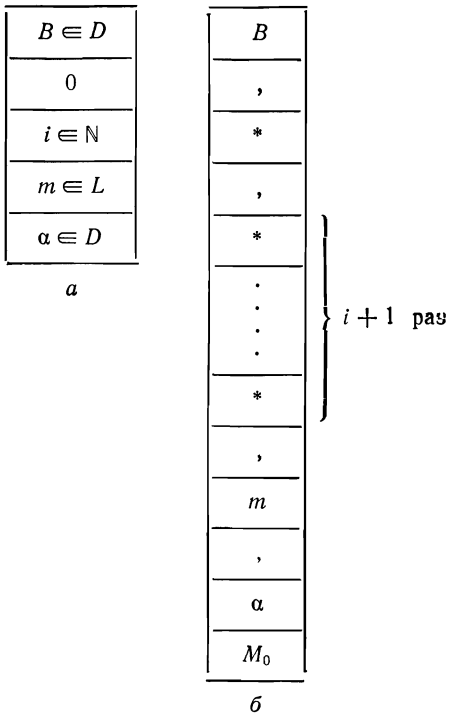


Рис. 7. 2. Моделирование массива A: *а* — массив A, *б* — соответствующий магазин PD.

того чтобы извлечь некоторое значение $A[i]$, мы сначала (в схеме S') копируем все значения магазина РА во вспомогательный магазин PDT1 (в обратном порядке). Затем с помощью представления i в виде $i+1$ звездочек стираются i запятых из PDT1, так что на верхушке магазина остается значение $A[i]$.

Подробности можно найти в приложении.

(7.4) Теорема. $\mathbb{F}(\mathbf{P}_{Am}, D) \equiv \mathbb{F}(\mathbf{P}_{pdsm}, D)$.

Доказательство. Утверждение следует из лемм (7.2) и (7.3).

(7.5) Лемма. $\mathbb{F}(\mathbf{P}_{Am}, D) \subseteq \mathbb{F}(\mathbf{P}_{Ae}, D)$.

Доказательство. Схемы класса $\mathbf{P}_{Ae}$ позволяют проверять на равенство значения индексов (см. определение (4.4)). Пусть дана схема $S \in \boldsymbol{P}_{Am}$ с маркерами $M_1, \ldots, M_n$. Мы транслируем S в эквивалентную схему S' класса P_{Ae} следующим образом.

Вместо n маркеров берем натуральные числа $1, 2, \ldots, n \in \mathbb{N}$. Однако мы должны уметь распознавать, является некоторое значение индексом или маркером. Для этого каждая простая переменная v (кроме ОМЕГА) схемы S представляется двумя переменными v и v_0, каждый массив A — двумя массивами A и A_0. (Мы используем ниже w_0 для представления как простой переменной w_0, так и индексированной переменной $w_0[v]$). Если $w = M_i$ во время выполнения схемы S, то $w \ominus i \in \mathbb{N}$, тогда как $w_0 = 1$. В противном случае w содержит обычное значение или индекс, а w_0 равно нулю или принимает значение из D.

Заметим, что в обоих случаях (и для простой, и для индексированной переменной w) w_0 и w вначале имеют значение Ω, так что сначала все правильно. Преобразуем теперь S так:

(1) Заменим каждый оператор $v \leftarrow w$ на

начало $v \leftarrow w$; $v_0 \leftarrow w_0$ **конец**

(2) Заменим каждый оператор $v \leftarrow f(\ldots)$ на

начало $v \leftarrow f(\ldots)$; $v_0 \leftarrow 0$ **конец**

(3) Заменим каждый оператор $v \leftarrow 0$ на

начало $v \leftarrow 0$; $v_0 \leftarrow 0$ **конец**

(4) Заменим каждый оператор $v \leftarrow w + 1$ на

если $w_0 \ominus 0$ **то начало** $v \leftarrow w + 1$; $v_0 \leftarrow 0$ **конец**
иначе $v \leftarrow 1$

(5) Заменим каждый оператор $v \leftarrow M_i$ на

начало $v \leftarrow i$; $v_0 \leftarrow 1$ **конец**

(6) Заменим каждый условный оператор

если $v \ominus M_i$ **то** S_1 **иначе** S_2

на

начало если $v_0 \ominus 0$ **то на** LS2;
если $v \ominus i$ **то** S_1
иначе LS2: S_2
конец

(7.6) Лемма. $\mathbb{F}(\mathbf{P}_{Ae}, D) \subseteq \mathbb{F}(\mathbf{P}_{Am}, D)$.

Доказательство. Пусть $S \in \mathbf{P}_{Ae}$. Покажем, как построить эквивалентную схему $S' \in \mathbf{P}_{Am}$. Для этого достаточно показать, как транслировать оператор

(7.7) если $w \ominus v$ **то** S_1 **иначе** S_2

где результат такой проверки определен в (4.4). Прежде всего введем новый массив M и два маркера m_0 и m_1. В начало схемы S вставим оператор $M[0] \leftarrow m_0$, а каждый оператор $v \leftarrow w + 1$ заменим на

начало $v \leftarrow w + 1$; $M[v] \leftarrow m_1$ **конец**

Это гарантирует, что если во время выполнения схемы (при некоторой интерпретации) вычисляется какое-то значение $i > 0$, то $M[0] = m_0$ и $M[1] = m_1$, ..., $M[i] = m_1$. Заменим теперь каждый оператор (7.7) в S на

начало $w1 \leftarrow w$; $v1 \leftarrow v$; L: **если** $M[w1] = m_0$
то если $M[v1] = m_0$ **то на** LS1
иначе на LS2
иначе если $M[v1] = m_0$ **то на** LS2
иначе;
$w1 \leftarrow w1 \dot{-} 1$; $v1 \leftarrow v1 \dot{-} 1$; **на** L;
LS1: S_1; **на** DONE; LS2: S_2;
DONE: **конец** [1])

Здесь L, LS1, LS2 и DONE — новые метки, а $v1$, $w1$ — новые простые переменные.

(7.8) Теорема. $\mathbb{F}(\mathbf{P}_{Ae}, D) \equiv \mathbb{F}(\mathbf{P}_{Am}, D)$.

Доказательство. Утверждение следует из лемм (7.5) и (7.6).

8. НЕЭФФЕКТИВНАЯ ЭКВИВАЛЕНТНОСТЬ КЛАССОВ $\mathbf{P}_A$ И $\mathbf{P}_{Am}$

В этом разделе мы покажем, что $\mathbf{P}_A \equiv \mathbf{P}_{Am}$ и что эта эквивалентность неэффективна.

По данной схеме S (из $\mathbf{P}_A$ или $\mathbf{P}_{Am}$) можно построить эквивалентную «тотально размеченную» схему, в которой все операторы (все операторы $\langle S \rangle$ в синтаксическом смысле, см. п. 2.1) помечены. Например, на рис. 8.1, *б* это сделано для схемы

[1]) Здесь в оригинале допущена ошибка: (7.7) заменяется на оператор, который «портит» переменные w и v, так что построенная схема будет неэквивалентна исходной. Мы приводим сразу правильную трансляцию для (7.7). — *Прим. перев.*

рис. 8.1, *а*. Поэтому можно предполагать, что каждая схема тотально размечена.

```
(v, w):   v ← F1(v, w);        (v  w):   L2:   v ← F1(v, w);
L1:       если P(v) то                   L1:    если P(v) то
          на L7                                    L3:    на L7
          иначе начало                   иначе L4:        начало
                  v ← F2(v);                              L5:  v ← F2(v);
                  на L1                                   L6:  на L1
                конец;                                    конец;
L7:       СТОП(v)                        L7:    СТОП(v)
```

а *б*

Рис. 8.1. Тотальная разметка схемы.

(8.1) Определение. *Поведением* (вычисления) *схемы* (при некоторой интерпретации) называется последовательность меток выполняемых операторов в том порядке, в котором они *начинают* выполняться.

Пусть схема на рис. 8.1, *б* выполняется при такой интерпретации, что при первом прохождении оператора, помеченного меткой L1, предикат $P(v)$ ложен, а при втором прохождении истинен. Тогда поведение схемы (при этой интерпретации) — это последовательность L2, L1, L4, L5, L6, L1, L3, L7. Для нас важным будет «автономное» поведение схемы.

(8.2) Определение. Пусть S — (тотально размеченная) схема из $\mathbf{P}_{Am}$, в которой используется n предикатов $P_1, \ldots, P_n$. Пусть $v = (v_1, \ldots, v_n)$ — вектор значений, каждое из которых либо **истина**, либо **ложь**. Тогда *v-автономным поведением* схемы S называется поведение схемы S в предположении, что

$$P_1 \equiv v_1, \quad P_2 \equiv v_2, \ \ldots, \ P_n \equiv v_n.$$

Заметим, что мы определили автономное поведение схемы независимо от базисных функций и входных значений какого бы то ни было вычисления. Разумеется, это не очевидно и нуждается в доказательстве.

(8.3) Лемма. *Пусть S — схема из $\mathbf{P}_{Am}$, использующая n предикатов $P_1, \ldots, P_n$. Предположим, что все предикаты постоянны. Тогда поведение схемы S не зависит от области D, интерпретации базисных функций и входных значений.*

Доказательство. Допустим, выполнение схемы начинается при двух различных интерпретациях или при одной и той

же интерпретации, но с разными входными значениями. Пусть при этом получаются две последовательности меток

$$L_1, L_2, \ldots, L_k, L_{k+1}, \ldots,$$
$$L_1, L_2, \ldots, L_k, L'_{k+1}, \ldots,$$

впервые различающиеся в $(k+1)$-й метке. Тогда очевидно, что метка L_k должна помечать условный оператор

если $P(\ )$ **то** S_1 **иначе** S_2

или

если $v = m$ **то** S_1 **иначе** S_2

Поскольку все предикаты постоянны, метка L_k должна помечать последний из двух приведенных условных операторов. Мы придем к противоречию с нашим предположением, если покажем, что

(8.4) к началу выполнения любого шага простая или индексированная переменная содержит при *обоих* вычислениях

(1) либо *один и тот же* маркер m,
(2) либо *одно и то же* значение индекса из $\mathbb{N}$,
(3) либо значения из области D (возможно, различные).

Докажем это утверждение индукцией по числу операторов, выполнение которых уже начато. Для $i = 1$ в момент начала выполнения оператора, помеченного L_1, все переменные принимают значения из области D. Пусть (8.4) справедливо для $n = 1, 2, \ldots, i < k$, и мы начинаем выполнять оператор, помеченный меткой L_i. Все операторы, способные изменить какое-то значение (и, возможно, привести к противоречию с (8.4)), имеют одну из следующих пяти форм:

(1) $v \leftarrow w$ (v и w — простые или индексированные переменные)
(2) $v \leftarrow m$
(3) $v \leftarrow 0$
(4) $v \leftarrow F(\ldots)$
(5) $v \leftarrow w + 1$

Поскольку при всяком вычислении вплоть до k-го шага выполняются *одни и те же* операторы, то после выполнения оператора L_i утверждение (8.4) будет справедливо независимо от того, какой из операторов (1)—(5) был выполнен.

Теперь можно начать обсуждение неконструктивной эквивалентности классов $\mathbf{P}_A$ и $\mathbf{P}_{Am}$.

(8.5) Теорема. *Пусть v — вектор длины n, составленный из логических значений* **истина** *и* **ложь**, *а S — схема из $\mathbf{P}_A$ с n предикатами. Тогда свойства конечности и бесконечности v-автономного поведения схемы S алгоритмически распознаваемы.*

Доказательство. Поскольку v-автономность поведения не зависит от интерпретации, можно начать построение такого поведения, не задавая никакой интерпретации. Пусть в S есть p меток (а стало быть, p операторов $\langle S \rangle$). Будем выполнять схему и записывать ее поведение до тех пор, пока либо не завершится вычисление, либо не будет записана $p+1$ метка. Если схема останавливается прежде, чем записана $p+1$ метка, то v-автономное поведение конечно. Если записана $p+1$ метка, то хотя бы одна метка была записана более чем один раз. Таким образом, какой-то оператор выполняется более одного раза, что означает наличие цикла. Этот зациклившийся процесс не может остановиться, поскольку все предикаты постоянны и единственная возможность изменить последовательность вычислений заключается в выполнении условного оператора

если $P(\ldots)$ **то** L_1: S_1 **иначе** L_2: S_2

где предикат $P(\ldots)$ принимает разные значения. Но этого быть не может.

(8.6) Лемма. *Проблема распознавания конечности и бесконечности v-автономного поведения схемы $S \in \mathbf{P}_{Am}$ алгоритмически неразрешима.*

Доказательство. По данной машине Тьюринга M с одной лентой, бесконечной вправо, и внешним алфавитом $\{0, 1\}$ можно построить схему программ $S_M \in \boldsymbol{P}_{Am}$, которая имеет конечное автономное поведение тогда и только тогда, когда M останавливается на пустой ленте (заполненной нулями). Тем самым проблема остановки машины Тьюринга сводится к проблеме конечности автономного поведения, а поскольку проблема остановки алгоритмически неразрешима, то неразрешима и проблема конечности.

Лента машины Тьюринга моделируется массивом T со счетчиком I (сначала его значение равно 0), указывающим позицию головки машины Тьюринга. Используются также два маркера 0 и 1. Если M записывает маркер m в i-ю ячейку ленты, то S_M выполняет оператор $T[i] \leftarrow m$. Если M считывает ячейку, то схема S_M осуществляет проверку

если $T[i] = m$ **то** ...

Если головка M сдвигается вправо, S_M выполняет оператор $i \leftarrow i+1$, а если головка M сдвигается влево, то, в силу теоремы

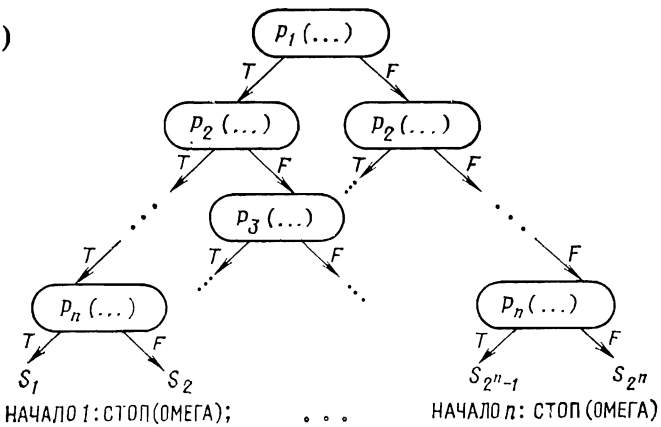
(4.3), схема S_M может выполнять оператор $i \leftarrow i \dot{-} 1$. Детали доказательства предоставляем читателю.

(8.7) Теорема. *По схеме* $S \in \mathbf{P}_{Am}$ *нельзя эффективно построить эквивалентную схему* $S' \in \mathbf{P}_A$.

Доказательство. Если схемы $S' \in \mathbf{P}_A$ и $S \in \mathbf{P}_{Am}$ эквивалентны, то v-автономное поведение схемы S конечно тогда и только тогда, когда конечно соответствующее поведение S'. Если бы схему S' можно было эффективно построить по S, то, применяя алгоритм, распознающий конечность автономного поведения S, мы получили бы противоречие с леммой (8.6).

Докажем, наконец, существование (но не эффективное построение) схемы $S' \in \mathbf{P}_A$, эквивалентной схеме $S \in \mathbf{P}_{Am}$. Теорема (5.6) показывает, что достаточно доказать существование ЛОКАТОРа в $\mathbf{P}_A$ для схемы S (см. определение (5.5)). Если в S используется n предикатов $P_1, \ldots, P_n$, то ЛОКАТОР имеет вид

(8.8)



где аргументы предикатов — все ОМЕГА, T — истина, F — ложь. (Таким образом, если у P_1 два аргумента, то в самом верхнем распознавателе будет написано P_1(ОМЕГА, ОМЕГА).) Отметим, что в качестве аргументов не обязательно брать неопределенное значение Ω, это лишь упрощает конструкцию. Здесь фактически можно было бы взять произвольные входные значения.

Предположим, что задана какая-то интерпретация области D, предикатов P_i и функций F_i и что ЛОКАТОР указанного выше вида выполняется для некоторых входных значений. Тогда ЛОКАТОР проверяет *все* предикаты P_i и выполняет один из операторов S_i в зависимости от того, какие значения истинно-

сти будут у предикатов. Существует 2^n различных операторов S_i в соответствии с 2^n возможными v-автономными поведениями схемы S. Оператор S_1, например, соответствует набору $v \equiv$ $\equiv$ (**истина**, ..., **истина**).

Предположим, что упомянутое выше вычисление обнаруживает, что $P_1(\ldots) = \ldots = P_n(\ldots) =$ **истина**. Тогда $v =$ (**истина**, ..., **истина**). Оператор S_1 должен осуществлять следующие действия:

(8.9) S_1 должен «моделировать» v-автономное поведение схемы S до тех пор, пока
(1) либо S_1 останавливается с тем же результатом, что и у S,
(2) либо будут осуществлены две проверки предиката P_i, дающие разные значения истинности. В этот момент в $RT_1, \ldots, RT_n$ засылается Ω, а в $RF_1, \ldots, RF_n$ — тот список аргументов $P_i(\ldots)$, который дает $P_i(\ldots) =$ **ложь**. Управление передается затем на метку НАЧАЛО i.

Таким образом, поскольку P_i(ОМЕГА) = **истина** для $i = 1, \ldots, n$, то S_1 считает все предикаты постоянными (и S_1 выполняется, по существу, как и S) до тех пор, пока не обнаруживает различия.

Теперь покажем, как построить S_1 (для $v =$ (**истина**, ..., **истина**). Построение остальных операторов S_i осуществляется подобным образом.

Рассмотрим два случая, связанные с тем, конечно или бесконечно v-автономное поведение схемы S.

(8.10) Построение S_1 в предположении конечности v-автономного поведения схемы S:

$$L_1,\ L_2,\ \ldots,\ L_k.$$

Не уменьшая общности, будем считать, что все предикаты имеют ранг 1. Приведенная выше последовательность показывает, в каком порядке начинается выполнение операторов. Нам, по существу, нужно «развернуть» «массив» циклов. Запишем нашу последовательность операторов в виде

(8.11)
начало L_1: $\langle S\rangle_1$
L_2: $\langle S\rangle_2$
. . . . . .
L_k: $\langle S\rangle_k$
конец

где $\langle S\rangle_i$ — оператор, помеченный меткой L_i.

Например, для $v =$ (**истина**) v-автономное поведение схемы, изображенной на рис. 8.1, *б*, будет L2, L1, L3, L7, так что в результате «разворачивания» мы получим последовательность

начало L2: $v \leftarrow F_1(v, w)$;
L1: **если** $P_1(v)$ **то** L3: **на** L7 **иначе**
L4: **начало** ... **конец**
L3: **на** L7;
L7: СТОП(v)
конец

Покажем, как транслировать (8.11) в операторы схем класса $\mathbf{P}_A$ с требуемыми свойствами. В качестве иллюстрации транслируем приведенную выше схему в

начало $v \leftarrow F_1(v, w)$;
если $P_1(v)$ **то**
иначе начало $RT \leftarrow$ ОМЕГА; $RF \leftarrow v$;
на НАЧАЛО 1
конец;
СТОП(w);
конец;

Заметим, что лишний оператор на L7 уничтожен, а условный оператор, содержащий проверку $P_1(v)$, преобразован, как это и требовалось.

Для того чтобы транслировать (8.11), разобьем сначала множество всех операторов $\langle S\rangle_i$ на 4 класса:

(1) $v \leftarrow w$, $v \leftarrow 0$, $v \leftarrow w + 1$; $v \leftarrow F(\ldots)$, СТОП(v), «пустой» оператор;

(2) **если** $P_i(v)$ **то** $\langle S\rangle$ **иначе** $\langle S\rangle$

(3) **если** $v = m$ **то** $\langle S\rangle$ **иначе** $\langle S\rangle$, **начало** $\langle S$-список$\rangle$ **конец**, **на** l

(4) $v \leftarrow m$

Затем для каждого $i = 1, \ldots, k$ будем выполнять в зависимости от типа $\langle S\rangle_i$ одно из преобразований:

(1) заменить $L_i : \langle S\rangle_i$ на $\langle S\rangle_i$
(оставляется только непомеченный оператор);

(2) заменить $L_i : \langle S\rangle_i$ на

если $P_i(v)$ **то иначе начало** $RT \leftarrow$ ОМЕГА;
$RF \leftarrow v$;
на НАЧАЛО i
конец;

(3) вычеркнуть оператор $L_i : \langle S\rangle_i$ (он лишний, поскольку оператор, выполняемый следующим, находится в следующей строке);

(4) заменить L_i : $\langle S\rangle_i$ на $v \leftarrow$ ОМЕГА.

Мы утверждаем, что этот процесс дает оператор, удовлетворяющий (8.9) в предположении, что P_i(ОМЕГА) = **истина** для всех i. Заметим, что если P_i(ОМЕГА) ≡ **истина**, то при выполнении S_1 выполняются непомеченные операторы v-автономного поведения схемы S, которые и дают нужное значение. Когда найден предикат P, не являющийся тождественно истинным, мы выходим из ЛОКАТОРа, как это и требовалось.

(8.12) Построение S_1 в случае бесконечного v-автономного поведения. Смысл оператора S_1 в том, чтобы найти предикат с разными значениями истинности либо остановиться с тем же результатом, что и для S. Выполняемые операторы нельзя «развернуть» в последовательность, как в случае конечного v-автономного поведения, но сейчас не нужно заботиться об остановке, поскольку S не останавливается. Нужно лишь обеспечить, чтобы схема S_1 находила предикат с двумя разными значениями истинности, если такой существует (независимо от того, находит ли такой предикат схема S). Это достигается систематическим порождением *всех* возможных значений, которые в принципе *могут* вычисляться схемой S, и проверкой предикатов на этих значениях.

Точнее, рассмотрим множество всех *функциональных выражений* схемы S, определяемых следующим образом. Пусть S имеет входные переменные $x_1, \ldots, x_n$, функции $f_1^p, \ldots, f_{k_p}^p$ ранга p $(p = 1, \ldots, m)$ и предикаты $p_1^q, \ldots, p_{l_q}^q$ ранга q $(q = 1, \ldots, r)$. Множество функциональных выражений порождается продукциями

$$E \to x_1 \mid x_2 \mid \ldots \mid x_n \mid \Omega$$

$$E \to f_1^1(E) \mid \ldots \mid f_{k_1}^1(E)$$

$$\ldots$$

$$E \to f_1^m(E, \ldots, E) \mid \ldots \mid f_{k_m}^m(E, \ldots, E)$$

Это множество выражений $\{E\}$ содержит все возможные значения, на которых могут проверяться предикаты схемы S.

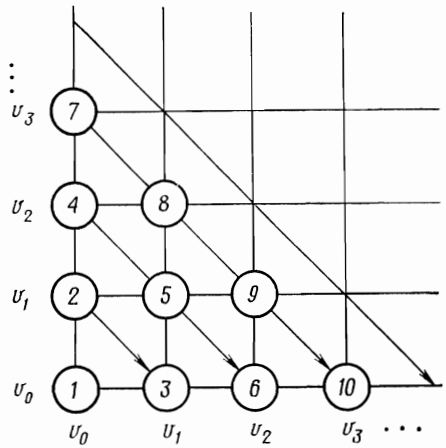
Схема S_1 просматривает все множество $\{E\}$. Вид схемы S_1 зависит от максимума m рангов функций и предикатов; S_1 для ранга $m+1$ строится индуктивно по S_1 для ранга m. Для того чтобы описать общий вид схемы S_1, опишем сначала S_1 для случая одного предиката P_1 ранга 1, одной функции ранга 2 и одного простого начального значения v_0. Напишем программу,

порождающую значения и проверяющую на них предикаты в следующем порядке:

$$v_1 = f(v_0,\ v_0)$$
$$v_2 = f(v_1,\ v_0),\ v_3 = f(v_0,\ v_1)$$
$$v_4 = f(v_2,\ v_0),\ v_5 = f(v_1,\ v_1),\ v_6 = f(v_0,\ v_2)$$
$$v_7 = f(v_3,\ v_0),\ v_8 = f(v_2,\ v_1),\ v_9 = f(v_1,\ v_2),\ v_{10} = f(v_3,\ v_0)$$
$$\ldots$$

Этот процесс проиллюстрирован на диаграмме (8.13), где строки (столбцы) указывают значения первого (второго) аргумента, а номера в точках решетки — порядок, в котором порождаются новые значения.

(8.13)



Приведенный ниже оператор для S_1 будет, очевидно, обладать нужными свойствами (здесь A хранит массив значений, K указывает, сколько значений порождено в A к текущему моменту, а I и J — счетчики для индексов значений первого и второго аргументов). Единственная трудность заключается в операторе, проверяющем I; дело в том, что такой оператор в схемах класса $\mathbf{P}_A$ не допускается.

начало $K \leftarrow 0$; $I \leftarrow 0$; $J \leftarrow 0$; $A[0] \leftarrow v_0$;
ЦИКЛ: **если** $P_1(A[K])$ **то**
иначе начало $RT \leftarrow$ОМЕГА; $RF \leftarrow A[K]$;
на НАЧАЛО1
конец;

} Проверка P_1 на новом значении и выход, если найден предикат, принимающий два разных значения истинности.

```
                K ← K + 1; A[K] ← f(A[I], A[J]);   } Порождение нового значения с использованием v_I, v_J в качестве аргументов.
                J ← J + 1;                         } Увеличение J.
SAVE ← I; I ← I ∸ 1;                               } Уменьшение I. Если I не изменяется, то мы находимся в конце диагонали (см. (8.13)) и должны начать с новой диагонали.
  если I = SAVE то
    начало I ← J; J ← 0 конец
на ЦИКЛ                                            } Проверка на новом значении.
конец
```

Чтобы избавиться от этой проверки счетчика I, мы будем пользоваться двумя дополнительными массивами. AD $[I]$ — массив «нижних» точек; AD $[I] = I \dot{-} 1$ (кроме значения AD [0], которое мы определим чуть ниже). Таким образом, оператор $I \leftarrow I \dot{-} 1$ заменяется на $I \leftarrow$ AD $[I]$.

Далее J заменяем массивом индексов AS. Если $A[I]$ берется в качестве первого аргумента функции f, то A [AS $[I]$] — ее второй аргумент. Заметим, что сначала $A[I]$ используется как первый аргумент, а вторым аргументом является A [0]. Итак, когда порождается новое значение $A[K]$, мы полагаем AS$[K] = 0$. Кроме того, во второй (третий и т. д.) раз $A[I]$ используется как первый аргумент, а в качестве второго берется A [1] (A [2] и т. д.). Это означает, что после использования $A[I]$ в качестве первого аргумента мы должны увеличить AS $[I]$ на 1. Это дает приведенную ниже программу, которая должна быть понятна, кроме, быть может, (1) оператора AD [0] ← AS [0] и (2) того, как мы начинаем новую диагональ диаграммы (8.13).

```
начало K ← 0; I ← 0; A[0] ← v_0;        } Загрузка начальных значений.
       AS[0] ← 0; AD[0] ← 0;
ЦИКЛ: если P_1(A[K]) то                  } Проверка на новом значении.
      иначе начало RT ← ОМЕГА;
           RF ← A[K]; на НАЧАЛО I
        конец;
```

$K \leftarrow K+1$; $A[K] \leftarrow f(A[I], A[\mathrm{AS}[I]])$; } Порождение нового значения.
$\mathrm{AS}[K] \leftarrow 0$; $\mathrm{AD}[K] \leftarrow K \dot{-} 1$; }

L1: $\mathrm{AS}[I] \leftarrow \mathrm{AS}[I]+1$; } Увеличение индекса для следующего повторения.

L2: $\mathrm{AD}[0] \leftarrow \mathrm{AS}[0]$; $I \leftarrow \mathrm{AD}[I]$; } Фиксация нижней нулевой точки.

на ЦИКЛ

конец

Пока мы уменьшаем I (двигаясь по диагонали диаграммы (8.13)), действия программы вполне понятны. Предположим теперь, что $I = 0$ и вычисляется $f(A[0], A[\mathrm{AS}[0]])$. Тогда в соответствии с (8.13) следующим вычисляемым значением будет $f(A[\mathrm{AS}[0]+1], A[0])$, а в I нужно заслать $\mathrm{AS}[0]+1$. Оператор, помеченный меткой L1, прибавляет единицу к $\mathrm{AS}[0]$, как и требовалось, а строка с меткой L2 засылает в I требуемое значение! В случае $I \neq 0$ оператор $\mathrm{AD}[0] \leftarrow \mathrm{AS}[0]$, по существу, ничего не делает, а в случае $I = 0$ он используется для засылки нужного значения в I.

Теперь можно обсудить построение S_1 в общем случае, предполагая бесконечность v-автономного поведения. Идея здесь та же, что и в случае одной функции ранга 2. Мы строим S_1 индукцией по максимальному рангу функций и предикатов.

Предположим, что все функции $f_1, \ldots, f_n$ и предикаты $P_1, \ldots, P_k$ имеют ранг 1, а входными переменными исходной схемы S являются $v_1, \ldots, v_l$. Выпишем тогда следующий оператор:

начало $I \leftarrow 0$;
$K \leftarrow 0$; $A[K] \leftarrow$ ОМЕГА;
$K \leftarrow K+1$; $A[K] \rightarrow v_1$;
. . . . . .
$K \leftarrow K+1$; $A[K] \leftarrow v_l$; } Загрузка начальных значений.

(8.14) ЦИКЛ: $I \leftarrow I+1$;
[проверка предикатов ранга 1 на $A[I \dot{-} 1]$];
$K \leftarrow K+1$; $A[K] \leftarrow f_1(A[I \dot{-} 1])$;
. . . . . .
$K \leftarrow K+1$; $A[K] \leftarrow f_n(A[I \dot{-} 1])$;
на ЦИКЛ

конец

Детали проверки предикатов ранга 1 мы оставляем читателю. Загрузка начальных значений засылает все входные значения в массив A. Переменная K указывает всегда, сколько значений находится в A. Эту схему можно было бы упростить, запоминая $I \dot{-} 1$ в точке программы перед оператором **на** ЦИКЛ; мы не сделали этого с целью упрощения индукции.

Если функций ранга более 1 нет, то очевидно, что описанный оператор работает так, как нужно. Все возможные значения помещаются в массив A, а I используется для перечисления его элементов при проверке предикатов на этих значениях.

Предположим, что есть оператор S_1, который «заботится» обо всех функциях и предикатах ранга $m > 0$ и ниже. Он имеет вид

начало $I \leftarrow 0$; } Загрузка начальных
S_1; ...; S_n; } значений.
ЦИКЛ: $I \leftarrow I + 1$;
. . . .
[проверка предикатов ранга m на $v_1, v_2, \ldots, v_m$]
(8.15) $K \leftarrow K + 1$; $A[K] \leftarrow f_1^m(v_1, v_2, \ldots, v_m)$;
. . . .
$K \leftarrow K + 1$; $A[K] \leftarrow f_{k_m}^m(v_1, v_2, \ldots, v_m)$;
на ЦИКЛ
конец

Предположения индукции здесь таковы: A перечисляет все возможные значения, K указывает, сколько значений заслано в A, $S_1, \ldots, S_n$ — операторы загрузки начальных значений, I изменяется только там, где это явно указано, и всякое входное значение рано или поздно попадает в массив A. Всевозможные наборы $(v_1, \ldots, v_m)$ используются в качестве аргументов функций $f_1^m, \ldots, f_{k_m}^m$. Заметим, что приведенный выше оператор для функций и предикатов ранга 1 удовлетворяет этим требованиям и имеет вид (8.15).

Предположим, кроме того, что в исходной схеме S употребляются предикаты и функции $f_p, \ldots, f_q$ ранга $m + 1$. Преобразуем (8.15) с помощью новых массивов AD, AS, $A1, \ldots, Am$ и простых переменных $I1$ и $J1$. Массивы $A1, \ldots, Am$ будут хранить всевозможные m-наборы; $J1$ — их счетчик. AD, AS и $I1$ будут использоваться точно так же, как и в ЛОКАТОРе ранга 2 для указания (на каждом шаге) значения $A[I1]$ и m-набора $A1[\mathrm{AS}[I1]], \ldots, Am[\mathrm{AS}[I1]]$. $S1$ и $S2$ — новые простые переменные.

Преобразуем (8.15) в

начало $I \leftarrow 0$; S_1; ...; S_n; } Старая загрузка начальных значений

$A1[0] \leftarrow$ ОМЕГА; ...; $Am[0] \leftarrow$ ОМЕГА; } Новая загрузка начальных значений
$\mathrm{AD}[0] \leftarrow 0$; $\mathrm{AS}[0] \leftarrow 0$;
$I1 \leftarrow 0$; $J1 \leftarrow 0$;

ЦИКЛ: $I \leftarrow I + 1$;

. . . . . .

[проверка предикатов ранга m на $v_1, \ldots, v_m$]
$K \leftarrow K + 1$; $A[K] \leftarrow f_1^m(v_1, \ldots, v_m)$;
. . . . . .
$K \leftarrow K + 1$; $A[K] \leftarrow f_{k_m}^m(v_1, \ldots, v_m)$;
} Как и раньше

$J1 \leftarrow J1 + 1$; $\mathrm{AD}[J1] \leftarrow J \dot{-} 1$; $\mathrm{AS}[J1] \leftarrow 0$;
$A1[J1] \leftarrow v_1$; ...; $Am[J1] \leftarrow v_m$;
} Засылка нового m-набора в $A1, \ldots, Am$.

$S1 \leftarrow I1$; $S2 \leftarrow \mathrm{AS}[I1]$;
} Нахождение индекса для ранга $m + 1$.

$\mathrm{AS}[I1] \leftarrow \mathrm{AS}[I1] + 1$; $\mathrm{AD}[0] \leftarrow \mathrm{AS}[0]$;
$I1 \leftarrow \mathrm{AD}[I1]$;
} Фиксация второго индекса и переход вниз по диагонали.

[проверка предикатов ранга $m + 1$ на $A[S1]$, $A1[S2]$; ..., $Am[S2]$]
} Проверка предикатов ранга $m + 1$.

$K \leftarrow K + 1$;
$A[K] \leftarrow f_p(A[S1], A1[S2], \ldots, Am[S2])$;
. . . . . .
$K \leftarrow K + 1$;
$A[K] \leftarrow f_q(A[S1], A1[S2], \ldots, Am[S2])$;
} Нахождение новых значений функций

на ЦИКЛ

конец

Заметим прежде всего, что наше преобразование не изменяет ни одного из исходных операторов и, поскольку ни одна из исходных переменных не изменяется новыми операторами, значения функций ранга m и меньше вычисляются и запоминаются точно так же, как и раньше. Этот новый оператор имеет вид (8.15), где $v_1 = A[S1]$, $v_2 = A1[S2]$, ..., $v_{m+1} = Am[S2]$.

(8.16) Из этого построения, а также из (7.1) следует $\mathbb{F}(\mathbf{P}_A, D) \equiv \mathbb{F}(\mathbf{P}_{AL}, D)$.

9. ИСПОЛЬЗОВАНИЕ ПРОСТЫХ И МНОГОМЕРНЫХ МАССИВОВ

В разд. 6 мы видели, что для всякой схемы из $\mathbf{P}_{pdsm}$ достаточно двух магазинов. Аналогичная ситуация и в $\mathbf{P}_A$.

(9.1) Теорема. *Для всякой схемы* $S \in \mathbf{P}_A$ ($\mathbf{P}_{AL}$, $\mathbf{P}_{Am}$, $\mathbf{P}_{Ae}$, ...) *существует эквивалентная схема* $S' \in \mathbf{P}_A$ ($\mathbf{P}_{AL}$, $\mathbf{P}_{Am}$, $\mathbf{P}_{Ae}$, ...), *использующая всего лишь один массив.*

Доказательство. Пусть в S используются массивы $A_0, \ldots, A_{n-1}$. Схема S' использует только один массив A, причем ячейки

$A[0], A[n], A[2n], \ldots$ представляют массив A_0,

$A[1], A[n+1], A[2n+1], \ldots$ представляют массив A_1,

. . . . . .

$A[n-1], A[2n-1], A[3n-1], \ldots$ представляют массив A_{n-1}.

Для того чтобы достичь этого, достаточно преобразовать каждый оператор $v \leftarrow w + 1$ в оператор $v \leftarrow w + n$ и $A_i[v]$ заменить на $A[v + i]$ для всех i.

Предположим, что допускается использование n-мерного массива $A[w_1, \ldots, w_n]$, $n > 1$. При этом естественно считать, что $A[w_1, \ldots, w_n]$ — та же простая переменная, что и $A[v_1, \ldots, v_n]$, тогда и только тогда, когда $w_i \ominus v_i$ для $i = 1, \ldots, n$ (в смысле определения (4.4)). Покажем, что использование многомерных массивов не увеличивает вычислительную мощь класса.

(9.2) Теорема. *Пусть* S *— схема из* $\mathbf{P}_{Ae}$, *использующая в дополнение к описанным ранее средствам один* n*-мерный массив* A, $n > 1$. *Тогда существует эквивалентная схема* $S' \in \mathbf{P}_{Ae}$, *использующая вместо* A $n+1$ *одномерных массивов* $B_0, B_1, \ldots, B_n$.

Доказательство. Кроме этих массивов S' использует переменную i. Предполагается, что эта переменная указывает, сколько различных элементов заслано в массив A. Если в схеме S переменной $A[w_1, \ldots, w_n]$ присвоено значение v, то в схеме S' для некоторого j будет

$$B_1[j] = w_1, \ldots, B_n[j] = w_n, B_0[j] = v$$

Пусть J — новая переменная, а оператор

начало $J \leftarrow 0$;
ЦИКЛ: **если** $I \ominus J$ **то**
начало $B_1[J] \leftarrow w_1$; ...; $B_n[J] \leftarrow w_n$;
$B_0[J] \leftarrow$ ОМЕГА; $I \leftarrow I + 1$;
конец
иначе **начало если** $B_1[J] = w_1$ **то**
. . . . . .
если $B_n[J] = w_n$ **то на** FOUND
иначе
. . .
иначе
$J \leftarrow J + 1$; **на** ЦИКЛ
конец
FOUND:
конец

записывается сокращенно COPY $[J, w_1, \ldots, w_n]$. Этот оператор отыскивает такой индекс J, что $B_1[J] = w_1, \ldots, B_n[J] = w_n$. Если такой индекс обнаруживается, то ячейку $A[w_1, \ldots, w_n]$ схемы S представляет ячейка $B_0[J]$ схемы S'. Если такого индекса нет, то добавляем соответствующие ячейки в $B_0, \ldots, B_n$.

Теперь для того, чтобы транслировать S в S', мы

(1) в начало схемы S поместим оператор $I \leftarrow 0$;

(2) преобразуем S так, чтобы единственными операторами, в которых фигурирует массив A, были

$$A[w_1, \ldots, w_n] \leftarrow v \quad \text{и} \quad v \leftarrow A[w_1, \ldots, w_n],$$

где w_i и v — простые переменные (очевидно, что сделать это очень просто);

(3) каждый оператор $A[w_1, \ldots, w_n] \leftarrow v$ заменим на

начало COPY $[J, w_1, \ldots, w_n]$; $B_0[J] \leftarrow v$ **конец**;

(4) каждый оператор $v \leftarrow A[w_1, \ldots, w_n]$ заменим на

начало COPY $[J, w_1, \ldots, w_n]$; $v \leftarrow B_0[J]$ **конец**.

(9.3) Теорема. *Для всякой схемы* $S \in \mathbf{P}_{AL}$ ($\mathbf{P}_{Am}$, $\mathbf{P}_{Ae}$) *с одномерными и многомерными массивами существует эквивалентная схема* $S' \in \mathbf{P}_{AL}$ ($\mathbf{P}_{Am}$, $\mathbf{P}_{Ae}$), *в которой используется всего лишь один одномерный массив.*

Доказательство. По $S \in \mathbf{P}_{AL}$ ($\mathbf{P}_{Am}$, $\mathbf{P}_{Ae}$) строим $S_1 \equiv S$, $S_1 \in \mathbf{P}_{Ae}$ (конструкция из теоремы (7.11) или леммы (7.5)). Затем с помощью конструкции, описанной в теореме (9.2), строим такую схему $S_2 \equiv S_1$ из $\mathbf{P}_{Ae}$, что в S_2 есть лишь

одномерные массивы. Далее строим $S_3 \equiv S_2$, $S_3 \in \mathbf{P}_{AL}(\mathbf{P}_{Am}, \mathbf{P}_{Ae})$, с помощью конструкций, описанных в лемме (7.6) и/или теореме (7.1). Наконец, применяем конструкцию теоремы (9.1) и строим $S' \equiv S_3$.

Все построения, проводившиеся в теореме (9.3), эффективны. Предположим теперь, что в $S \in \mathbf{P}_A$ используются многомерные массивы. Можно рассмотреть схему S как схему класса $\mathbf{P}_{Ae}$ и построить схему $S_1 \equiv S$, $S_1 \in \mathbf{P}_{Ae}$, в которой будут лишь одномерные массивы. Согласно теореме (8.16), существует $S_2 \in \mathbf{P}_A$, $S_2 \equiv S_1$, но в общем случае схему S_2 *нельзя построить эффективно.* Наконец, применим теорему (9.1) для построения такой схемы $S' \in \mathbf{P}_A$, $S' \equiv S_2$, что в ней используется только один одномерный массив. Итак, доказана

(9.4) Теорема. *Для всякой схемы $S \in \mathbf{P}_A$ с одномерными и/или многомерными массивами существует эквивалентная схема $S' \in \mathbf{P}_A$, в которой только один одномерный массив.*

Мы подозреваем, что в теореме (9.4) S' нельзя построить по S эффективно.

10. РЕЗЮМЕ

В статье устанавливается несколько интересных фактов. Выясняется, что по крайней мере на уровне схем с помощью массивов можно делать больше, чем с помощью рекурсивных процедур. Фактически мы не смогли разработать ни одного программистского средства, которое нельзя было бы выразить в $\mathbf{P}_A$ (в предположении, что функции и предикаты всюду определены), хотя эффективное построение эквивалентных схем возможно не всегда.

Некоторые вопросы, касающиеся рассматриваемых классов схем, остаются открытыми. Во-первых, мы знаем, что

$$\mathbf{P}_R \subseteq \mathbf{P}_{(1,0)} \subseteq \mathbf{P}_{(2,2)} \equiv \mathbf{P}_A$$

($\mathbf{P}_{(i,j)}$ — это класс схем с i магазинами и j маркерами). Но мы не знаем, где в точности лежат классы $\mathbf{P}_{(1,0)}$ и $\mathbf{P}_{(1,1)}$. Во-вторых, остается вопрос: $\mathbf{P}_A \equiv \mathbf{P}_{AR}$ или $\mathbf{P}_A \subset \mathbf{P}_{AR}$? Напомним, что мы моделировали рекурсию посредством массива, и здесь могут возникнуть некоторые проблемы. Мы предполагаем, что $\mathbf{P}_A \equiv \mathbf{P}_{AR}$.

Хорошо было бы выяснить, возможно ли *эффективное* построение схемы S' из $\mathbf{P}_A$, эквивалентной схеме S из $\mathbf{P}_A$, использующей многомерные массивы.

Мы предполагаем, что классы $\mathbf{P}_{AL}$, $\mathbf{P}_{Ae}$, $\mathbf{P}_{Am}$ универсальны. Нам кажется, что в $\mathbf{P}_{AeLm}$ без всяких трудностей можно реали-

зовать другие программистские средства, такие, как подстановка именем, подстановка результатом, подстановка ссылкой, сопрограммы и т. д.

ПРИЛОЖЕНИЕ

Приведем детали некоторых построений, наброски которых были даны в статье. Для удобства чтения следует возвращаться к основному тексту.

(5.1) Теорема. *По данной схеме S из $\boldsymbol{P}_R$ можно эффективно построить эквивалентную схему S' из $\boldsymbol{P}_{AL}$.*

Построение

Шаг 1 (вставка операторов инициализации). Вставить перед ⟨S-списком⟩ основного ⟨тела⟩ схемы S операторы инициализации глобальных счетчиков ТОР и АЗ:

$$\text{ТОР} \leftarrow 0; \qquad \text{АЗ} \leftarrow 0;$$

Шаг 2 (замена простых переменных на индексированные). Заменить каждую простую переменную V_i в ⟨теле⟩ каждого определения функции на $A[\text{АЗ} + i]$.

Шаг 3 (замена операторов СТОП). Заменить СТОП(v) в ⟨теле⟩ каждого определения функции на

начало $RV \leftarrow v$; **на** $A[\text{АЗ} + (p + 1)]$ **конец**

где p — число переменных, используемых данным определением функции. RV — единственная новая простая переменная.

Шаг 4. Заменить каждый заголовок функции $f(V_1, \ldots, V_{Rf})$: на Lf:, где Lf — новая метка.

Шаг 5 (замена всех вызовов функций). Для каждого вызова небазисной функции

$$v \leftarrow f(v_1, \ldots, v_{Rf})$$

в полном тексте схемы породить новую метку RL и заменить этот оператор на

начало ТОР	$\leftarrow$ ТОР $+ 1$;	(часть действия 1)
А [ТОР]	$\leftarrow$ АЗ;	(действие 2)
$A[\text{ТОР} + (p + 1)]$	$\leftarrow RL$;	(действие 3)
$A[\text{ТОР} + 1]$	$\leftarrow v_1$;	
.	.	
.	.	
.	.	

```
   A[TOP + Rf]      ← v_Rf;
   A[TOP + (Rf + 1)] ← ОМЕГА;          (действие 4)
          .              .
          .              .
          .              .
   A[TOP + p]       ← ОМЕГА:
        АЗ          ← TOP;             (действие 5)
       TOP          ← TOP + (p + 1);   (конец действия 1)
   на Lf                               (обращение к функции)
RL: АЗ ← [АЗ]; v ← RV                  место возврата
   конец
```

Здесь Lf — метка, заменившая на шаге 4 $f(v_1, \ldots, v_{Rf})$:, а p — число простых переменных, используемых в определении функции f.

(6.1) Лемма. $\mathbb{F}(\mathbf{P}_{(n,\,m)},\ D) \subseteq \mathbb{F}(\mathbf{P}_{(2,\,n+m+1)},\ D)$.

Доказательство. Пусть S — схема из $\mathbf{P}_{(n,\,m)}$ с $n \geqslant 2$ магазинами $\mathrm{PD}_1, \ldots, \mathrm{PD}_n$. Мы строим схему $S' \in \mathbf{P}_{(2,\,n+m+1)}$, эквивалентную S. Если в S используются маркеры $M_1, \ldots, M_m$, то в S' используются дополнительно маркеры $A_0, A_1, \ldots, A_n$. В S' используются магазины PD и AUX.

S' строится по S следующим образом:

(1) В начало схемы S помещаются операторы

PD ← ОМЕГА; PD ← A_0;

(2) Каждый оператор P$D \leftarrow v$ в схеме S заменяется на
начало PD ← v; PD ← A_i **конец**

(3) Порождаются две новые простые переменные $T1$ и $T2$. Для каждого оператора $v \leftarrow \mathrm{PD}_i$ порождаются новые метки LOOP и RESTORE, а сам оператор заменяется на

```
начало AUX ← ОМЕГА; AUX ← A0;      } Начальная загрузка дополнительного магазина.
LOOP: T1 ← PD; T2 ← PD;            } Выборка следующего элемента из основного магазина.
  если T1 = Ai то                  } Найдено нужное значение, оно выбирается, и осуществляется переход к восстановлению магазина.
      начало v ← T2;
           на RESTORE
      конец
```

иначе если $T1 = A_0$ **то**
начало PD $\leftarrow T2$; PD $\leftarrow T1$;
на RESTORE
конец

} Основной магазин пуст; тогда пуст и магазин PD_i в S. Переход к восстановлению.

иначе начало AUX $\leftarrow T2$; AUX $\leftarrow T1$;
на LOOP
конец;

} Значение верхушки PD не содержится в PD_i. Оно помещается в AUX, и проверяется следующее значение.

RESTORE: $T1 \leftarrow$ AUX; $T2 \leftarrow$ AUX;

} Содержимое PD восстанавливается из AUX.

если $T1 \neq A_0$ **то**
иначе начало PD $\leftarrow T2$; PD $\leftarrow T1$;
на RESTORE
конец
конец

(6.2) Лемма. $\mathbb{F}(\mathbf{P}_{(2,m)}, D) \subseteq \mathbb{F}(\mathbf{P}_{(2,2)}, D)$.

Доказательство. Пусть в S из $\mathbf{P}_{(2,m)}$ используются маркеры $M_1, \ldots, M_m$. Эквивалентная схема S' из $\mathbf{P}_{(2,2)}$ использует маркеры M_0 и M. Каждый маркер M_i в S представляется в S' с помощью $m+1$ маркеров:

$$M_i \equiv M_0, \underbrace{M, \ldots, M}_{i-1}, M_0, \underbrace{M, \ldots, M}_{m-i}.$$

Каждая простая переменная v в S представляется в S' с помощью $m+1$ переменной $v, v_1, v_2, \ldots, v_m$. Для трансляции S в S' осуществляются следующие шаги:

(1) Каждый оператор $v \leftarrow w$ заменяется на

начало $v \leftarrow w$; $v_1 \leftarrow w_1$; $\ldots$; $v_m \leftarrow w_m$ **конец**

(2) Каждый оператор $v \leftarrow M_i$ заменяется на

начало $v \leftarrow M_0$; $v_1 \leftarrow M$; $\ldots$; $v_m \leftarrow M$; $v_i \leftarrow M_0$ **конец**

(3) Для каждого оператора

если $v = M_i$ **то** S_1 **иначе** S_2

порождается новая метка L, а сам оператор заменяется на

если $v = M_0$ **то если** $v_i = M_0$
то S_1
иначе на L
иначе L: S_2

(4) Каждый оператор PD $\leftarrow v$ (где PD — магазин) заменяется на

если $v = M_0$
то начало PD $\leftarrow v$; PD $\leftarrow v_1$; ...; PD $\leftarrow v_m$ **конец**
иначе PD $\leftarrow v$

(5) Каждый оператор $v \leftarrow$ PD заменяется следующей конструкцией (усложненной из-за того, что в случае, когда магазин PD пуст, этот оператор должен быть равносилен пустому):

начало
если $v = M_0$ **то**
начало $v \leftarrow M$; $v \leftarrow$ PD;
если $v = M$ (если $v = M$, то магазин PD был пуст)
то начало $v \leftarrow M_0$;
на OVER
конец
иначе
конец
иначе $v \leftarrow$ PD;
если $v = M_0$ **то**
начало $v_1 \leftarrow$ PD; ...; $v_m \leftarrow$ PD **конец**
OVER:
конец

(6.4) Теорема. $\mathbb{F}(\mathbf{P}_R, D) \subseteq \mathbb{F}(\mathbf{P}_{(1,0)}, D)$.

Доказательство. В приведенных ниже конструкциях (6.4a) и (6.4b) показывается, что для схемы S из $\mathbf{P}_R$ с n предикатами $P_1, \ldots, P_n$ существуют P_i-имитаторы и ЛОКАТОР в $\mathbf{P}_{(1,0)}$. После этого схема S' из $\mathbf{P}_{(1,0)}$, эквивалентная S, строится с помощью техники теоремы (5.6).

(6.4a) Лемма. *По схеме S из $\mathbf{P}_R$, использующей предикат P_i, можно построить ее P_i-имитатор из $\mathbf{P}_{(1,0)}$.*

Доказательство. Предположим, не уменьшая общности, что в ⟨теле⟩ каждой из функций, а также в основном ⟨теле⟩ схемы используется p переменных $V_1, \ldots, V_p$, где $V_1, \ldots, V_{Rf}$ — формальные параметры ($Rf = 0$ для основной схемы). Для по-

строения имитатора с одним магазином PD берутся новые переменные $A_1, \ldots, A_{Rf}$ и RV:

Шаг 1. Каждый вызов небазисной функции $V \leftarrow f(v_1, \ldots, v_{Rf})$ заменяется на

начало $A_1 \leftarrow v_1; \ldots; A_{Rf} \leftarrow v_{Rf};$ (запоминаются аргументы)

$\mathrm{PD} \leftarrow V_1; \ldots; \mathrm{PD} \leftarrow V_p$ (переменные размещаются в магазине)

$\mathrm{PD} \leftarrow \mathrm{RL};$ (в магазине запоминается метка возврата)

$V_1 \leftarrow A_1; \ldots; V_{Rf} \leftarrow A_{Rf};$ (инициализируются формальные параметры)

$V_{Rf+1} \leftarrow$ ОМЕГА; $\ldots$; $V_p \leftarrow$ ОМЕГА; (фиксируются остальные локальные переменные)

на $Lf;$ (обращение к функции)

RL: $V_i \leftarrow RV;$ (нахождение значения функции)

конец

Шаг 2. Внутри определения функции каждый оператор СТОП(V_j) заменяется на:

начало $RV \leftarrow V_j;$ (вырабатываемое значение)

$\mathrm{RET} \leftarrow \mathrm{PD};$ (метка, по которой осуществляется возврат)

$V_p \leftarrow \mathrm{PD}; \ldots; V_1 \leftarrow \mathrm{PD};$ (восстановление старых значений переменных)

на RET (возврат)

конец

Шаг 3. Каждый заголовок функции «$f(V_1, \ldots, V_{Rf})$» заменяется на «Lf:».

Это дает эквивалентную схему S_1 из $\mathbf{P}_{pdsL}$ с одним магазином PD. Далее нужно воспользоваться конструкцией, аналогичной той, которая приводилась в теореме (5.3) для замены вхождений переменных-меток с помощью значений аргументов, на которых предикат P_i принимает разные значения истинности. Это предоставляется читателю.

(6.4b) Лемма. *По схеме S из* $\mathbf{P}_R$ *можно построить ее* ЛОКАТОР *из* $\mathbf{P}_{(1,\,0)}$.

Доказательство. В данной схеме S раскрываются все вызовы первого уровня. Затем выполняются шаги 1, 2, 3 леммы (6.4a). В полученной схеме (из класса $\mathbf{P}_{pdsL}$) используется один магазин. Она содержит вхождения переменных-меток только в операторах возврата. Осуществляются шаги c) и d) теоремы (5.7).

(7.3) Лемма. $\mathbb{F}(\mathbf{P}_{Am}, D) \subseteq \mathbb{F}(\mathbf{P}_{pdsm}, D)$.

Доказательство. Опишем этапы построения схемы S' из $\mathbf{P}_{pdsm}$, эквивалентной данной схеме S из $\mathbf{P}_{Am}$.

Шаг 1. Пусть схема S такова:

$$(w_1, \ldots, w_n)\text{:} \quad \langle S\text{-список}\rangle;\ \text{СТОП}(v)$$

где $w_1, \ldots, w_n$ — простые переменные. S заменяется на

$$(\text{IN}_1, \ldots, \text{IN}_n)\text{:} \quad w_1 \leftarrow \text{IN}_1;\ \ldots;\ w_n \leftarrow \text{IN}_n;\ \langle S\text{-список}\rangle;\ \text{СТОП}(v)$$

где $\text{IN}_1, \ldots, \text{IN}_n$ — новые простые переменные. Предположим далее, что функция или предикат максимального ранга имеет ранг $m \geqslant 1$. Тогда порождается m новых простых переменных $\text{ARG}_1, \ldots, \text{ARG}_m$ и новая простая переменная V_0.

Каждый оператор $v \leftarrow f(v_1, \ldots, v_{Rf})$ заменяется на

начало $V_0 \leftarrow v_1;\ \text{ARG}_1 \leftarrow V_0;$

$\vdots$

$V_0 \leftarrow v_{Rf};\ \text{ARG}_{Rf} \leftarrow V_0;$

$\text{ARG}_1 \leftarrow f(\text{ARG}_1, \ldots, \text{ARG}_{Rf});\ V_0 \leftarrow \text{ARG}_1;\ v \leftarrow V_0$

конец

Каждый оператор

если $p(v_1, \ldots, v_{Rp})$ **то** S_1 **иначе** S_2

заменяется на

начало $V_0 \leftarrow v_1;\ \text{ARG}_1 \leftarrow V_0;$

$\vdots$

$V_0 \leftarrow v_{Rp};\ \text{ARG}_{Rp} \leftarrow V_0;$

если $p(\text{ARG}_1, \ldots, \text{ARG}_{Rp})$ **то** S_1 **иначе** S_2

конец

Каждый оператор СТОП(v) заменяется на

начало $V_0 \leftarrow v;$ СТОП(V_0) **конец**

Очевидно, все эти замены дают эквивалентную схему S_1. Цель преобразований состоит в том, чтобы получить схему, в которой входные переменные, а также аргументы функций, предикатов и операторов СТОП можно было бы не рассматривать как магазинные. (Они никогда не содержат ни маркера m, ни значения из $\mathbb{N}$. В противном случае в $\mathbf{P}_{Am}$ вместо них подставляется одна звездочка $*$.) Значит, переменные ARG_i и IN_i не нужно представлять магазином в S'.

Шаг 2. Читателю предоставляется доказать самому, что схему S_1 можно далее преобразовать в эквивалентную схему S_2 из $\mathbf{P}_{Am}$, допускающую простые операторы следующих 12 типов (здесь T — одна из переменных ARG_i или IN_j, v и w — *простые* переменные, отличные от ARG_i и IN_j, а m — маркер).

(1) $T \leftarrow v$
(2) $v \leftarrow T$
(3) $v \leftarrow w$ (v и w различны)
(4) $v \leftarrow w + 1$ (v и w различны)
(5) $v \leftarrow m$
(6) $v \leftarrow A[w]$ (v и w различны)
(7) $A[v] \leftarrow w$ (v и w различны)
(8) $T \leftarrow f(\mathrm{ARG}_1, \ldots, \mathrm{ARG}_{Rf})$
(9) **если** $p(\mathrm{ARG}_1, \ldots, \mathrm{ARG}_{Rp})$ **то** S_1 **иначе** S_2
(10) **если** $v = m$ **то** S_1 **иначе** S_2
(11) **на** l
(12) $v \leftarrow 0$

Если вспомнить, что каждая переменная (за исключением ARG_i, IN_j и V_0) будет представлена магазином, то мы увидим, что операторы должны быть соответственно преобразованы.

Шаг 3 (преобразование простых переменных в магазинные). Пусть M_0 — новый маркер. Каждую простую переменную v (за исключением ARG_i, IN_j и V_0) будем рассматривать теперь как магазин. Для каждой такой переменной v перед началом схемы помещаются операторы $v \leftarrow M_0$; $v \leftarrow$ ОМЕГА устанавливающие Ω в качестве начального значения магазина. *Каждый магазин всегда будет содержать некоторое значение и всегда первым его элементом будет* M_0.

Предположим, что мы уже умеем записывать один (составной) оператор, полностью опустошающий магазин v, и другой, копирующий содержимое магазина w в пустой магазин v без изменения w. Обозначим их через

$$\mathrm{EMPTY}(v) \quad \text{и} \quad \mathrm{COPY}(w, v).$$

Покажем, как преобразовать операторы типов (1)—(5), (10) и (12); здесь ТЕМР — новая простая переменная:

(1) **начало** $T \leftarrow v$; $v \leftarrow T$ **конец**

(Напомним, что T может содержать лишь значение из D и это значение должно находиться в верхушке магазина v.)

(2) **начало** EMPTY(v); $v \leftarrow M_0$; $v \leftarrow T$ **конец**
(3) **начало** EMPTY(v); COPY(w, v) **конец**
(4) **начало** EMPTY(v); COPY(w, v);
TEMP $\leftarrow v$; $v \leftarrow$ TEMP;
если TEMP $= '*'$ **то** $v \leftarrow$ TEMP
иначе начало EMPTY(v); $v \leftarrow M_0$; TEMP $\leftarrow '*'$;
$v \leftarrow$ TEMP; $v \leftarrow$ TEMP
конец
конец

(Напомним, что $v + 1 = 0 + 1$, когда $v \notin \mathbb{N}$. Поэтому сначала мы изменяем $v \in D$.)

(5) **начало** EMPTY(v); TEMP$\leftarrow m$; $v \leftarrow M_0$; $v \leftarrow$TEMP **конец**
(10) **начало** TEMP $\leftarrow v$; $v \leftarrow$ TEMP;
если TEMP $= m$ **то** S_1 **иначе** S_2
конец
(12) **начало** EMPTY(v); $v \leftarrow M_0$; $v \leftarrow '*'$ **конец**

Шаг 4. Итак, не приведены к форме, допускаемой классом $\mathbf{P}_{pdsm}$, лишь операторы типов (6) и (7), работающие с массивом. Покажем, как представляются массивы.

Пусть $'$, $'$ — новый маркер. Каждый массив A в S моделируется в S' некоторым магазином PD. Пусть массив A таков, как на рис. 7.2, *а*. Тогда соответствующий магазин PD показан на рис. 7.2, *б*. Таким образом, *элементы массива* A разделяются в PD маркером $'$, $'$.

Отметим, что если к некоторому моменту выполнения наибольший по номеру элемент массива, которому было осуществлено присваивание, есть $A[i]$, то содержимое соответствующего магазина PD в S' будет в этот момент представлять в точности элементы $A[0], \ldots, A[i]$.

Итак, для каждого массива A, используемого в S_3 (результат шага 3), порождается магазин PD и в начало S_3 вставляются операторы PD $\leftarrow M_0$; PD $\leftarrow$ ОМЕГА;

Аналогично инициализируются три вспомогательных магазина PDT1, PDT2 и PDT3.

Оператор $v \leftarrow A[w]$ (v и w различны) транслируется в составной оператор, который

(1) помещает *в обратном порядке* содержимое PD в магазин PDT1, не изменяя содержимого PD;

(2) копирует магазин w в PDT2 (EMPTY(PDT2); COPY(w, PDT2));

(3) опустошает магазин v(EMPTY(v));

(4) удаляет из PDT1 «элементы массива» до тех пор, пока в верхушке не окажется элемент, описываемый магазином PDT2 (напомним, что элементы массива разделяются в магазине запятыми ′, ′, а PDT2 содержит последовательность из нуля или более звездочек ′*′, что указывает на число элементов массива, которые нужно удалить);

(5) помещает элемент массива, содержащийся в верхушке PDT1, в v.

Если число звездочек в PDT2 больше, чем число запятых в PDT1, то это значит, что выполняется $v \leftarrow A[w]$, где $A[w]$ не определено. В этом случае переменной v присваивается Ω.

Аналогично оператор $A[w] \leftarrow v$ (v и w различны) транслируется в составной оператор, который

(1) помещает в обратном порядке содержимое PD в магазин PDT1 и опустошает PD;

(2) копирует w в PDT2 без изменения w;

(3) копирует в обратном порядке из PDT1 в PD столько «элементов массива», сколько звездочек в PDT2;

(4) удаляет из PDT1 элемент массива, находящийся в верхушке (это PD[w]);

(5) помещает остаток содержимого PDT1 в PD (конечно, в обратном порядке).

Если на шаге 3 окажется, что в PDT2 больше звездочек, чем запятых в PDT1, то в PD помещаются дополнительные элементы массива, содержащие Ω.

Программистские детали оставляются читателю.

СПИСОК ЛИТЕРАТУРЫ

1. Hopcroft J., Ullman J., Formal languages and their relation to automata, Addison — Wesley, London, 1969.
2. Luckham D. C., Park D. M. R., Paterson M. S., On formalized computer programs, *J. Comput. System Sci.*, **4** (1970), 220—249. (Русский перевод: Лаккэм Д., Парк Д., Патерсон М., О формализованных машинных программах, «Кибернетический сборник», новая серия, вып. 12, изд-во «Мир», М., 1975.)
3. McCarthy J., Recursive functions of symbolic expressions and their computation by machine, Part 1, *Comm. ACM*, **3** (1960), 184—195.
4. Paterson M. S., Equivalence problems in a model of computation, Doctoral thesis, Cambridge Univ., Cambridge, England, 1967.
5. Paterson M. S., Hewitt C. E., Comparative schematology, Conf. Record of Project MAC Conference on Concurrent Systems and Parallel Computation, Assoc. for Computing Machinery, New York, 1970, p. 119—128. (Русский перевод: Патерсон М., Хьюитт К., Сравнительная схематология, «Кибернетический сборник», новая серия, вып. 13, изд-во «Мир», М., 1976.)
6. Strong H. R., Translating recursion equations into flow charts, *J. Comput. System Sci.*, **5** (1971), 254—285.
7. Strong H. R., High level languages of maximum power, Proc. IEEE Conf. on Switching and Automata Theory, 1971, pp. 1—4.

Алгебраические методы в распознавании образов [1])

Юлиуш Куликовский
Академия наук, Варшава, ПНР

Оглавление

ПРЕДИСЛОВИЕ

За последние годы применение электронной техники для обработки данных и реализации вычислительных процедур достигло довольно высокого уровня зрелости. Все более сложные виды исходных данных — цифровые, алфавитно-цифровые, тексты — могут автоматически вводиться в ЭВМ и обрабатываться на них. В последнее время растет внимание к обработке изображений. Информация, представленная в виде графиков или рисунков, играет важную роль в тех случаях, когда человек включается в информационную систему в качестве одного из ее элементов. Это объясняется тем обстоятельством, что зрительный тракт человека способен обрабатывать информацию в параллельном режиме и приспосабливаться к изменяющейся обстановке. Именно поэтому проблема непосредственного обмена визуальной информацией между человеком и вычислительной машиной становится весьма актуальной. Первым шагом к ее разрешению является распознавание образов с помощью специализированных автоматических устройств и электронных вычислительных машин. К изображениям, представляющим значительный прикладной интерес, относятся не только машинописные и рукописные тексты, но и чертежи, схемы электронных приборов, графики, отпечатки пальцев, метеорологические кар-

[1]) Yuliusz Kulikowski, Algebraic Methods in Pattern Recognition, Academy of Sciences — Warsaw, International Centre for Mechanical Sciences, Courses and Lecture — No. 85, Course held at the Department of Automation and Information, July 1971, Udine 1971, Springer — Verlag, Wien, New York.

ты, микрофотографии и т. д. При введении определенных допущений их можно разделить на более или менее формализованные классы, подлежащие автоматическому различению.

Нам кажется целесообразным рассматривать изображения в качестве выражений, записанных на некотором «плоскостном» языке согласно нескольким морфологическим и синтаксическим правилам. Если необходимо выбрать алгоритм распознавания, возникает задача определения набора грамматических правил, соответствующих заданным классам образов. Судя по всему, не существует ни универсального «плоскостного» языка, ни универсального алгоритма распознавания, ориентированного на машинную реализацию. Тем не менее можно считать, что при работе со сложными образами наиболее эффективным оказывается лингвистический подход. Другие методы, предусматривающие использование геометрических, статистических или функциональных моделей распознавания, могут быть использованы в качестве отдельных этапов при структурно-лингвистическом анализе сложного образа. Возникает необходимость в универсальном метаязыке, позволяющем описывать и теоретически исследовать «плоскостные» языки, которые используются при анализе конкретных случаев. Последнее, с нашей точки зрения, может быть достигнуто на основе общей теории отношений. При этом подходе любое простое или сложное изображение рассматривается как реализация обобщенного отношения, заданного на упорядоченном семействе множеств значений локальных входных сигналов. Эта концепция развивается в настоящем лекционном курсе. Мы покажем, что на множестве допустимых отношений можно определить некоторую булеву алгебру. Это означает, что известные методы минимизации булевых функций могут быть применены для минимизации длины выражения, являющегося формальным описанием образа.

Рассмотрение будет сопровождаться численными примерами. Их, однако, не следует рассматривать как рекомендации к решению практических задач.

Я с большим удовольствием исполню свой долг, выразив искреннюю признательность профессору Луиджи Собреро и профессору Анджело Марцолло за проявленную ими инициативу, которая привела к тому, что представляемый курс был включен в программу летней школы Международного центра технических наук, состоявшейся в июне 1971 года.

Удине, июнь 1971 *Ю. Куликовский*

1. ВВОДНЫЕ ЗАМЕЧАНИЯ

В данном курсе распознавание образов рассматривается как один из разделов кибернетики, посвященный изучению общих

принципов принятия решений в системах передачи информации со стационарными источниками при введении допущения о счетности множества решений. Обобщенная схема модели, которую мы будем рассматривать, приведена на рис. 1. В ее состав, как обычно, включены «источник информации», «кодирующее устройство», «канал передачи», включающий «источник шума», «декодирующее устройство» и «система принятия решений». Мы не будем, однако, вводить допущения, касающиеся физической природы сигналов и характеристик сообщений, аналогичные вышеупомянутым ограничениям — конечности множества сообщений и статистической природе сигналов. Проблема обнаружения сигнала в присутствии шума — частный случай задачи распознавания; при ее решении предполагается, что множество сооб-

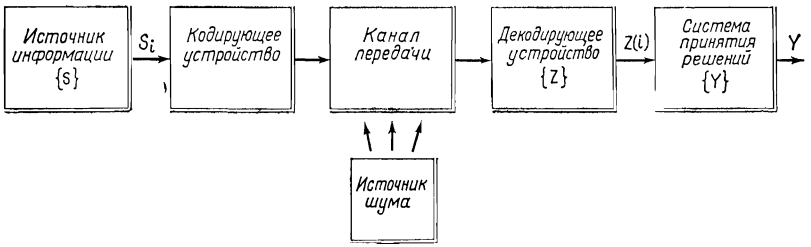


Рис. 1.

щений конечно, и сигналы можно рассматривать как реализации некоторых случайных процессов. Чтение и интерпретация машинописного текста — пример более сложной задачи распознавания образов; в этом случае статистические свойства сигналов имеют меньшее значение, но сложность задачи определяется тем обстоятельством, что множества передаваемых сообщений и множества решений бесконечны. Идентификация говорящего человека основана на анализе звуков, из которых состоит его речь; эта задача также относится к категории задач распознавания образов и соответствует случаю, когда сигналы являются случайными, причем их статистические характеристики априори неизвестны. Алгоритм принятия решений в этом случае может быть построен на основе последовательного статистического анализа и непараметрических методов. Распознавание геометрической формы и определение размеров твердого тела на ощупь — задача распознавания образов, не имеющая, пока, формальной интерпретации. И т. д.

Распознавание образов начиная с 1958 года — времени появления первой работы Фрэнка Розенблатта, посвященной концепции «перцептрона», — претерпело специфическую эволюцию. Во-первых, возникли два различных подхода к распознаванию

образов. В первом делался упор на необходимость исследования моделей нейрофизиологических механизмов, лежащих в основе процессов распознавания, свойственных головному мозгу. Идеи Ф. Розенблатта сыграли роль первотолчка в развитии этого направления исследований, которые в определенном смысле связаны и с предложенными несколько раньше Мак-Каллохом и Питтсом концепциями искусственных нейронных сетей. Второй подход ориентирован в основном не на моделирование естественных зрительного и слухового трактов, а на изучение новых принципов распознавания образов. Этот подход связан с известными методами оптимального обнаружения сигналов и оценки параметров. С другой стороны, проблема распознавания образов является и более общей, и более трудной: она заключается в определении алгоритмов распознавания и классификации машинописных и рукописных символов, графических изображений, симптомов заболеваний, микрофотографий и т. д. Таким образом, статистические модели приема сигналов представляют собой всего лишь частный случай общей постановки задачи распознавания образов.

Введем следующие обозначения: S — сообщение, передаваемое источником, $\{S\}$ — множество всех сообщений, которые могут быть переданы, Z — сигнал, который воспроизводится на выходе канала передачи, Y — решение, вырабатываемое распознающим устройством. Пространство сигналов, которые могут поступать на вход распознающего устройства, будет обозначаться через $\{Z\}$, а через $\{Y\}$ — множество возможных решений. Правило, определяющее процедуру распознавания, будет формально задаваться с помощью функции

$$Y = h(Z), \tag{1.1}$$

обеспечивающей однозначную проекцию множества входных сигналов $\{Z\}$ на множество решений $\{Y\}$. Предполагается, что множество решений $\{Y\}$ счетно; символ $\varkappa_Y$ обозначает его кардинальное число.

Пусть Σ — счетное семейство подмножества $\{S_i\} \subset \{S\}$, а $\varkappa_\Sigma$ — кардинальное число семейства Σ. Будем считать, что

$$\varkappa_\Sigma = \varkappa_Y \tag{1.2}$$

и что для семейства Σ и множества $\{Y\}$ существует некоторая взаимно однозначная проекция. Другими словами, если подмножество $\{S\}_i$ представляет «образ», то существует «решение» $Y_i \in \{Y\}$, соответствующее этому образу. Под образами будем понимать не столько отдельные сообщения, сколько некоторые классы сообщений. Так, например, если сообщением служит графическое изображение заданного на плоскости треугольника, то образ можно определить как множество представлений тре-

угольников всех возможных форм и размеров. Однако графические представления, следующие по каналу передачи (независимо от физической природы канала), обычно искажаются под воздействием каких-то внешних процессов. Поэтому процесс передачи образов формально можно представить проекцией множества $\{S\}$ в пространство $\{Z\}$, в общем случае являющейся неоднозначной и необратимой. Пусть $\{Z\}_i$ — подмножество сигналов, поступающих на вход распознающего устройства и соответствующих образу $\{S\}_i$. Будем называть задачу распознавания образов простой, если для всех $i \neq j$

$$\{Z\}_i \cap \{Z\}_j = \varnothing, \tag{1.3}$$

где символ $\varnothing$ обозначает пустое множество; в противном случае задача будет называться сложной. Обычно сложные задачи распознавания возникают при наличии случайных возмущений или аддитивного шума.

Общая теоретическая модель распознавания, которую можно считать «классической», основывается на следующей геометрической интерпретации. Допустим, что пространство $\{Z\}$ представляет собой многомерное евклидово пространство E. Пусть это пространство разбито на непересекающиеся «отсеки» C_i, ограниченные множеством гиперповерхностей заданной степени регулярности и образующие в целом конечное семейство [11]. В таком случае «функцию решения», заданную выражением (1.1), можно реализовать в виде

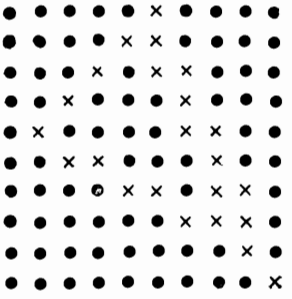
$Y = Y_i$ в том и только том случае,

$$\text{если } Z \in C_i,\ i = 1,\ 2,\ 3,\ \ldots. \tag{1.4}$$

Поскольку геометрическая модель распознавания образов является довольно общей, на начальном этапе развития теории распознавания казалось, что ее основная задача заключается в изучении методов оптимального разбиения пространства на отсеки в соответствии со свойствами образов. Хотя это, возможно, и правильно, но довольно скоро выяснилось, что на практике задача оказалась много сложней. Ситуация была прекрасно обрисована одним из участников Симпозиума в Суханово (СССР, 1967 г.). Во время дискуссии он заметил: «Практически невозможно построить поверхность, отделяющую ... воду от губки, даже если теоретически существование такой поверхности совершенно очевидно». Популярные в теории распознавания методы, в частности основанные на использовании статистических решающих функций [6, 7], потенциальных функций [1, 3], геометрических соображений [7, 8, 11], и многие другие хорошо работают только в случае сравнительно простых образов. В 1962 году была обнародована новая концепция,

предложенная Р. Нарасимханом [24] и положившая начало исследованиям в области так называемых структурных методов распознавания образов. Использование структурных методов предусматривает многоуровневый подход к анализу изображения. В первую очередь с помощью «классических» методов выясняется наличие на изображении характерных локальных признаков. Следующий уровень анализа изображения предусматривает изучение отношений между локальными признаками. Иногда на конечном этапе анализа изображения рассматриваются отношения высшего порядка, связывающие отношения низших порядков. Применение структурного метода обеспечивало получение в результате анализа изображения его описания на формальном языке вместо привычного решения, поэтому структурный подход иногда называют лингвистическим. Локальные признаки можно рассматривать в качестве элементов «словаря» языка, а отношения более высокого порядка образуют своего рода «грамматику».

Например, в геометрической форме не так просто представить множество всех возможных двоичных векторов, соответствующих всем возможным треугольникам, спроецированным на дискретную сетчатку, как это показано ниже:



Изображение можно представить с помощью двоичного вектора в 100-мерном пространстве входных сигналов. Последнее изоморфно множеству вершин единичного куба, построенного в 100-мерном евклидовом пространстве. Тем не менее едва ли можно получить описание множества вершин, соответствующего изображениям всех возможных треугольников, без учета их формы, размеров и положения на сетчатке. Можно, однако, построить тесты для обнаружения следующих локальных признаков: «отрезок прямой линии», «угол, образованный двумя отрезками прямых», «концевая точка отрезка прямой» и т. п. Таким образом, понятие «треугольник» можно отождествить с одновременным наличием следующих признаков: «три отрезка прямых линий и три образованных ими угла». Этого, од-

нако, недостаточно: необходимо ввести ограничение на количество свободных концевых точек отрезков прямых — отрезки прямых образуют треугольник в том и только том случае, когда число таких свободных точек равно нулю. Все это можно еще больше формализовать. Введем обозначения для следующих событий: A — на сетчатке воспроизведен отрезок прямой линии, B — два отрезка прямых образуют угол, C — отрезок прямой имеет свободную концевую точку. Для того чтобы разделить независимые события, каждому символу приписывается индекс. Формальное определение «треугольника» задается следующим предложением:

$$D = A_1 \wedge A_2 \wedge A_3 \wedge B_1 \wedge B_2 \wedge B_3 \wedge \neg(C_1 \vee C_2 \vee C_3 \vee C_4 \vee C_5 \vee C_6).$$

Наличие на сетчатке изображения треугольника считается доказанным в том и только том случае, если D истинно в логическом смысле.

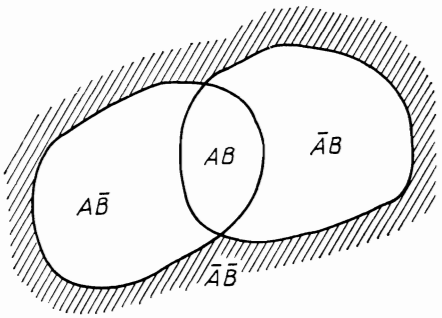


Рис. 2.

Имеется большая разница между решениями, принимаемыми на низших и высших уровнях. При работе на более высоком уровне задано не только множество возможных решений, но априори известны и отношения между решениями. Другими словами, решения, принимаемые на более высоких уровнях, не являются независимыми. Если, например, обнаружено три отрезка прямых, очевидно, что они не могут образовать больше двенадцати углов; если имеется только три угла, то очевидно, что число свободных концевых точек может быть равно либо трем, либо нулю, и т. д. Другими словами, существует множество отношений между решениями, принятие которых потенциально возможно на высших уровнях. Таким образом, процесс принятия решения больше напоминает логическое доказательство, чем просто обнаружение признаков, соответствующих высшим уровням анализа изображения.

В некоторых случаях упомянутые выше отношения можно задавать, пользуясь хорошо известным аппаратом теории графов [32]. Рассмотрим следующий пример. Структурное описание анализируемых образов зависит от наличия двух локальных признаков A и B. Множество всех входных сигналов можно разделить на области, как это показано на рис. 2. Если процесс обнаружения первого признака может привести к получению либо ответа A, либо $\bar{A}$, а соответствующие результаты для второго признака могут быть B и $\bar{B}$, то альтернативными областями решения будут AB, $\bar{A}B$, $A\bar{B}$ и $\bar{A}\bar{B}$. Влияние решений, принятых

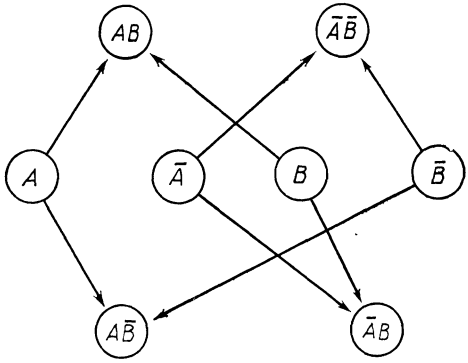


Рис. 3.

на первом уровне, на решения второго уровня можно проиллюстрировать с помощью графа, изображенного на рис. 3. Эту простую схему можно использовать для решения 14 различных задач распознавания:

1. AB, а не $\bar{A}B$ или $A\bar{B}$ или $\bar{A}\bar{B}$,
2. $\bar{A}B$, а не AB или $A\bar{B}$ или $\bar{A}\bar{B}$,
3. $A\bar{B}$, а не AB или $\bar{A}B$ или $\bar{A}\bar{B}$,
4. $\bar{A}\bar{B}$, а не AB или $\bar{A}B$ или $A\bar{B}$,
5. AB или $\bar{A}B$, а не $A\bar{B}$ и не $\bar{A}\bar{B}$,
6. AB или $A\bar{B}$, а не $\bar{A}B$ и не $\bar{A}\bar{B}$,
7. AB или $\bar{A}\bar{B}$, а не $\bar{A}B$ и не $A\bar{B}$,
8. $\bar{A}B$ или $A\bar{B}$, а не AB и не $\bar{A}\bar{B}$,
9. $\bar{A}B$ или $\bar{A}\bar{B}$, а не AB и не $A\bar{B}$,
10. $A\bar{B}$ или $\bar{A}\bar{B}$, а не AB и не $\bar{A}B$,
11. AB или $\bar{A}B$, а не $A\bar{B}$ или $\bar{A}\bar{B}$,
12. AB или $A\bar{B}$, а не $\bar{A}B$ или $\bar{A}\bar{B}$,
13. $\bar{A}B$ или $A\bar{B}$, а не AB или $\bar{A}\bar{B}$,
14. AB, а не $\bar{A}B$ и не $A\bar{B}$ и не $\bar{A}\bar{B}$.

(некоторые из этих формулировок явно можно упростить). Если решения первого уровня заданы, то направления стрелок на графе определяют систему решений второго уровня. Очевидно, существует сильная зависимость между логической структурой метода получения решений высшего уровня и внутренней структурой образов. Семейство альтернативных образов больше нельзя считать набором изолированных элементов — для их описания необходимо использовать более совершенные математические методы. С другой стороны, если мы хотим получить достаточно общую теорию, математические допущения не должны заходить слишком далеко. Следовательно, при рассмотрении общего случая задачи распознавания методы описания изображений, предусматривающие использование теории множеств, топологических и алгебраических понятий, оказываются предпочтительнее методов, основанных на использовании вероятностных, геометрических и функциональных принципов.

Следует ввести еще несколько допущений, относящихся к множеству входных сигналов $\{Z\}$. Рассмотрим конечное множество точек на плоскости, образующих прямоугольную сетчатку; при этом любая упорядоченная пара (m, n) целых чисел $m \in \langle \overline{1, M} \rangle$, $n \in \langle \overline{1, N} \rangle$ указывает адрес одного из рецепторов, образующих светочувствительную сетчатку. Всякое подмножество фоторецепторов предполагается линейно упорядоченным по адресам (в лексикографическом смысле), т. е. для любой пары адресов (m', n'), (m'', n'') предполагается выполнение неравенства

$$(m', n') < (m'', n'')$$

(читается: фоторецептор (m', n') предшествует фоторецептору (m'', n'')), если $m' < m''$ или $m' < m''$ и $n' < n''$.

Каждому фоторецептору ставится в соответствие конечное множество

$$\{Z\}_{m,n} = U_{m,n} = \{0, 1, \ldots, k-1\}, \quad k = 2, 3, 4, \ldots, \tag{1.5}$$

возможных локальных значений входных сигналов в точке с координатами (m, n). Множество всех входных сигналов можно определить как прямое произведение множеств:

$$\{Z\} = \bigtimes_{n=1}^{N} \bigtimes_{m=1}^{M} \{X\}_{m,n}, \tag{1.6}$$

в таком случае всякий входной сигнал

$$z = |z_{1,1}, z_{1,2} \ldots, z_{M,N}| \in \{Z\} \tag{1.7}$$

представляется NM-мерным вектором, компоненты которого принимают значения из конечного множества мощности k. Это число представляет количество уровней освещенности, поскольку

наш анализ в основном будет относиться к распознаванию монохроматических графических изображений. В общем случае «области решения», соответствующие определенным образам, будут задаваться с помощью подмножеств

$$C_i = \{Z\}_i \subset \{Z\}. \tag{1.8}$$

Входные сигналы $z \in C_i$ будем называть допустимыми реализациями i-го образа. Необходимо разделить допустимые и истинную реализации образа $\{S\}_i$; последняя в принципе может принадлежать любой области решения C_j, причем j не обязательно совпадает с i. Распознавание считается правильным, если $j = i$; в остальных случаях оно ошибочно. Истинную реализацию i-го образа будем обозначать через $z(i)$.

Наша задача состоит в удовлетворительном, если не оптимальном, выборе областей решения C_i. Для ее решения будут использованы алгебраические методы, по крайней мере на высших уровнях принятия решений. Основным понятием нашей теории является обобщенное отношение.

Если U' и U'' — некоторые непустые множества и $U' \times U''$ — прямое произведение этих множеств, то любое его подмножество

$$R \subset U' \times U'' \tag{1.9}$$

называется отношением, связывающим элементы $u \in U'$ и $v \in U''$. Так, например, если

$$U' = \{0, 1\},$$
$$U'' = \{0, 1\},$$

то

$$U' \times U'' = \{(0, 0), (0, 1), (1, 0), (1, 1)\},$$

и

$$R_1 = \{(0, 0), (1, 1)\},$$
$$R_2 = \{(0, 1), (1, 0)\}$$

представляют примеры отношений (R_1 называется отношением тождества, а R_2 — отрицанием). Отметим, что мы не вводили каких-либо допущений о характере множеств U' и U''. Это означает, что отношения можно определять на множествах точек, на семействах множеств, на классах функций, а также и на множествах отношений. Поэтому можно построить специальную многоуровневую «структуру» отношений, что представляет интерес с точки зрения описания сложных образов. Нам, однако, очевидно, недостаточно прямого произведения в том виде, как оно было введено выражением (1.9). Поэтому мы предлагаем более общее определение отношения.

Пусть $\langle U_1, U_2, \ldots, U_N\rangle$ — линейно упорядоченное семейство множеств и

$$U = \bigtimes_{n=1}^{N} U_n \tag{1.10}$$

— прямое произведение множеств, взятых в соответствующем порядке. Всякое подмножество

$$R \subset U \tag{1.11}$$

будем называть (обобщенным) отношением, связывающим элементы множеств $U_1, U_2, \ldots,$ и множества U_n. В таком случае область решения представляется обобщенным отношением между сигналами, одновременно присутствующими на фотографиях сетчатки. К нашей концепции, однако, в первую очередь следует отнестись как к формальной. Так, например, если $U_1, U_2, \ldots, U_6$ — множества всех натуральных чисел, некоторое отношение R можно определить как множество всех последовательностей целых чисел $\langle u_1, u_2, \ldots, u_6\rangle$, таких, что

$$v = \frac{u_1}{u_2} u_3 + \frac{u_4 - u_5}{u_6}$$

есть целое число.

Отношение R будем называть пустым, если R — пустое подмножество множества U, и тривиальным, если оно совпадает с множеством U. Пустое отношение будем обозначать символом θ. Реализацией отношения R назовем всякое упорядоченное множество значений $\langle u_1, u_2, \ldots, u_N\rangle$, принадлежащих $U_1, U_2, \ldots, U_N$ соответственно, такое, что

$$u = \langle u_1, u_2, \ldots, u_N\rangle \in R. \tag{1.12}$$

2. АЛГЕБРА ОТНОШЕНИЙ

Рассмотрим конечное семейство множеств $U_1, U_2, \ldots, U_N$, и пусть R_1 и R_2 — произвольные отношения, заданные на прямом произведении множеств U, которое определено выражением (1.10). Поскольку отношения определены как некоторые подмножества множества U, их можно комбинировать, пользуясь обычными алгебраическими операциями над множествами. В результате можно дать следующие определения:

а) дизъюнкция отношений

$$R = R_1 \cup R_2 \tag{2.1}$$

— отношение, выполняющееся на всех реализациях, для которых справедливо хотя бы одно из отношений R_1, R_2;

б) конъюнкция отношений

$$R = R_1 \cap R_2 \tag{2.2}$$

— отношение, выполняющееся на всех реализациях, для которых справедливы оба отношения R_1 и R_2;

в) асимметрическая разность отношений

$$R = R_1 \dot{-} R_2 \tag{2.3}$$

— отношение, выполняющееся на всех реализациях, для которых справедливо отношение R_1, и одновременно не выполняющееся на всех реализациях, для которых справедливо отношение R_2; и т. д.

Приведем пример, который послужит иллюстрацией введенных понятий. Пусть U_1, U_2, ..., U_9 представляют множество состояний участка сетчатки, имеющего следующую форму:

$$\begin{matrix}\langle U_1 & U_2 & U_3 \\ U_4 & U_5 & U_6 \\ U_7 & U_8 & U_9\rangle\end{matrix}$$

Введем ряд отношений:

а) «горизонтальная линия проходит через верхнюю половину участка» —

$$R_a = \{\langle 111\ 000\ 000\rangle,\ \langle 000\ 111\ 000\rangle\},$$

б) «горизонтальная линия проходит через нижнюю половину участка» —

$$R_b = \{\langle 000\ 111\ 000\rangle,\ \langle 000\ 000\ 111\rangle\};$$

используя введенные отношения, можно определить новые:

в) «горизонтальная линия проходит через участок» —

$$R_c = R_a \cup R_b = \{\langle 111\ 000\ 000\rangle,\ \langle 000\ 111\ 000\rangle,\ \langle 000\ 000\ 111\rangle\},$$

г) «горизонтальная линия проходит через центр участка» —

$$R_d = R_a \cap R_b = \{\langle 000\ 111\ 000\rangle\},$$

д) «горизонтальная линия проходит по верхней границе участка» —

$$R_e = R_a \dot{-} R_b = \{\langle 111\ 000\ 000\rangle\},$$

и т. д. В принципе на прямом произведении N множеств, каждое из которых включает k элементов, можно определить 2^{k^N} различных отношений. Однако для того, чтобы обеспечить эффективное распознавание анализируемых образов, совершенно необязательно использовать все возможные отношения.

Отношение R_a называется подотношением отношения R_b, если

$$R_a \cup R_b = R_b \quad \text{и} \quad R_a \cap R_b = R_a. \tag{2.4}$$

Отношения R_a и R_b называются взаимно не пересекающимися, если

$$R_a \cap R_b = \theta. \tag{2.5}$$

До сих пор речь шла об алгебраических операциях над отношениями, заданными на одном и том же семействе множеств. Целесообразно, очевидно, предусмотреть нечто подобное для случая, когда отношения заданы на различных семействах, например на разных участках сетчатки. Рассмотрим семейство множеств $\{U_1, U_2, \ldots, U_N\}$. Впредь при рассмотрении подсемейства множеств $\{U_{\nu_1}, \ldots, U_{\nu_p}\}$ будем считать, что линейный порядок, введенный в полном семействе множеств, сохраняется в подсемействе. Рассмотрим следующие два подсемейства множеств: $\{U_{\nu_1}, \ldots, U_{\nu_p}\}$ и $\{U_{\mu_1}, \ldots, U_{\mu_q}\}$. Оба подсемейства являются линейно упорядоченными в том смысле, что

$$\nu_1 < \nu_2 < \ldots < \nu_p < N$$

и

$$\mu_1 < \mu_2 < \ldots < \mu_q < N,$$

если множества $U_1, U_2, \ldots, U_N$ упорядочены в соответствии со своими индексами. Аналогичным образом будет предполагаться сохранение порядка при рассмотрении дизъюнкции, конъюнкции и разности подсемейств. Обозначим рассматриваемые линейно упорядоченные подсемейства символами $\langle U' \rangle$ и $\langle U'' \rangle$ соответственно, а обозначения R^{I} и R^{II} используем для неких отношений, заданных на подсемействах $\langle U^{\mathrm{I}} \rangle$ и $\langle U^{\mathrm{II}} \rangle$.

Пусть $\langle U^{\mathrm{III}} \rangle \subset \langle U^{\mathrm{I}} \rangle$ — подсемейство множеств, такое, что если $u^{\mathrm{I}} \in R^{\mathrm{I}}$ — последовательность элементов, на которых выполняется отношение R^{I}, то $u^{\mathrm{III}} \subset u^{\mathrm{I}}$ является подпоследовательностью элементов последовательности u^{I}, принадлежащих множествам, которые входят в подсемейство $\langle U^{\mathrm{III}} \rangle$. Множество всех подпоследовательностей u^{III}, удовлетворяющих такому условию, обозначим через R^{III}. Очевидно, что R^{III} можно рассматривать как новое отношение, заданное на подсемействе $\langle U^{\mathrm{III}} \rangle$ и зависящее в определенном смысле от отношения R^{I}. Отношение R^{III} будем называть проекцией отношения R^{I} в подсемейство $\langle U^{\mathrm{III}} \rangle$:

$$R^{\mathrm{III}} = R_{\langle U^{\mathrm{III}} \rangle}. \tag{2.6}$$

Рассмотрим пример, иллюстрирующий введенные понятия. Если

$$R_c = \{\langle 111\ 000\ 000 \rangle, \langle 000\ 111\ 000 \rangle, \langle 000\ 000\ 111 \rangle\}$$

— отношение, заданное на семействе двоичных множеств $\langle U_1, U_2, \ldots, U_9 \rangle$, и $\langle U_2, U_3, U_5, U_6, U_8, U_9 \rangle$ — подсемейство

множеств этого семейства, то отношение

$$R = \{\langle 11\ 00\ 00\rangle, \langle 00\ 11\ 00\rangle, \langle 00\ 11\ 00\rangle\}$$

можно считать проекцией отношения R_c («горизонтальная линия проходит через участок размера 3×3») в подучасток сетчатки размера 3×2, имеющий вид

$$\begin{array}{ll} . & \langle U_2 \quad U_3 \\ . & U_5 \quad U_6 \\ . & U_8 \quad U_9\rangle \end{array}$$

Ясно, что отношение R представляет событие «горизонтальная линия проходит через подучасток».

Рассмотрим подсемейство $\langle U^{\mathrm{I}}\rangle$ семейства множеств $\langle U\rangle$. Пусть R^{I} и R — некоторые отношения, заданные на подсемействе $\langle U^{\mathrm{I}}\rangle$ и семействе $\langle U\rangle$ соответственно. Рассмотрим все реализации отношения R, такие, что их проекции на семейство $\langle U\rangle$ удовлетворяют отношению R^{I}. Множество реализаций R^{II}, удовлетворяющих этому условию, есть подотношение

$$R^{\mathrm{II}} \subset R, \tag{2.7}$$

а проекция подотношения R^{II} в подсемейство $\langle U^{\mathrm{I}}\rangle$ есть подотношение

$$R^{\mathrm{II}}_{\langle U^{\mathrm{I}}\rangle} = R. \tag{2.8}$$

Подотношение R^{II} назовем условным отношением R для заданного отношения R^{I} и используем для него следующее обозначение:

$$R^{\mathrm{II}} = R\,(R^{\mathrm{I}}). \tag{2.9}$$

Рассмотрим пример. Заданы те же, что и в предыдущем примере, семейства множеств:

$$\langle U\rangle = \langle U_1, U_2, U_3, U_4, U_5, U_6, U_7, U_8, U_9\rangle,$$
$$\langle U'\rangle = \langle U_2, U_3, U_5, U_6, U_8, U_9\rangle \subset \langle U\rangle.$$

Определены следующие отношения:

а) «горизонтальная линия проходит через участок» —

$$R = \{\langle 111\ 000\ 000\rangle, \langle 000\ 111\ 000\rangle, \langle 000\ 000\ 111\rangle\},$$

б) «линия проходит через элемент сетчатки № 6» —

$$R^{\mathrm{I}} = \{\langle 01\ 01\ 01\rangle, \langle 10\ 01\ 00\rangle, \langle 00\ 11\ 00\rangle, \langle 00\ 01\ 10\rangle\}.$$

Условное отношение «линия проходит через весь участок при условии, что она проходит через элемент сетчатки № 6» имеет следующий вид:

$$R^{\mathrm{II}} = R\,(R^{\mathrm{I}}) = \{\langle 000\ 111\ 000\rangle\}.$$

Понятие условного отношения легко поддается обобщению. Рассмотрим два семейства множеств $\langle U^{\mathrm{I}}\rangle$ и $\langle U^{\mathrm{II}}\rangle$, их конъюнкцию и дизъюнкцию:

$$\langle U^{\mathrm{III}}\rangle = \langle U^{\mathrm{I}}\rangle \cap \langle U^{\mathrm{II}}\rangle, \tag{2.10}$$

$$\langle U^{\mathrm{IV}}\rangle = \langle U^{\mathrm{I}}\rangle \cup \langle U^{\mathrm{II}}\rangle; \tag{2.11}$$

будем считать, что семейство множеств $\langle U^{\mathrm{III}}\rangle$ непустое. Пусть R^{I} и R^{II} — отношения, заданные на семействах множеств $\langle U^{\mathrm{I}}\rangle$ и $\langle U^{\mathrm{II}}\rangle$ соответственно. Рассмотрим проекции $R^{\mathrm{I}}_{\langle U\rangle} \ldots$ и $R^{\mathrm{II}}_{\langle U\rangle} \ldots$ и их конъюнкцию

$$R^{\mathrm{V}} = R^{\mathrm{I}}_{\langle U\rangle} \ldots \cap R^{\mathrm{II}}_{\langle U\rangle} \ldots . \tag{2.12}$$

Отношение R^{V} содержит все такие подпоследовательности элементов, принадлежащих множествам $\langle U^{\mathrm{III}}\rangle$, что любая из этих подпоследовательностей представляет собой общую часть двух последовательностей, соответственно удовлетворяющих отношениям R^{I} и R^{II}. Объединение последних двух последовательностей приводит к получению последовательности, принадлежащей семейству множеств $\langle U^{\mathrm{IV}}\rangle$. Обозначим множество всех объединенных последовательностей через R; его можно рассматривать как новое отношение, заданное на семействе множеств $\langle U^{\mathrm{IV}}\rangle$ и связанное определенным образом с отношениями R^{I} и R^{II}. Это отношение мы будем называть сверткой отношений:

$$R = R^{\mathrm{I}} * R^{\mathrm{II}}. \tag{2.13}$$

Теперь стало очевидно, что условное отношение $R^{\mathrm{I}}(R^{\mathrm{II}})$ представляет собой частный случай свертки $R^{\mathrm{I}} * R^{\mathrm{II}}$, соответствующий условию $\langle U^{\mathrm{II}}\rangle \subset \langle U^{\mathrm{I}}\rangle$. Понятие свертки иллюстрируется следующим примером.

Рассмотрим участок сетчатки, имеющий следующую форму:

$$\begin{array}{cccc} \langle U_1 & U_2 & U_3 & \\ U_4 & U_5 & U_6 & U_7 \\ U_8 & U_9 & U_{10} & U_{11} \\ & U_{12} & U_{13} & U_{14}\rangle \end{array}$$

Зададим два семейства множеств:

$$\langle U^{\mathrm{I}}\rangle = \langle U_1, U_2, U_3, U_4, U_5, U_6, U_8, U_9, U_{10}\rangle,$$
$$\langle U^{\mathrm{II}}\rangle = \langle U_5, U_6, U_7, U_9, U_{10}, U_{11}, U_{12}, U_{13}, U_{14}\rangle,$$

и определим их конъюнкцию:

$$\langle U^{\mathrm{III}}\rangle = \langle U^{\mathrm{I}}\rangle \cap \langle U^{\mathrm{II}}\rangle = \langle U_5, U_6, U_9, U_{10}\rangle.$$

На семействах множеств $\langle U^{\mathrm{I}}\rangle$ и $\langle U^{\mathrm{II}}\rangle$ зададим отношения R^{I} и R^{II} «линия проходит через центр участка»:

$$R^{\mathrm{I}} = \{\langle 000\ 111\ 000\rangle, \langle 010\ 010\ 010\rangle, \langle 100\ 010\ 001\rangle, \langle 001\ 010\ 100\rangle\},$$

$$R^{\mathrm{II}} = \{\langle 000\ 111\ 000\rangle, \langle 010\ 010\ 010\rangle, \langle 100\ 010\ 001\rangle, \langle 001\ 010\ 100\rangle\}.$$

Следует иметь в виду, что отношения R^{I} и R^{II} определены на различных участках сетчатки и поэтому не идентичны, хотя соответствующие формулы совпадают. Свертка этих отношений имеет вид

$$R = R^{\mathrm{I}} * R^{\mathrm{II}} = \{\langle 100\ 0100\ 0010\ 001\rangle\}$$

и представляет прямую линию, проходящую через элементы сетчатки с номерами 1, 5, 10 и 14.

Очевидно также, что при $\langle U^{\mathrm{I}}\rangle = \langle U^{\mathrm{II}}\rangle$ справедливо равенство

$$R^{\mathrm{I}} * R^{\mathrm{II}} = R^{\mathrm{I}} \cap R^{\mathrm{II}}, \tag{2.14}$$

и, следовательно, свертку отношений можно рассматривать как обобщение конъюнкции отношений. Возникает вопрос: можно ли определить на отношениях новую операцию? Пусть R^{I} и R^{II} — некоторые отношения, заданные на семействах множеств $\langle U^{\mathrm{I}}\rangle$ и $\langle U^{\mathrm{II}}\rangle$ соответственно. Рассмотрим следующие подсемейства множеств:

$$\langle U^{\mathrm{III}}\rangle = \langle U^{\mathrm{I}}\rangle \dot{-} \langle U^{\mathrm{II}}\rangle, \tag{2.15}$$

$$\langle U^{\mathrm{IV}}\rangle = \langle U^{\mathrm{II}}\rangle \dot{-} \langle U^{\mathrm{I}}\rangle. \tag{2.16}$$

Обозначим через R^{III} и R^{IV} прямые произведения множеств, соответственно принадлежащих подсемействам $\langle U^{\mathrm{III}}\rangle$ и $\langle U^{\mathrm{IV}}\rangle$. Рассмотрим множество всех последовательностей элементов, принадлежащих семейству множеств $\langle U^{\mathrm{I}}\rangle \cup \langle U^{\mathrm{II}}\rangle$, которые построены одним из следующих способов:

1° реализация отношения R^{I} объединяется с реализацией отношения R^{IV};

2° реализация отношения R^{III} объединяется с реализацией отношения R^{II}.

Дизъюнкцию полученных такими способами множеств последовательностей можно рассматривать как новое отношение R, определенное на $\langle U^{\mathrm{I}}\rangle \cup \langle U^{\mathrm{II}}\rangle$. Это отношение будем называть прямым произведением отношений R^{I} и R^{II}:

$$R = R^{\mathrm{I}} \times R^{\mathrm{II}}. \tag{2.17}$$

Рассмотрим пример. Задано семейство множеств двоичных элементов и отношения R^{I} и R^{II}, аналогичные рассматривав-

шимся в примере, посвященном раскрытию понятия свертки. Разности подсемейств множеств имеют следующий вид:

$$\langle U^{III}\rangle = \langle U^{I}\rangle \dot{-} \langle U^{II}\rangle = \langle U_1, U_2, U_3, U_4, U_8\rangle,$$
$$\langle U^{IV}\rangle = \langle U^{II}\rangle \dot{-} \langle U^{I}\rangle = \langle U_7, U_{10}, U_{12}, U_{13}, U_{14}\rangle.$$

Очевидно, что число возможных элементов прямого произведения множеств, принадлежащих разности $\langle U^{III}\rangle$, равно 2^5; столько же элементов содержит прямое произведение множеств, принадлежащих разности $\langle U^{IV}\rangle$. Объединяя все возможные реализации отношения R^I с реализациями отношения R^{IV}, а также реализации отношения R^{III} с реализациями отношения R^{II}, получаем:

$$\begin{aligned} R^I \times R^{II} = \{&\langle 000\ 111.\ 000.\ \ldots\rangle, \langle 010\ 010.\ 010.\ \ldots\rangle,\\ &\langle 100\ 010.\ 001.\ \ldots\rangle, \langle 001\ 010.\ 100.\ \ldots\rangle,\\ &\langle \ldots\ .000\ .111\ 000\rangle, \langle \ldots\ .010\ .010\ 010\rangle,\\ &\langle \ldots\ .100\ .010\ 001\rangle, \langle \ldots\ .001\ .010\ 100\rangle\} \end{aligned}$$

(в это выражение вместо точек могут быть подставлены все возможные наборы из нулей и единиц). Итак, прямое произведение отношений $R^I \times R^{II}$ включает в данном случае $8 \cdot 2^5 = 256$ реализаций, а общее число различных двоичных наборов, которые можно получить на дизъюнкции множеств $\langle U^I\rangle \cup \langle U^{II}\rangle$, равно $2^{14} = 16\,384$.

Таким образом, свертка отношений представляет собой максимальное множество реализаций, одновременно удовлетворяющих обоим отношениям R^I и R^{II}, а их прямое произведение — минимальное множество реализаций, соответствующих хотя бы одному элементу реализаций R^I и/или R^{II}.

Очевидно, что в общем случае

$$R^I * R^{II} \subset R^I \times R^{II}, \tag{2.18}$$

и прямое произведение отношений тождественно дизъюнкции отношений, если $\langle U^I\rangle = \langle U^{II}\rangle$, за исключением случая, когда один из элементов отношения R^I или R^{II} является пустым отношением; в последнем случае прямое произведение не определено. Однако для того, чтобы довести до конца аналогию между дизъюнкцией и прямым произведением отношений, можно принять, что

$$R \times \theta = \theta \times R = R. \tag{2.19}$$

С другой стороны, свертка отношений является пустой, если условия, соответствующие отношениям R^I и R^{II}, не «пересекаются». Это означает, что некоторые из отношений пусты и/или ни одна пара их реализаций не имеет общих элементов, при-

надлежащих непустой конъюнкции семейств множеств, на которых отношения заданы. В этом случае отношения называются взаимоисключающими.

Как свертка, так и прямое произведение отношений коммутативны, что является следствием сохранения линейного порядка на семействах множеств. Обозначим через $Q_{\langle U\rangle}$ тривиальное отношение, заданное на некотором линейно упорядоченном семействе множеств $\langle U\rangle$. Всякое отношение R, заданное на подсемействе $\langle U'\rangle \subset \langle U\rangle$, может быть легко распространено на все семейство $\langle U\rangle$, если вместо отношения R воспользоваться отношением $Q_{\langle U^{II}\rangle} \times R$, где $\langle U^{II}\rangle = \langle U\rangle \dot{-} \langle U^{I}\rangle$. Следовательно, отношение $\bar{R}$, дополнительное к R, заданному на семействе $\langle U^{I}\rangle$, можно определить как

$$\bar{R} = Q_{\langle U^{I}\rangle} \dot{-} R; \tag{2.20}$$

это выражение, однако, эквивалентно отношению

$$Q_{\langle U^{II}\rangle} \times (Q_{\langle U^{I}\rangle} \dot{-} R) = Q_{\langle U\rangle} \dot{-} (Q_{\langle U^{II}\rangle} \times R) \equiv Q_{\langle U\rangle} \dot{-} R. \tag{2.21}$$

Отметим, наконец, что, обозначив символом $A(R)$ высказывание «существует реализация, на которой справедливо отношение R^{II}», по определению получим следующее:

$$A(R^{I} \times R^{II}) \Leftrightarrow A(R^{I}) \vee A(R^{II}), \tag{2.22}$$

$$A(R^{I} * R^{II}) \Leftrightarrow A(R^{I}) \wedge A(R^{II}). \tag{2.23}$$

Итак, вполне очевидно, что свертка и прямое произведение отношений обладают формальными свойствами, аналогичными свойствам логической конъюнкции и строгой дизъюнкции. В частности, для отношений справедливы законы де Моргана:

$$\overline{R^{I} \times R^{II}} = \overline{R^{I}} * \overline{R^{II}}, \tag{2.24}$$

$$\overline{R^{I} * R^{II}} = \overline{R^{I}} \times \overline{R^{II}}. \tag{2.25}$$

Можно также доказать ассоциативность обеих операций:

$$R^{I} \times (R^{II} \times R^{III}) = (R^{I} \times R^{II}) \times R^{III} = R^{I} \times R^{II} \times R^{III}, \tag{2.26а}$$

$$R^{I} * (R^{II} * R^{III}) = (R^{I} * R^{II}) * R^{III} = R^{I} * R^{II} * R^{III}, \tag{2.26б}$$

а также дистрибутивность каждой операции относительно другой:

$$R^{I} \times (R^{II} * R^{III}) = (R^{I} \times R^{II}) * (R^{I} \times R^{III}), \tag{2.27а}$$

$$R^{I} * (R^{II} \times R^{III}) = (R^{I} * R^{II}) \times (R^{I} * R^{III}). \tag{2.27б}$$

Другими словами, упорядоченный набор из шести элементов $\langle\langle U\rangle, Q_{\langle U\rangle}, \theta, *, \times, \bar{}\,\rangle$ образует булеву алгебру, в которой ну-

стое отношение является нулевым элементом и тривиальное отношение $Q_{\langle U\rangle}$ — единичным.

Последнее утверждение может оказаться очень важным с практической точки зрения. Оно означает, что общепринятые методы минимизации булевых функций и конструктивные методы синтеза логических систем могут быть использованы для построения алгоритмов интерпретации изображения и распознавания образов.

Хотя основные из введенных нами понятий иллюстрировались на двоичных множествах, собственно алгебра отношений не содержит каких-либо ограничений, связанных с природой множеств. Рассмотрим, например, линейно упорядоченное множество случайных переменных $\langle X_1, X_2, \ldots, X_N\rangle$. Зададим следующее множество отношений $R_2, R_3, \ldots, R_N$: отношение R_n содержит все последовательности случайных переменных $\langle X_{i_1}, X_{i_2}, \ldots, X_{i_n}\rangle$, являющихся в данном множестве статистически зависимыми. Возникает вопрос о характере отношений (высшего порядка) между отношениями R_n. Прежде всего можно определить отношение «пересечения» P как последовательность отношений $\langle R_{n_1}, R_{n_2}, \ldots, R_{n_k}\rangle$, такую, что существует хотя бы одна случайная переменная X_i, удовлетворяющая условиям всех входящих в эту последовательность отношений. Это не означает, однако, наличия статистической зависимости между остальными аргументами отношений $R_{n_1}, R_{n_2}, \ldots, R_{n_k}$, поскольку можно легко доказать, что статистическая зависимость случайных переменных не является транзитивным свойством. Тем не менее если случайные переменные $X_{i_1}, X_{i_2}, \ldots, X_{i_n}$ попарно статистически зависимы, т. е. любая случайная переменная находится в состоянии статистической зависимости с любой из всех остальных случайных переменных, то все такие случайные переменные в общем статистически зависимы. Следовательно, над отношениями R_n может быть задано отношение, отличное от P-отношения: это T-отношение можно определить как отношение, включающее все такие последовательности отношений $\langle R_{n_1}, R_{n_2}, \ldots, R_{n_m}\rangle$, что отношение R_{n_m} выполняется при условии выполнения всех предшествующих отношений $R_{n_1}, R_{n_2}, \ldots, R_{n_{m-1}}$. В таком случае возникает вопрос о характере отношения «высшего порядка», связывающего отношения P и T. Эти отношения, в частности, пересекаются в обсуждавшемся смысле. Изучение отношений подобного типа между случайными переменными может оказаться интересным с точки зрения анализа структур информационных систем. Последнее, однако, выходит за пределы непосредственных задач распознавания образов.

3. СТРУКТУРНЫЙ ПОДХОД К РАСПОЗНАВАНИЮ ОБРАЗОВ

Автоматическая классификация сложных изображений предусматривает несколько уровней обработки. Важнейшими из них являются следующие:

а) выравнивание, центрирование растра, изменение масштаба, коррекция яркости и т. д.;

б) фильтрация шума, устранение дефектов, утоньшение линий, выделение контуров и т. д.;

в) выделение локальных признаков;

г) выделение и интерпретация признаков высших порядков.

Эти уровни обработки изображения не следует считать строго изолированными друг от друга. Порой решения, принимаемые на высшем уровне, требуют возвращения назад, к решениям более низкого уровня.

Алгоритмы принятия решений, относящиеся к низшим уровням вплоть до выделения локальных признаков, могут строиться на основе «классических» методов распознавания. На этой стадии обработки изображения могут использоваться и детерминированные, и статистические, и обучающиеся алгоритмы. Трудности, однако, возникают обычно в связи с проблемой выбора системы основных признаков, образующей «словарь» языка описания образов. Обычно эта проблема разрешается эвристически, однако некоторые авторы предпринимали попытки решать ее с помощью более совершенных методов [6, 13]. Процедура оптимизации вполне осуществима, если определен критерий оптимальности. Последнее, однако, само по себе является трудной задачей.

Задачу распознавания образов можно рассматривать как задачу уменьшения информационной избыточности. Так, например, если сетчатка состоит из N элементов, которые могут находиться в одном из двух возможных состояний («черное — белое»), и множество конечных решений содержит допустимые элементы (образы, подлежащие распознаванию), то коэффициент уменьшения избыточности определяется следующим образом:

$$c = \frac{N}{\log_2 x}; \tag{3.1}$$

в большинстве практических случаев его величина больше единицы. Процесс снижения информационной избыточности распределен между уровнями обработки изображения. Значительное уменьшение избыточности на одном уровне приводит в то же время к возникновению серьезных практических затруднений. Количество необходимых уровней обработки изображения d можно определить с помощью неравенства

$$d \log \gamma \geqslant \log c, \tag{3.2}$$

где γ — максимальное значение коэффициента уменьшения информационной избыточности, допустимое для одного уровня.

Количество классов изображений x', которые необходимо разделить на первом уровне обработки изображения, задается формулой

$$\log_2 x' = \frac{N}{\gamma}, \tag{3.3}$$

при этом проблема выбора системы основных признаков сводится к задаче оптимального разделения исходного набора изображений на x' классов на первом уровне распознавания. Не следует, однако, смешивать определенное таким способом число классов x' с количеством локальных признаков, которые необходимо выделить, поскольку местонахождение (адрес) локального признака позволяет получить некоторое количество дополнительной информации о распознаваемом образе. Так, например, если локальные признаки отыскиваются на подучастке изображения размера $M' \times N'$, где $M' < M$, $N' < N$, M и N — размеры прямоугольной сетчатки, и если число шагов при последовательном перемещении «окна» по сетчатке равно b, то результаты анализа изображения на первом уровне можно закодировать с помощью b символов алфавита, число элементов которого равно общему числу допустимых локальных решений. Другими словами, требуемое число локальных признаков равно $(x')^{1/b}$. Теперь очевидно, что в принципе один и тот же результат можно получить с помощью большого набора достаточно сложных локальных признаков, заданных на подучастках большого размера, и с помощью весьма ограниченного набора простых локальных признаков, заданных на небольших подучастках.

Разница между детерминированными и статистическими процедурами выделения локальных признаков изображения очень относительна: детерминированные процедуры можно получить как асимптотический вариант статистических, соответствующий случаю малого разброса дисперсий. В частности, основой распространенного метода выделения признаков, предусматривающего оценку расстояния между локальным сигналом и эталонами, является статистический метод максимального правдоподобия. В задачах распознавания обычно используются следующие метрики. Пусть

$$z^* = \langle z_1^*, z_2^*, \ldots, z_n^* \rangle \tag{3.4}$$

— эталонный сигнал, представляющий подлежащий обнаружению локальный признак, и

$$z = \langle z_1, z_2, \ldots, z_n \rangle \tag{3.5}$$

— локальная подреализация реального входного сигнала, поступившая на обработку. Чаще всего используется расстояние, предложенное Р. У. Хэммингом:

$$\rho(z^*, z) = \sum_{\nu=1}^{N} (x_\nu^* - x_\nu). \tag{3.6}$$

В случае когда компоненты вектора являются многоградационными, предпочтительнее пользоваться евклидовым расстоянием:

$$\rho(z^*, z) = \left[\sum_{\nu=1}^{N} (z_\nu^* - z)^2\right]^{1/2}; \tag{3.7}$$

в случае двоичных векторов евклидово и хэммингово расстояния эквивалентны.

Еще одна метрика была предложена М. А. Айзерманом, Э. М. Браверманом и Л. И. Розоноэром [1, 3]. Потенциальная функция определяется как

$$K(z^*, z) = \sum_{\nu=1}^{N} \lambda_\nu^2 \varphi_\nu(x^*)\, \varphi_\nu(x), \tag{3.8}$$

где $\{\varphi_\nu(x)\}$ — множество скалярных функций, заданных на множестве векторных аргументов, а λ_ν^2 — некоторые действительные коэффициенты. При этом расстояние имеет следующий вид:

$$\rho(z^*, z) = [K(z^*, z^*) + K(z, z) - 2K(z^*, z)]^{1/2}. \tag{3.9}$$

Это расстояние совпадает с евклидовым, если допустить, что

$$K(z^*, z) = (z^*, z) = \sum_{\nu=1} z_\nu^* z_\nu. \tag{3.10}$$

Если z^* и z — двоичные векторы и

$$w(x) = \sum_{\nu=1}^{N} |z_\nu| \tag{3.11}$$

— вес вектора (число единичных компонент), то можно ввести расстояние

$$g(z^*, z) = \frac{(z^*, z)}{w(z^*) + w(z) - (z^*, z)}, \tag{3.12}$$

и мера близости, предложенная Дж. Роджерсом и Т. Танимото [31], примет вид

$$\rho(z^*, z) = -\log_2 g(z^*, z). \tag{3.13}$$

Основной недостаток процедур выделения признаков, основанных на использовании расстояния, заключается в том, что получаемые в результате их применения решения неинвариантны относительно так называемых оптических преобразо-

ваний изображения. Оптическое преобразование изображения, заданного вектором с действительными компонентами, определяется формулой

$$T(z) = \langle az_1 + b, \ldots, az_n + b \rangle, \tag{3.14}$$

где a и b — некоторые положительные коэффициенты. Понятие инвариантности процедуры обнаружения относительно оптических преобразований можно раскрыть на следующем примере. Два изображения, представляющие один и тот же образ (букву «L»), должны быть отнесены к одному классу, несмотря на то что для их задания использованы векторы различной длины:

0	0	0	0	0
0	1	0	0	0
0	1	0	0	0
0	1	0	0	0
0	1	1	1	0
0	0	0	0	0

1	1	1	1	1
1	3	1	1	1
1	3	1	1	1
1	3	1	1	1
1	3	3	3	1
1	1	1	1	1

В данном случае $a = 2$, $b = 1$.

Эту трудность можно преодолеть, если вместо расстояния пользоваться мерой сходства

$$r(z^*, z) = \frac{\sum_{\nu=1}^{N} (z_\nu^* - \bar{z}_\nu^*)(z_\nu - \bar{z}_\nu)}{\sqrt{\sum_{\nu=1}^{N} (z_\nu^* - \bar{z}_\nu^*)^2 \sum_{\mu=1}^{N} (z_\mu - \bar{z}_\mu)^2}}, \tag{3.15}$$

где

$$\bar{z} = \frac{1}{N} \sum_{\nu=1}^{N} z_\nu \tag{3.16}$$

($\bar{z}^*$ определяется аналогичным образом).

Значительное количество примеров мер сходства приведено в статье Г. Н. Житкова, помещенной в сборнике [35].

Если для выделения локальных признаков используются метод, основанный на понятии расстояния, или корреляционный метод, то возникает задача оптимального выбора множества эталонных сигналов z^*; ее решению посвящена масса работ [2, 4, 5, 7, 8, 10, 12], которые мы не будем здесь подробно обсуждать, поскольку непосредственно с алгебраическими методами они не связаны.

Применение алгоритма обнаружения, предусматривающего использование коэффициента корреляции, начинает вызывать затруднения, если низшие уровни обработки изображения осно-

ваны на использовании логических процедур. Следовательно, представляет интерес задача аппроксимации корреляционного алгоритма множеством логических операций.

Рассмотрим локальный образ типа «концевая точка отрезка прямой». Это высказывание можно представить образами, имеющими, например, следующие реализации:

1 0 0 0 0	0 0 0 1 0	0 0 0 0 0	0 0 0 0 0	0 0 0 0 0
1 0 0 0 0	0 0 1 1 0	1 0 0 0 0	0 0 0 0 0	0 0 1 0 0
0 1 0 0 0	0 0 1 0 0	1 1 1 0 0	1 1 1 0 0	0 0 1 0 0
0 0 1 0 0	0 0 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 0	0 0 0 0 0	0 0 0 0 0	0 0 1 0 0

и т. д. Тест для локального образа подобного типа можно задать на языке отношений как отношение, которое выполняется на 9-мерных векторах $z(m, n)$ следующего вида:

$$\begin{matrix} z_{m-1,\,n-1} & z_{m-1,\,n} & z_{m-1,\,n+1} \\ z_{m,\,n-1} & z_{m,\,n} & z_{m,\,n+1} \\ z_{m+1,\,n-1} & z_{m+1,\,n} & z_{m+1,\,n+1} \end{matrix}$$

Отношение $R_1(m, n)$, соответствующее высказыванию «концевая точка отрезка прямой совпадает с точкой (m, n)», задается следующей формулой:

$$R_1(m, n) = \bigsqcup_{i=1}^{4} R_{11}^{(i)}(m, n) * R_{12}^{(i)}(m, n), \tag{3.17}$$

где

$$R_{11}^{1}(m, n) = \left\{ \begin{matrix} \langle 0\ 1 & . & \langle 1\ 0 & . & \langle 1\ 1 & . & \langle 1\ 0 & . \\ 0\ 1\rangle & . & 0\ 1\rangle & . & 0\ 1\rangle & . & 1\ 1\rangle & . \\ . \ . & ., & . \ . & ., & . \ . & ., & . \ . & . \end{matrix} \right\}, \tag{3.18}$$

$$R_{12}^{(1)}(m, n) = \left\{ \begin{matrix} . \ . & \langle 0 \\ 0\ 1 & 0 \\ 0\ 0 & 0\rangle \end{matrix} \right\} \tag{3.19}$$

(точка «.» означает, что для данной компоненты отношение не определено). Отношения $R_{11}^{(2)}$, $R_{12}^{(2)}$, $R_{11}^{(3)}$, $R_{12}^{(3)}$, $R_{11}^{(4)}$, $R_{12}^{(4)}$ можно получить из отношений $R_{11}^{(1)}$ и $R_{12}^{(1)}$ с помощью поворота квадратного участка против часовой стрелки на 90, 180 и 270° соответственно.

Применяя тест, построенный для отношения $R_1(m, n)$, к изображению, рассмотренному в приведенном выше примере, получаем

```
. . . . .  . . . . .  . . . . .  . . . . .  . . . .
. 0 0 0 .  . 0 0 0 .  . 0 0 0 .  . 0 0 0 .  . 0 1 0
. 0 0 0 .  . 0 1 0 .  . 0 1 0 .  . 0 0 0 .  . 0 0 0
. 0 1 0 .  . 0 0 0 .  . 0 0 0 .  . 0 1 0 .  . 0 0 0.
. . . . .  . . . . .  . . . . .  . . . . .  . . . .
```

(единицы соответствуют элементам, на которых отношения $R_1(m, n)$ выполняются).

Однако обычно изображение поступает на обработку далеко не в таком четком виде. Линии «размыты» подобно тому, как это представлено на следующих примерах:

```
0 1 1 1 0 0 0 0 0 0      0 0 0 0 0 0 0 0 0 0
0 0 1 1 1 1 0 0 0 0      0 0 0 0 0 0 0 0 0 0
0 0 0 1 1 1 0 0 0 0      0 0 0 1 1 1 1 1 0 0
0 0 0 1 1 1 1 0 0 0      1 1 1 1 1 1 1 0 0 0
0 0 0 0 1 1 1 0 0 0      1 1 1 0 0 1 1 1 0 0
0 0 0 0 1 1 1 1 0 0      1 1 0 0 0 0 1 1 1 0
0 0 0 0 0 1 1 0 0 0      1 1 0 0 0 0 1 1 0 0
0 0 0 0 0 0 0 0 0 0      1 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0      1 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0      0 0 0 0 0 0 0 0 0 0
```

Кроме того, до перехода к поиску концевой точки необходимо выделить «скелет» изображения.

Зададим для каждой компоненты $x_{m,n}$ ее «окрестность» с помощью вектора

$$\zeta(m, n) = \langle z_{m-1,n-1}, z_{m,n-1}, z_{m+1,n-1}, z_{m+1,n}, z_{m+1,n+1}, z_{m,n+1}, z_{m+1,n-1}, z_{m-1,n}\rangle. \quad (3.20)$$

Введем вес $w(z)$ вектора z.

Процедура выделения «скелета» размытого изображения основана на преобразовании

$$z_{m,n} := \begin{cases} 0, & \text{если } z_{m,n} = 0 \text{ или } z_{m,n} = 1 \text{ и } 2 < w(\zeta_{m,n}) \leqslant \theta, \\ 1 & \text{в остальных случаях} \end{cases} \quad (3.21)$$

(знак := следует читать «принимает значение»). Применяя к рассматриваемым примерам формулу (3.21), получаем

```
. . . . . .  . . . . . .
. 0 0 1 1 0 0 0 0 .  . 0 0 0 0 0 0 0 0 .
. 0 0 1 1 1 0 0 0 .  . 0 0 0 0 0 0 0 0 .
. 0 0 0 1 1 0 0 0 .  . 1 1 1 1 1 1 0 0 .
. 0 0 0 1 1 1 0 0 .  . 1 0 0 0 0 1 0 0 .
. 0 0 0 0 1 1 0 0 .  . 1 0 0 0 0 1 1 0 .
. 0 0 0 0 0 0 0 0 .  . 0 0 0 0 0 0 0 0 .
. 0 0 0 0 0 0 0 0 .  . 0 0 0 0 0 0 0 0 .
. 0 0 0 0 0 0 0 0 .  . 0 0 0 0 0 0 0 0 .
. . . . . .  . . . . . .
```

В случае необходимости эту процедуру можно повторить несколько раз. Первое повторное применение дает соответственно

```
. . . . . .  . . . . . .
. . . . . .  . . . . . .
. . 0 0 1 0 0 0 . .  . . 0 0 0 0 0 0 . .
. . 0 0 1 1 0 0 . .  . . 1 1 1 1 1 0 . .
. . 0 0 0 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 1 . .
. . 0 0 0 0 0 0 . .  . . 0 0 0 0 0 0 . .
. . 0 0 0 0 0 0 . .  . . 0 0 0 0 0 0 . .
. . 0 0 0 0 0 0 . .  . . 0 0 0 0 0 0 . .
. . . . . .  . . . . . .
. . . . . .  . . . . . .
```

Теперь оба изображения подготовлены к применению алгоритма, проверяющего выполнение «четких» локальных отношений.

Положение усложняется, если изображение искажено шумом, как, например, следующее:

исходное изображение

```
0 0 0 1 1 1 0 0 0 0
0 0 0 1 1 1 0 0 0 0
0 0 0 1 1 1 0 0 0 0
0 0 1 1 1 0 0 0 0 0
0 0 1 1 1 0 0 0 0 0
0 0 1 1 1 0 0 0 0 0
0 0 0 1 1 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0
```

размытое изображение

```
0 0 0 1 1 0 0 0 0 0
0 0 0 1 1 1 0 0 0 0
0 0 0 0 1 1 0 1 0 0
0 0 1 1 1 0 0 0 0 0
0 0 1 1 0 0 0 0 1 0
0 0 1 0 1 0 0 0 0 0
0 0 0 1 1 0 0 0 0 0
0 0 0 0 0 0 0 0 1 0
0 0 1 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0
```

С помощью использованного в предыдущем случае алгоритма, основанного на принципе «очищения», получаем следующее изображение, на котором оттеняются и искажения, и локальные информативные признаки:

```
. . . . . . . . .
. 0 0 0 1 0 0 0 0 .
. 0 0 0 1 0 0 1 0 .
. 0 0 0 0 0 0 0 0 .
. 0 0 1 0 0 0 0 1 .
. 0 0 0 1 0 0 0 0 .
. 0 0 0 0 0 0 0 0 .
. 0 0 0 0 0 0 0 0 .
. 0 1 0 0 0 0 0 0 .
. . . . . . . . .
```

Положение можно улучшить, использовав вместо алгоритма, предусматривающего локальный анализ изображения, алгоритм, обеспечивающий анализ изображения в целом. Продолжим рассмотрение предыдущего примера.

Выделим на сетчатке подучастки размера 3×3 и введем следующие подвекторы:

$$z'(m, n) = \langle z_{m-1, n-1}, z_{m-1, n}, z_{m-1, n+1}, z_{m, n-1}, z_{m, n}, z_{m, n+1}, z_{m+1, n-1}, z_{m+1, n}, z_{m+1, n+1} \rangle. \quad (3.22)$$

Положив $m = 2, 4, 6, 8, \ldots$ и $n = 2, 4, 6, 8, \ldots$, вычисляем значения весов $w[z'(m, n)]$ и формируем следующую матрицу:

$$\begin{matrix} w[z'(2,2)] & w[z'(2,4)] & \ldots \\ w[z'(4,2)] & w[z'(4,4)] & \ldots \\ \vdots & \vdots & \ddots \\ \vdots & \vdots & \ddots \\ \vdots & \vdots & \ddots \end{matrix}$$

Зашумленный вариант изображения, приведенный в примере, порождает следующую матрицу:

```
0 5 5 1 .
2 6 3 2 .
4 6 2 1 .
1 3 1 1 .
```

Обозначим элементы этой матрицы через $w_{i, j}$, $i = 1, 2, 3, \ldots$, $j = 1, 2, 3, \ldots$, и рассмотрим снова подвекторы размерности 3×3, на которых заданы следующие отношения:

отношение $R^1(i, j)$ справедливо для всех таких подвекторов, что

$$w_{i-1, j-1} + w_{i, j} + w_{i+1, j+1} > w_{i-1, j} + w_{i-1, j+1} + w_{i, j+1} \quad (3.23a)$$

и

$$w_{i-1,\,j-1} + w_{i,\,j} + w_{i+1,\,j+1} > w_{i,\,j-1} + w_{i-1,\,j+1} + w_{i+1,\,j}; \quad (3.23б)$$

отношение $R^{II}(i,\ j)$ справедливо для всех таких подвекторов, что

$$w_{i-1,\,j} + w_{i,\,j} + w_{i+1,\,j} > w_{i-1,\,j-1} + w_{i,\,j-1} + w_{i+1,\,j-1} \quad (3.24а)$$

и

$$w_{i-1,\,j} + w_{i,\,j} + w_{i+1,\,j} > w_{i-1,\,j+1} + w_{i,\,j+1} + w_{i+1,\,j+1}; \quad (3.24б)$$

отношение $R^{III}(i,\ j)$ справедливо для всех таких подвекторов, что

$$w_{i-1,\,j+1} + w_{i,\,j} + w_{i+1,\,j-1} > w_{i-1,\,j-1} + w_{i-1,\,j} + w_{i,\,j-1} \quad (3.25а)$$

и

$$w_{i-1,\,j+1} + w_{i,\,j} + w_{i+1,\,j-1} > w_{i,\,j+1} + w_{i+1,\,j} + w_{i+1,\,j+1}; \quad (3.25б)$$

отношение $R^{IV}(i,\ j)$ справедливо для всех таких подвекторов, что

$$w_{i,\,j-1} + w_{i,\,j} + w_{i,\,j+1} > w_{i-1,\,j-1} + w_{i-1,\,j} + w_{i-1,\,j+1} \quad (3.26а)$$

и

$$w_{i,\,j-1} + w_{i,\,j} + w_{i,\,j+1} > w_{i+1,\,j-1} + w_{i+1,\,j} + w_{i+1,\,j+1}. \quad (3.26б)$$

Можно сказать, что отношения R^{I}, R^{II}, R^{III} и R^{IV} соответственно определяют следующие направления прямой, проходящей через центр $(i,\ j)$ участка: «север — запад — юг — восток», «север — юг», «север — восток — юг — запад», «запад — восток». Применив эти отношения к матрице, составленной из элементов w_{ij}, получаем

$$\left.\begin{matrix} R^{II}(2,2) — \\ \\ R^{II}(2,4) \\ R^{IV}(2,4) \end{matrix}\right\} —$$

это означает, что прямая, имеющая направление «север — юг», проходит через левую половину исходного участка сетчатки (через нижний подучасток прямая проходит в горизонтальном направлении).

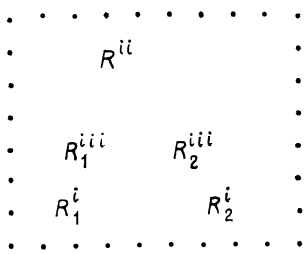
Р. Нарасимхан предложил еще ряд интересных и эффективных алгоритмов, предназначенных для реализации первых этапов структурного метода обработки зашумленных и размытых изображений [24, 25, 26, 27].

4. ВЫДЕЛЕНИЕ ИНТЕГРАЛЬНЫХ ПРИЗНАКОВ

Описание изображения, полученное в результате реализации начальных уровней обработки исходного сигнала, представляет собой набор локализованных локальных признаков. Так, например, входной сигнал вида

0 0 0 0 0 0 0 0 0 0
0 0 0 1 1 0 0 0 0 0
0 0 0 1 1 1 0 0 0 0
0 0 0 1 1 1 1 0 0 0
0 0 1 1 0 1 1 1 0 0
0 0 1 1 0 0 1 1 0 0
0 0 1 1 1 1 1 1 1 0
0 0 1 1 1 0 1 1 0 0
0 0 1 1 0 0 1 1 0 0
0 0 0 0 0 0 0 0 0 0

может быть перекодирован в следующую форму:



где через R^i, R^{ii} и R^{iii} обозначены следующие локальные отношения, выполнение которых было обнаружено:

R^i — «край линии»,

R^{ii} — «изгиб линии»,

R^{iii} — «разветвление линии».

Для того чтобы иметь возможность классифицировать исходное изображение как образ, необходимо ввести отношения высшего порядка. Прежде всего понадобятся отношения, характеризующие «взаимное положение»:

а) Отношение «отношение R^{I} расположено выше отношения R^{II}» можно определить как дизъюнкцию всех допустимых прямых произведений двух непустых локальных отношений R^{I} и R^{II} при условии, что эти отношения заданы на соответствующих подсемействах множеств $\langle U^{\mathrm{I}}\rangle$, $\langle U^{\mathrm{II}}\rangle \subset \langle U\rangle$, представляющих подучастки сетчатки, первый из которых расположен над вторым. Понятие «выше» можно относить и к положениям «центров тяжести», и к положениям верхних и нижних границ.

б) Отношение «отношение R^{I} расположено левее отношения R^{II}» можно аналогичным образом определить как дизъюнкцию

всех допустимых прямых произведений двух непустых локальных отношений R^{I} и R^{II} при условии, что эти отношения заданы на соответствующих подсемействах множеств $\langle U^{\mathrm{I}}\rangle$, $\langle U^{\mathrm{II}}\rangle \subset \subset \langle U\rangle$, представляющих подучастки сетчатки, первый из которых расположен левее второго.

Теперь описание образа можно представить в следующем виде:

$$R_{\mathrm{A}} = (R_1^{iii} \text{ выше } R_1^{i}) * (R_2^{iii} \text{ выше } R_2^{i}) * (R^{ii} \text{ выше } R_1^{iii}) * \\ * (R^{ii} \text{ выше } R_2^{iii}) * (R_1^{iii} \text{ левее } R_2^{iii}) * (R_1^{i} \text{ левее } R_2^{i}).$$

Данный метод описания образов инвариантен сдвигу, вращению и изменению формы, производимым в «разумных» пределах. «Буква А», описанная сугубо формально, вполне различима, несмотря на влияние шума и типографские дефекты, если описание достаточно адекватно и тесты, предназначенные для проверки наличия локальных отношений, обладают достаточной мощностью.

Набор отношений, характеризующих взаимное расположение участков, на которых проверяется наличие локальных признаков, может быть при необходимости дополнен отношениями, определяющими топологические или геометрические характеристики.

Рассмотрим алгебру отношений

$$G = \langle\langle U\rangle, Q_{\langle U\rangle}, \theta, *, -\rangle. \tag{4.1}$$

Всякое отношение R, принадлежащее этой алгебре, может определять образ, если интерпретацией $\langle U\rangle$ является семейство множеств значений исходного сигнала. Если, однако, задано подмножество некоторых «локальных» отношений $\{R_i\}$, то можно задать подмножество $B_{\{R_i\}} \subset Q_{\langle U\rangle}$ всех отношений, представимых посредством некоторых алгебраических комбинаций отношений, принадлежащих $B_{\{R_i\}}$. Всякое отношение $R \in B_{\{R_i\}}$ можно представить термом, составленным из локальных отношений и символов операций дополнения $^-$, свертки $*$ и прямого произведения $\times$ отношений. Очевидно, что для заданного множества образов, представленных отношениями $R_{\mathrm{A}}, \ldots, R_{\mathrm{K}}$, множество $B_{\{R_i\}}$ следует выбирать так, чтобы

$$R_{\mathrm{A}}, \ldots, R_{\mathrm{K}} \in B_{\{R_i\}}. \tag{4.2}$$

С другой стороны, при выполнении последнего условия возникает задача построения кратчайших выражений, представ-

ляющих заданные отношения. Благодаря наличию изоморфизма между алгеброй отношений G и булевыми алгебрами эту задачу можно решать с помощью общепринятых методов минимизации нормальных форм, представляющих булевы функции.

Попробуем более точно определить значение понятия «алгебраическое выражение, образованное отношениями, принадлежащими $\{R_i\}$»:

а) всякий символ R_j, обозначающий отношение, принадлежащее $\{R_i\}$, есть выражение;

б) если q — выражение, то $\bar{q}$ также является выражением;

в) если q^{I} и q^{II} — некоторые выражения, то их свертка $q^{\mathrm{I}} * q^{\mathrm{II}}$ и прямое произведение $q^{\mathrm{I}} \times q^{\mathrm{II}}$ также являются выражениями;

г) если $q_1, q_2, \ldots, q_i, \ldots$ — некоторые выражения, то $*q_i$ — выражение, эквивалентное выражению $q_1 * q_2 * \ldots * q_i * \ldots$, и $\bigtimes_i q_i$ — выражение, эквивалентное выражению $q_1 \times q_2 \times \ldots \ldots \times q_i \times \ldots$.

Содержание пунктов а) — г) — это не что иное, как грамматические правила формального языка, описывающего принадлежащие $B_{\{R_i\}}$ отношения. Множество $B_{\{R_i\}}$ образует семантическое поле языка, а заданные отношения $R_{\mathrm{A}}, \ldots, R_K$, удовлетворяющие условию (4.2), составляют его прагматику.

Два выражения q^{I} и q^{II} называются эквивалентными в формальном смысле, если каждое из них может быть преобразовано в другое с помощью формул (2.24) — (2.27). Доказательство формальной тождественности выражения играет важную роль в структурном методе распознавания. Понятно, что заданный образ может иметь различные формальные описания, и, следовательно, необходимо уметь доказывать, что выражение, полученное в результате конкретного распознавательного эксперимента, описывает тот же самый образ, который представлен выражением, полученным дедуктивным способом. Задача установления формальной тождественности выражений близка задаче автоматического доказательства теорем и может решаться теми же методами.

Формальную тождественность выражений не следует смешивать с семантической. Выражения называются тождественными семантически, если они представляют идентичные отношения, т. е. отношения, выполняющиеся на одних и тех же множествах реализации. Обозначим класс выражений, тождественных выражению q формально, через $h(q)$ и класс выражений, тождественных выражению q семантически, через $H(q)$. Очевидно, что

$$h(q) \subset H(q). \tag{4.3}$$

Рассмотрим в качестве примера задачу распознавания печатных букв. Множество основных отношений может содержать как обсуждавшиеся выше отношения типа R^{i}, R^{ii}, R^{iii}, так и отношения высшего порядка типа «выше» «левее» и т. п. Можно определить множество основных отношений и по-другому, включив в него отношения типа «вертикальный отрезок прямой», «горизонтальный отрезок прямой», «выпуклая дуга» и т. д. Выражения, описывающие образ, в формальном смысле тождественны не будут, хотя они могут оказаться тождественными в семантическом смысле.

Множество $\langle U\rangle$ с заданным на нем семейством отношений $\{R_i\}$ образует структуру в алгебраическом смысле этого понятия. Задачи распознавания образов можно с точки зрения теории отношений отнести к конструктивным задачам последней.

Рассмотрим два линейно упорядоченных семейства множеств $\langle U\rangle$ и $\langle V\rangle$ и два отношения R_U и R_V, заданные на семействах $\langle U\rangle$ и $\langle V\rangle$ соответственно. Пусть ψ — взаимно однозначная проекция семейства $\langle U\rangle$ в семейство $\langle V\rangle$ и $\{\varphi_i(x)\}$ — семейство взаимно однозначных проекций вида

$$\varphi_i\colon\ U_i \to V_i,\quad V_i = \psi(U_i). \tag{4.4}$$

Другими словами, $\varphi_i(x)$ — функция, реализующая проекцию множества U_i в множество V_i, если между множествами U_i и V_i имеет место взаимно однозначное соответствие, предусматриваемое ψ-проекцией. Очевидно, что семейства множеств $\langle U\rangle$ и $\langle V\rangle$, а также находящиеся в отношении соответствия множества U_i и V_i являются попарно равномощными.

Допустим, что семейства множеств $\langle U\rangle$ и $\langle V\rangle$ линейно упорядочены. Будем считать, что взаимно однозначная проекция ψ обеспечивает сохранение порядка, если для всякого $U_1 \prec \prec U_2 \prec \ldots U_k \in \langle U\rangle$ (порядок, введенный в семействе $\langle U\rangle$) выполняется

$$\psi(U_1)' \prec \psi(U_2)' \prec \ldots{}' \prec \psi(U_k) \tag{4.5}$$

(порядок, введенный в семействе $\langle V\rangle$). Нас интересуют только те проекции ψ, которые обеспечивают сохранение порядка. Любая реализация $z \in R_U$ проецируется семейством проекций φ_i в V-пространство последовательностей; все полученные таким способом последовательности можно рассматривать как новое отношение — обозначим его R'_U. Отношения R_U и R_V называются изоморфными, если проекции ψ, $\{\varphi_i(x)\}$ можно выбрать таким образом, что проекция ψ обеспечивает сохранение порядка и отношение R'_U тождественно отношению R_V.

Проиллюстрируем эти понятия примером. Рассмотрим следующие подсемейства семейства множеств, характеризующего

состояние сетчатки:

$$\begin{matrix}
\langle U_{k-1,\,l-1} & U_{k-1,\,l} & \dots & . & . \\
U_{k,\,l-1} & U_{k,\,l}\rangle & \dots & . & . \\
. & . & . & . & . \\
. & . & . & . & . \\
. & . & . & . & . \\
. & . & \dots & \langle U_{m-1,\,n-1} & U_{m-1,\,n} \\
. & . & \dots & U_{m,\,n-1} & U_{m,\,n}\rangle
\end{matrix}$$

Рассмотрим семейство множеств $\langle U_{k-1,\,l-1},\ U_{k-1,\,l},\ U_{k,\,l-1},\ U_{k,\,l}\rangle$ и заданное на нем отношение

$$R' = \{\langle 10\ 10\rangle,\ \langle 01\ 01\rangle\}.$$

Рассмотрим также семейство множеств $\langle U_{m-1,\,n},\ U_{m,\,n},\ U_{m-1,\,n-1},\ U_{m,\,n-1}\rangle$ и отношение

$$R^{\mathrm{II}} = \{\langle 11\ 00\rangle,\ \langle 00\ 11\rangle\}.$$

Отношения R^{I} и R^{II}, конечно, не тождественны; тем не менее они изоморфны. Зададим проекцию:

$$\psi(U_{k-1,\,l-1}) = U_{m-1,\,n},\quad \psi(U_{k,\,l-1}) = U_{m-1,\,n-1},$$
$$\psi(U_{k-1,\,1}) = U_{m,\,n},\qquad \psi(U_{k,\,l}) = U_{m,\,n-1}.$$

Очевидно, что данная проекция обеспечивает сохранение порядка. Далее можно задать еще четыре проекции:

$$\varphi_1(1) = \varphi_2(1) = \varphi_3(1) = \varphi_4(1) = 1,$$
$$\varphi_1(0) = \varphi_2(0) = \varphi_3(0) = \varphi_4(0) = 0.$$

Проекция отношения R^{I} в семейство множеств $\langle U_{m-1,\,n},\ U_{m,\,n},\ U_{m-1,\,n-1},\ U_{m,\,n-1}\rangle$ приводит к отношению

$$R^{\mathrm{III}} = \{\langle 11\ 00\rangle,\ \langle 00\ 11\rangle\},$$

тождественному отношению R^{II}. В таком случае отношения R^{I} и R^{II} изоморфны. Этот тип изоморфизма можно определить как «перенос».

Изоморфизм отношений, заданных на одном и том же семействе множеств (при разных способах упорядочения), назовем автоморфизмом. Отношение R^{I}, введенное в последнем примере, и отношение R^{IV}, заданное на семействе множеств $\langle U_{k,\,l-1},\ U_{k-1,\,l-1},\ U_{k,\,l},\ U_{k-1,\,1}\rangle$ в виде

$$R^{\mathrm{IV}} = \{\langle 11\ 00\rangle\ \langle 00\ 11\rangle\},$$

взаимно автоморфны: первое отношение описывает горизонтальный отрезок прямой, второе — вертикальный. Этот тип автоморфизма можно назвать «поворотом».

Можно определить и другие разновидности изоморфизмов, например изменение масштаба, и т. п. Заметим, с другой стороны, что оптическое преобразование, заданное формулой (3.14), не является изоморфизмом отношений в упомянутом выше смысле.

Рассмотрим более подробно общий случай. Заданы два семейства множеств $\langle U\rangle$ и $\langle V\rangle$ с отношениями R_U и R_V соответственно. Пусть ψ — взаимно однозначная проекция семейства $\langle U\rangle$ в семейство $\langle V\rangle$. Будем считать, однако, что проекции, определенные отображением (4.4), однозначные, но не взаимно обратные. В таком случае всякая реализация отношения R_U может быть спроецирована в одну и только одну реализацию отношения R_V, а всякая реализация отношения R_V может быть проекцией по крайней мере одной реализации отношения R_U. В данном случае отношение R_V гомоморфно отношению R_U. Следующий пример иллюстрирует понятие гомоморфизма отношений.

Рассмотрим множества U_i, $i = 1, 2, \ldots, 9$, в состав которых входят три элемента 0, 1, 2, и множества V_j, $j = 1, 2, \ldots, 9$, в состав которых входят по два элемента — 0 и 1. Представим семейства множеств, как мы это обычно делали, в виде квадратных массивов:

$$\begin{array}{lll} \langle U_1 & U_2 & U_3 \\ U_4 & U_5 & U_6 \\ U_7 & U_8 & U_9\rangle \end{array} \qquad \begin{array}{lll} \langle V_1 & V_2 & V_3 \\ V_4 & V_5 & V_6 \\ V_7 & V_8 & V_9\rangle \end{array}$$

Определим отношение «пересечение»:

$$R_U = \{\langle 101\ 010\ 101\rangle, \langle 212\ 121\ 212\rangle, \langle 202\ 020\ 202\rangle, \langle 010\ 111\ 010\rangle, \langle 121\ 222\ 121\rangle, \langle 020\ 222\ 020\rangle\}.$$

Отношение

$$R_V = \{\langle 101\ 010\ 101\rangle, \langle 010\ 111\ 010\rangle\},$$

заданное на семействе множеств $\langle V\rangle$, можно считать гомоморфным относительно отношения R_U. Обычно квантование сигнала по уровням, сжатие или ограничение приводят к гомоморфизму отношений. В то же время метод преобразования сигнала, описанный в предыдущем разделе и основанный на использовании усреднения, нельзя отнести к категории ведущих к гомоморфизму преобразований в нашем смысле.

Представляют интерес и некоторые другие разновидности морфизмов. Если задана проекция R'_U отношения R_U в определенное подсемейство множеств и отношение R_V, то последнее называется подизоморфным отношению R_U при условии, что оно изоморфно (соответственно гомоморфно) отношению R'_U.

С другой стороны, отношение R_V называется надизоморфным (надгомоморфным) отношению R_U, если существует проекция R'_V отношения R_V, изоморфная (соответственно гомоморфная) отношению R_U. Всякая проекция отношения подизоморфна (точнее, подавтоморфна) исходному отношению. Можно также определить под-, над- и гомоморфизм для подотношений. Мы будем пользоваться понятием частичного морфизма отношений R_U и R_V в случае, когда существуют некоторые подотношения $R'_U \subset R_U$ и $R'_V \subset R_V$ и для них имеет место некий морфизм, который не распространяется на отношения R_U и R_V.

Введенные понятия морфизмов отношений мы будем использовать для классификации морфизмов алгебраических структур. Рассмотрим две алгебраические структуры

$$T_U = \langle\langle U\rangle, \{R^i_U\}\rangle, \tag{4.6а}$$

$$T_V = \langle\langle V\rangle, \{R^i_V\}\rangle. \tag{4.6б}$$

Назовем структуры T_U и T_V изоморфными, если существует такое множество проекций ψ, $\{\varphi_i\}$, что отношения $R^i_U \in \{R^i_U\}$ и $R^i_V \in \{R^i_V\}$ одновременно попарно изоморфны. Аналогичным образом определяется гомоморфизм, а также и под-, над- и частичные морфизмы структур. Очевидно, что задачи распознавания образов нет необходимости рассматривать вне пределов классов изоморфных структур. Частичный характер, однако, может иметь не только морфизм отношений, но и морфизм структур. В частности, задачи распознавания, связанные с автоматическим считыванием букв английского и русского алфавита, можно формально представить посредством частично изоморфных структур, так как отношения, характеризующие печатные буквы «А», «В», «С», «Е», «Н», в обоих случаях одинаковы. Проблема подобия структур, однако, много глубже. Хорошо известно, например, что любая практическая задача распознавания может быть эффективно решена множеством формально различных способов. С другой стороны, формальное подобие отдельных подходов не гарантирует их совпадение по эффективности. Необходимо, следовательно, пользоваться более гибкими понятиями структурного подобия и эквивалентности, чем те, что основаны на концепции морфизма. В принципе этого можно добиться, введя в пространство структур какие-то характеристики расстояния.

Допустим, что нас интересует соответствие на нижнем уровне структуры. Это означает, что заданы семейства множеств $\langle U\rangle$ и $\langle V\rangle$ и класс проекций ψ, $\{\varphi_i\}$. Аппарат алгебры отношений позволяет описать метаотношения на отношениях $\{R^i_U\}$ и на отношениях $\{R^i_V\}$. Эти метаотношения могут быть пред-

ставлены в терминах подотношений, сверток, прямых произведений и т. д. Заметим, что истинны следующие метаотношения: пусть $A(R)$ означает, что отношение R выполняется на данной реализации; в таком случае

$$\begin{aligned}
&\text{а)}\ A(R) \Leftrightarrow A(R);\\
&\text{б)}\ A(R) \Rightarrow A(R\langle U^{\mathrm{I}}\rangle) \text{ для любого подсемейства множеств } \langle U^{\mathrm{I}}\rangle;\\
&\text{в)}\ A(R^{\mathrm{I}}) \Rightarrow A(R) \text{ для любого } R^{\mathrm{I}} \subset R;\\
&\text{г)}\ A(R^{\mathrm{I}}) \Rightarrow A(R^{\mathrm{I}} \times R^{\mathrm{II}}) \text{ для любого отношения } R^{\mathrm{II}};\\
&\text{д)}\ A(R^{\mathrm{I}} * R^{\mathrm{II}}) \Rightarrow A(R^{\mathrm{I}}),\ A(R^{\mathrm{II}});\\
&\text{е)}\ A(\bar{R}) \Rightarrow \neg A(R).
\end{aligned} \tag{4.7}$$

Эти свойства позволяют ввести полуупорядочение на множестве отношений. Пусть

$$R^{\mathrm{I}} \prec R^{\mathrm{II}} \text{ в том и только том случае, если}$$
$$A(R^{\mathrm{I}}) \Rightarrow A(R^{\mathrm{II}}). \tag{4.8}$$

Нетрудно показать, что эти новые метаотношения обратимы (см. (4.7а)), антисимметричны (если $R^{\mathrm{I}} \prec R^{\mathrm{II}}$ и $R^{\mathrm{II}} \prec R^{\mathrm{I}}$, то $R^{\mathrm{I}} \equiv R^{\mathrm{II}}$) и транзитивны. Для представления таких метаотношений можно использовать граф, вершины которого соответствуют отношениям рассматриваемой структуры, а ориентированные ребра — импликациям. Рассмотрим соответствующий пример.

Дано семейство множеств

$$\langle U_1\ \ U_2\ \ U_3$$
$$U_4\ \ U_5\ \ U_6$$
$$U_7\ \ U_8\ \ U_9\rangle$$

и следующие отношения:

$$R_1 = \{\langle 100\ 100\ 100\rangle,\ \langle 010\ 010\ 010\rangle,\ \langle 001\ 001\ 001\rangle,$$
$$R_2 = \{\langle 111\ 000\ 000\rangle,\ \langle 000\ 111\ 000\rangle,\ \langle 000\ 000\ 111\rangle\}.$$

Рассмотрим их прямое произведение

$$R_3 = R_1 \cup R_2 \equiv R_1 \times R_2 = \{\langle 100\ 100\ 100\rangle,\ \langle 010\ 010\ 010\rangle,$$
$$\langle 001\ 001\ 001\rangle,\ \langle 111\ 000\ 000\rangle,\ \langle 000\ 111\ 000\rangle,\ \langle 000\ 000\ 111\rangle\}$$

и проекцию

$$R_4 = R_{3\langle U_1,\ U_4,\ U_7\rangle} = \{\langle 000\rangle,\ \langle 111\rangle,\ \langle 100\rangle,\ \langle 010\rangle,\ \langle 001\rangle\}.$$

Множество отношений $\{R_1, R_2, R_3, R_4\}$ можно полуупорядочить следующим образом:

$$R_3 \succ R_1, \quad R_3 \succ R_2, \quad R_3 \prec R_4.$$

Граф, представляющий данные метаотношения, приведен на рис. 4.

При сопоставлении структур можно исходить из подобия графов, представляющих их внутреннюю логическую микроструктуру. Эту общую идею можно усовершенствовать, введя дополнительное индексирование вершин графа: всем вершинам, соответствующим изоморфным отношениям, можно приписать один и тот же индекс класса изоморфных отношений. В приве-

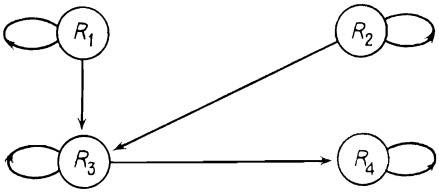


Рис. 4.

денном выше примере такой дополнительный индекс следует приписать вершинам, соответствующим отношениям R_1 и R_2.

5. ФОРМАЛЬНЫЕ ЯЗЫКИ ДЛЯ ОБРАБОТКИ ИЗОБРАЖЕНИЙ

В последние годы все большее внимание уделяется построению систем распознавания образов, основанных на работе человека с вычислительной машиной в режиме взаимодействия. Наиболее представительным примером подобного подхода является вычислительная система Illiac III, разработанная и реализованная в Университете штата Иллинойс, США [21, 22, 28, 30]. В то же время в практических приложениях важную роль, вероятно, будут играть меньшие и более специализированные системы распознавания и обработки изображений, построенные на основе мини-ЭВМ третьего поколения. В техническом проектировании, при обработке экспериментальных данных и проведении криминологических экспертиз возможность управлять процессом принятия решения, по крайней мере на его ключевых этапах, кажется чрезвычайно привлекательной. Система SHOW-and-TELL Interactive Programming System for Image Processing, предназначенная для обработки изображений в режиме взаимодействия, разработана в Отделении вычислительной математики и кибернетики Университета штата Иллинойс. В системе используется формальный язык, ориентированный на реализацию алгоритмов обработки изображений. Общие принципы построения этого языка, описанные Р. Нарасимханом [26],

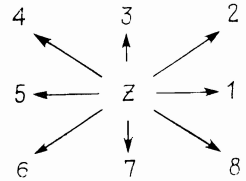
были также реализованы с некоторыми модификациями на польской вычислительной машине типа ODRA 1204. Этот вариант языка, названный PICTURE ALGOL 1204, следует рассматривать как удобное средство для разработки и испытания новых алгоритмов распознавания и обработки изображений, а не как окончательный вариант программного обеспечения, предназначенный для нормальной эксплуатации. Язык PICTURE ALGOL 1204A позволяет работать с двоичной («черно-белой») графической информацией, задаваемой в виде построенной из точек матрицы $M \times N$, причем число столбцов N в настоящее время ограничивается стандартным размером слова в вычислительной машине типа ODRA 1204 (24 бита). Число строк M не ограничено ничем, кроме размера оперативной памяти. Поскольку PICTURE ALGOL 1204A является подъязыком АЛГОЛа, имеется возможность использовать его стандартные процедуры, предназначенные для работы не с реальными, а с целыми переменными. При реализации процедур «PICTURE ALGOL» используются целые переменные M и N, а также несколько вспомогательных переменных («BOUND» и «THR»).

В языке «PICTURE ALGOL» применяются стандартные процедуры нескольких типов. Алгебраические процедуры над множествами позволяют реализовывать основные теоретико-множественные операции над множествами элементов, из которых состоят изображения. Эти процедуры соответствуют основным операциям дизъюнкции, конъюнкции и определения разности отношений, заданных на бинарных множествах и представленных отдельными реализациями. Еще две процедуры предназначены для стирания изображения и заполнения соответствующего участка (отведенного для массива определенной переменной) нулями или записи в него нового изображения. Остальные процедуры этой первой группы предназначены для присвоения значения «true» булевой переменной, если изображение состоит из одних нулей или если два изображения идентичны.

Вторая группа стандартных процедур обеспечивает возможность установления принадлежности точки заданному изображению (является ли изображение в этой точке «черным»), а также добавления этой точки к изображению или устранению ее из изображения.

Следующая группа процедур позволяет реализовывать более сложные преобразования изображений. В принципе способ обработки конкретного изображения зависит от некоторого другого изображения $Z2$, называющегося «контекстом», а результат обработки представляется в виде какого-то третьего изображения $Z3$. Каждой точке изображения z ставятся в соответствие девять точек, образующих окрестность (включая саму точку z). Адреса этих точек представляются вспомогательной перемен-

ной «направление», принимающей значения согласно следующей схеме:



адрес самой точки z соответствует направлению, равному 0.

Процедура MARK ($Z1$, $Z3$, направление) вводит в изображение $Z3$ все точки, принадлежащие определенному направлению окрестности точек изображения $Z1$. Так, в частности, если направление есть «123», то это означает, что все точки с направлениями 1, 2 и 3 принадлежат рассматриваемой окрестности, и если изображение $Z1$ имеет вид

```
0 0 0 0 0 0 0 0
0 0 0 1 1 1 0 0
0 0 0 1 1 1 0 0
0 0 0 1 1 1 0 0
0 0 1 1 1 0 0 0
0 1 1 1 0 0 0 0
0 1 1 1 0 0 0 0
0 0 0 0 0 0 0 0
```

то изображение $Z3$, полученное в результате применения процедуры MARK ($Z1$, $Z3$, «123»), предстает в следующем виде:

```
0 0 0 1 1 1 1 0
0 0 0 1 1 1 1 0
0 0 0 1 1 1 1 0
0 0 1 1 1 1 0 0
0 1 1 1 1 0 0 0
0 1 1 1 1 0 0 0
0 0 1 1 1 0 0 0
0 0 0 0 0 0 0 0
```

Эта стандартная процедура не зависит от контекста.

Еще одна процедура, называющаяся CMARK ($Z1$, $Z2$, $Z3$, направление), вводит точку в изображение $Z3$, если она принадлежит определенной окрестности точки изображения $Z1$ и в то же время принадлежит заданному изображению $Z2$. Последнее можно выбрать таким образом, чтобы $Z3$ было запи-

сано в пределах определенного участка. Так, если в предыдущем примере придать изображению $Z2$ вид

```
0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0
0 0 1 1 1 1 0 0
0 0 1 1 1 1 0 0
0 0 1 1 1 1 0 0
0 0 1 1 1 1 0 0
0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0
```

то применение стандартной процедуры CMARK ($Z1$, $Z2$, $Z3$, «123») приведет к получению следующего результата:

```
0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0
0 0 0 1 1 1 0 0
0 0 1 1 1 1 0 0
0 0 1 1 1 0 0 0
0 0 1 1 1 0 0 0
0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0
```

Для процедур MARK и CMARK можно ввести следующее обобщение. Вводится булева переменная bex(THR), где целое, обозначенное THR, представляет вес окрестности заданной точки изображения $Z2$, подсчитанный по указанным направлениям. Значение переменной bex равно «истина», если величина веса, вычисленная для окрестности рассматриваемой точки изображения $Z2$, больше значения порога или равна ему. Применив, в частности, процедуру THRESHOLDCMARK ($Z1$, $Z2$, $Z3$, «12345678», THR $<$ 8) к изображению $Z1$ вида

```
0 0 0 0 0 0 0 0 0
0 0 0 0 1 1 0 0 0
0 0 0 1 1 1 0 0 0
0 0 1 1 1 1 0 0 0
0 0 0 1 1 1 0 0 0
0 0 0 1 1 1 0 0 0
0 0 0 1 1 1 1 0 0
0 0 1 1 1 1 1 0 0
0 0 0 0 0 0 0 0 0
```

получаем контур изображения:

```
0 0 0 0 0 0 0 0 0
0 0 0 0 1 1 0 0 0
0 0 0 1 1 1 0 0 0
0 0 1 1 0 1 0 0 0
0 0 0 1 0 1 0 0 0
0 0 0 1 0 1 0 0 0
0 0 0 1 0 1 1 0 0
0 0 1 1 1 1 1 0 0
0 0 0 0 0 0 0 0 0
```

Некоторые дополнительные возможности предоставляет использование идеи «цепочки». Последняя состоит из всех точек, принадлежащих заданному направлению на обоих изображениях $Z1$ и $Z2$ относительно точки, принадлежащей изображению $Z1$. Эту идею можно проиллюстрировать следующим образом. Заданы изображения $Z1$ и $Z2$, имеющие соответственно вид

```
0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0
0 0 1 0 0 0 0 0 0
0 0 0 1 0 0 0 0 0
0 0 0 0 0 0 0 0 0
0 0 0 0 0 1 0 0 0
0 0 0 0 0 0 0 0 0
```

```
0 0 0 0 0 0 0 0 0
0 0 1 1 1 1 1 0 0
0 1 1 1 1 1 1 1 0
0 1 1 1 1 1 1 1 0
0 1 1 1 1 1 1 1 0
0 1 1 1 1 1 1 1 0
0 0 1 1 1 1 1 0 0
0 0 0 0 0 0 0 0 0
```

Применение процедуры CHAIN ($Z1$, $Z2$, $Z3$, направление) для направления «1, 2» позволяет получить изображение $Z3$ следующего вида:

```
0 0 0 0 0 0 0 0 0
0 0 0 0 1 0 0 0 0
0 0 0 1 0 1 0 0 0
0 0 1 1 1 1 1 1 0
0 0 0 1 1 1 1 1 0
0 0 0 0 0 0 0 1 0
0 0 0 0 0 1 0 0 0
0 0 0 0 0 0 0 0 0
```

Процедура CHAIN в сочетании с процедурой вычисления порога обеспечивает возможность обнаруживать на размытом изображении некоторые локальные признаки. Для реализации этой операции предусмотрена соответствующая стандартная процедура THRESHOLDCHAIN ($Z1$, $Z2$, $Z3$, направление, bex).

В результате применения этой процедуры строится цепочка точек, принадлежащих изображению $Z1$, относительно точки изображения $Z2$; она переносится на изображение $Z3$ только в том случае, если булева переменная bex принимает значение «истина». Ее значение определяется из неравенства

$$\mathrm{THR} > w,$$

где THR — целая переменная, представляющая вес цепочки, а w — заданное значение порога.

Рассмотрим следующий пример. Задано искаженное помехами изображение $Z1$:

```
0 0 0 1 0 1 0 1
1 0 0 1 1 0 0 1
0 0 1 0 0 1 1 0
0 0 1 1 1 0 0 1
1 1 0 0 0 1 0 0
0 1 1 0 1 1 0 0
1 1 1 0 0 0 1 1
0 1 1 0 1 0 0 0
```

Сопоставляются следующие гипотезы: нечеткая линия проходит в направлении «26» или в направлении «48». Вводятся эталоны $Z21$ и $Z22$, имеющие вес 22:

```
1 1 0 0 0 0 0 0        0 0 0 0 0 0 1 1
1 1 1 0 0 0 0 0        0 0 0 0 0 1 1 1
0 1 1 1 0 0 0 0        0 0 0 0 1 1 1 0
0 0 1 1 1 0 0 0        0 0 0 1 1 1 0 0
0 0 0 1 1 0 0 0        0 0 1 1 1 0 0 0
0 0 0 0 1 1 1 0        0 1 1 1 0 0 0 0
0 0 0 0 0 1 1 1        1 1 1 0 0 0 0 0
0 0 0 0 0 0 1 1        1 1 0 0 0 0 0 0
```

Воспользуемся двумя процедурами: THRESHOLDCHAIN ($Z1$, $Z21$, $Z31$, «026» $\mathrm{THR} > 1$) и THRESHOLDCHAIN ($Z1$, $Z22$, $Z32$, «048», $\mathrm{THR} > 0$). Их применение приводит к получению двух изображений $Z31$ и $Z32$:

```
0 1 0 0 0 0 0 0        0 0 0 0 0 0 1 1
1 0 0 0 0 0 0 0        0 0 0 0 0 1 0 1
0 0 1 1 0 0 0 0        0 0 0 0 0 1 1 0
0 0 1 1 1 0 0 0        0 0 0 1 1 0 0 0
0 0 0 1 0 1 0 0        0 0 0 0 1 0 0 0
0 0 0 0 1 1 0 0        0 1 0 0 0 0 0 0
0 0 0 0 0 0 1 0        1 1 1 0 0 0 0 0
0 0 0 0 0 0 0 0        0 1 0 0 0 0 0 0
```

Второе из них, имеющее больший вес, соответствует более вероятному направлению линии. Далее, применив процедуру CHAIN (*Z*32, *Z*22, «26»), получаем восстановленное изображение:

0 0 0 0 0 0 1 1
0 0 0 0 0 1 1 1
0 0 0 0 1 1 1 0
0 0 0 1 1 1 0 0
0 0 1 1 1 0 0 0
0 1 1 1 0 0 0 0
1 1 1 0 0 0 0 0
1 1 0 0 0 0 0 0

При необходимости изображение можно сжать с помощью процедуры CHAIN (*Z*5, *Z*6, *Z*7, «0»), изображение *Z*6 имеет при этом следующий вид:

0 0 0 0 0 0 0 1
0 0 0 0 0 0 1 0
0 0 0 0 0 1 0 0
0 0 0 0 1 0 0 0
0 0 0 1 0 0 0 0
0 0 1 0 0 0 0 0
0 1 0 0 0 0 0 0
1 0 0 0 0 0 0 0

Полученное в результате изображение *Z*7 идентично изображению *Z*6.

Для выделения локальных отношений, представленных несколькими альтернативными реализациями, предназначена стандартная процедура TPANSFORM (*Z*1, *Z*2, *Z*3, bf). Символом «bf» обозначена последовательность, в которую могут входить числа от 0 до 8, причем некоторые из них могут быть подчеркнуты и некоторые последовательности могут отделяться знаком «+». Запись этой процедуры в виде TRANSFORM (*Z*1, *Z*2, *Z*3, «$\overline{23} + \overline{45}$») означает, что изображение *Z*3 состоит из всех точек, принадлежащих изображению *Z*1 и таких, что их окрестности относительно изображения *Z*2 непусты в направлении «3» и пусты в направлении «2» или непусты в направлении «4» и пусты в направлении «5». В частности, для того чтобы обнаружить все края линии, можно использовать следующую процедуру: TRANSFORM (*Z*1, *Z*2, Z3, «12345678 +

$+\ \overline{12345678} + \overline{12345678} + \overline{12345678} + \overline{12345678} + \overline{12345678} +$
$+\ \overline{12345678} + \overline{12345678}$»). Для изображения $Z1$ и $Z2$ вида

0 1 0 0 0 0 0 1	0 0 0 0 0 0 0 0
0 1 0 0 1 1 1 0	0 0 0 0 0 0 0 0
0 0 1 0 1 0 0 0	0 0 0 0 0 0 0 0
0 0 1 1 1 1 1 0	0 0 0 0 0 0 0 0
0 1 1 0 0 0 0 0	1 1 1 1 0 0 0 0
0 0 0 1 1 1 0 0	1 1 1 1 0 0 0 0
0 1 1 1 0 0 1 0	1 1 1 1 0 0 0 0
0 0 0 0 0 0 0 1	1 1 1 1 0 0 0 0

изображение $Z3$, полученное в результате реализации процедуры, принимает следующий вид:

0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0
0 1 0 0 0 0 0 0
0 0 0 0 0 0 0 0
0 1 0 0 0 0 0 0
0 0 0 0 0 0 0 0

При осуществлении операций, связанных с вводом и выводом информации, используются и некоторые другие стандартные процедуры языка «PICTURE ALGOL 1204A».

Заслуживают упоминания и несколько языков обработки изображений, относящихся к подмножеству ФОРТРАНа (PAX, PAX II, COMPAX, STANDPAX). Язык PAX II позволяет обрабатывать изображения произвольных размеров и многих уровней зачерненности [19]. Прогресс в данной области благоприятствует развитию методов распознавания, но, вообще говоря, проблема формальных языков в распознавании много шире. Изображение — одна из основных форм представления научной, технической и многих других разновидностей информации. Сами конструкторские чертежи, электронные схемы, топографические карты, структурные формулы химических веществ и т. п. можно рассматривать в качестве некоторых формальных языков.

Задача распознавания образов представляет собой задачу трансляции «плоскостного» языка в «линейный», оперирующий последовательностями символов. Общая теория плоскостных языков находится, однако, на самом начальном этапе развития. Ясно, что в принципе плоскостной язык, как и линейный, можно определить следующей упорядоченной парой [20]:

$$L = \langle A, F \rangle, \tag{5.1}$$

где A — множество основных элементов (символы, слова), а F — множество «выражений», правильных в некотором формальном смысле. Основная сложность заключается в определении структуры «выражения». В линейных языках, как это было показано в гл. 4, структура задается с помощью бинарных ассоциативных и некоммутативных операций «сочленения» последовательностей элементов, принадлежащих алфавиту A. В плоскостных языках ситуация оказывается много более сложной. Взаимное расположение двух или нескольких фрагментов изображения обычно определяется указаниями типа «над» или «под», «близко», или «далеко», «внутри» или «снаружи» и т. д. С формальной точки зрения эти характеристики обратимы, антисимметричны и транзитивны. Другими словами, они вводят некоторое полуупорядочение в пространство выражений, содержащее множество A в качестве собственного подмножества. В частном случае, если задано правило упорядочения и вводимый им порядок линеен, определено простое сочленение, и в результате мы имеем линейный язык.

В грамматике плоскостных языков обычно можно выделить несколько уровней. На нижнем определяются цвета точек, заданных на плоскости, и интенсивность выраженности цвета; этот уровень соответствует фонетическому или семиотическому. Уровень, на котором дается описание основных графических символов, соответствует морфологическому уровню структурной лингвистики. Высшие уровни теории плоскостных языков посвящены принципам составления комбинации графических символов; этот уровень соответствует синтаксическому уровню грамматики. Практическое изучение структурных грамматик плоскостных языков может способствовать пополнению общей теории экспериментальными данными, которые должны быть учтены в процессе ее развития.

Одной из основных теоретических проблем является эквивалентность плоскостных и линейных языков. Поскольку любое изображение можно с произвольной точностью представить последовательностью двоичных символов, а, с другой стороны, любое двоичное сообщение можно записать графически, потенциальная эквивалентность линейных и плоскостных языков не может вызывать сомнений. Существует еще, однако, проблема отыскания линейного представления для данного плоскостного языка, причем оптимальным в некотором смысле образом. Типичным примером проблемы эквивалентности является задача формального описания форм и размеров деталей и их множеств при автоматизации процесса технического проектирования. До сих пор удовлетворительные результаты были в этой области получены лишь для формального описания простейших деталей — плоских и тел вращения. Типичными примерами резуль-

татов таких работ служат проблемно-ориентированные языки машинного проектирования типа APT, ADAPT (США), CLAM, PROFILEDATA, COCOMAT (Великобритания), ГЕОМЕТР 66 (СССР). Очевидно, что требования к точности не позволяют воспользоваться в этом случае дискретной точечной аппроксимацией геометрической формы, аналогично использованной в обсуждавшихся выше языках обработки изображений. Геометрические языки скорее должны основываться на выделении основных геометрических элементов: точек, прямых линий, плоскостей, треугольников, прямоугольников, окружностей, сфер и т. п. — в евклидовом пространстве, сформированном заданной системой координат. Более сложные геометрические формы можно получить из простейших с помощью теоретико-множественных операций. Универсальный геометрический язык должен содержать новые типы переменных — геометрические, — кроме целых, реальных и булевых. Геометрические переменные могут принимать значения «пусто» либо задаваться с помощью набора неравенств аналитической геометрии. Например, геометрическую переменную «треугольник» можно задать следующими неравенствами:

$$ax + by + c < 0,$$
$$dx + ey + f < 0,$$
$$gx + hy + i < 0,$$

где a, b, c, d, e, f, g, h, i — некоторые действительные коэффициенты, удовлетворяющие ряду дополнительных условий. В результате оказывается возможным вызвать подпрограмму, обеспечивающую проверку принадлежности данной точки z данному «треугольнику», наличия непустой общей части у двух геометрических объектов и т. д. Следовательно, решение задачи сводится к численной процедуре, полностью поддающейся алгоритмизации.

Геометрический язык, однако, должен обладать и другими возможностями. Может возникнуть необходимость проверки некоторых топологических отношений: включен ли данный геометрический объект полностью в другой либо они соприкасаются? Расположены ли заданные геометрические объекты по одну сторону кривой или поверхности? Может возникнуть также необходимость представления геометрических объектов произвольной формы. Возможность осуществления стандартных геометрических преобразований типа сдвига или поворота геометрических объектов относительно некоторой системы координат или относительно друг друга, построения новых геометрических объектов, изменения размеров при выполнении ряда геометрических, топологических или другого рода ограничений поможет

существенно облегчить работу проектировщика. Наличие столь мощного инструмента может привести к ограничению или полному прекращению автоматического распознавания и обработки некоторых видов информации, представляемой в плоскостном виде. В частности, технологическая информация может полностью обрабатываться в линейно закодированном виде вплоть до момента вывода на печать для получения стандартной технологической документации; графическая же информация будет использоваться человеком только для осуществления контроля.

Другими словами, системы распознавания образов и обработки изображений следует рассматривать как системы обработки информации, работающие с информацией любой природы и любых типов.

ПЕРЕЧЕНЬ ОСНОВНЫХ ОБОЗНАЧЕНИЙ

$\cup$ — сумма множеств или дизъюнкция отношений,
$\cap$ — пересечение множеств или конъюнкция отношений,
$\dot{-}$ — разность множеств или отношений,
$\in$ — принадлежность элемента,
$\subset$ — включение подмножества или подотношения,
$\times$ — прямое произведение пары множеств или отношений,
$\mathbf{\times}$ — прямое произведение системы множеств или отношений,
$\bar{}$ — дополнение к отношению,
$\vee$ — логическая дизъюнкция,
$\wedge$ — логическая конъюнкция,
$\neg$ — логическое отрицание,
$\Rightarrow$ — логическая импликация,
$\Leftrightarrow$ — логическая эквивалентность (двусторонняя импликация),
$\langle\ \rangle$ — линейно упорядоченное множество или набор,
$\{\ \}$ — неупорядоченное множество,
$\{S\}_i$ — i-е множество входных сообщений (i-й образ),
s — реализация входного сообщения,
$\{Z\}$ — множество принятых сигналов,
z — реализация принятого сигнала — множество решений,
$\{Y\}$ — множество решений,
y — решение,
N, M — целые числа, обозначающие размерность входных сигналов,
R — отношение,
$R_{\langle Z' \rangle}$ — проекция отношения на подсемейство множеств $\langle Z' \rangle$,
$\varkappa$ — мощность множества,
U — множество значений входного сигнала, описывающих состояние сетчатки,

U_i — множество значений составляющих i-го входного сигнала,
$\rho(z^*, z)$ — расстояние между векторами z^* и z,
θ — пустое отношение,
$Q_{\langle U \rangle}$ — тривиальное отношение, заданное на семействе множеств $\langle U \rangle$.

СПИСОК ЛИТЕРАТУРЫ

А. Общие вопросы

1. Айзерман М. А., Браверман Э. М., Розоноэр Л. И., Теоретические основы метода потенциальных функций в задаче об обучении автоматов распределению входных ситуаций на классы, *Автоматика и телемеханика*, **25**, № 6 (1964), стр. 917—936.
2. Автоматическое чтение текста, сб. статей под ред. А. И. Михайлова и др., ВИНИТИ, М., 1967.
3. Браверман Э. М., О методе потенциальных функций, *Автоматика и телемеханика*, **26**, № 12 (1965), стр. 2205—2213.
4. Читающие автоматы и распознавание образцов, сб. статей, отв. ред. В. А. Ковалевский, «Наукова думка», Киев, 1965.
5. Читающие устройства, сб. докладов на Конференции по обработке информации, машинному переводу и автоматическому чтению текста, АН СССР, Ин-т научной информации, М., 1962.
6. Fu K. S., Learning control systems. Computer and information sciences, Washington, 1964.
7. Kulikowski J. L., Cybernetyczne układy rozpoznające — Cybernetical recognition systems, in Polish, PWN, Warzawa, to be printed.
8. Опознание образов, сб. статей под ред. д. т. н., проф. И. Т. Турбовича, «Наука», М., 1968.
9. Principles of self-organization, Trans. of the University of Illinois Symp. on Self-Organization, Pergamon Press, 1962.
10. Самонастраивающиеся системы. Распознавание образов. Релейные устройства и конечные автоматы, Труды III Всесоюзного совещания по автоматическому управлению (технической кибернетике), Одесса, 1965, «Наука», 1967.
11. Себестиан Г. С., Процессы принятия решений при распознавании образов, «Техніка», Киев, 1965.
12. Труды III Всесоюзной конференции по информационно-поисковым системам и автоматической обработке научно-технической информации, в 4-х т., под общ. ред. д. т. н., проф. А. И. Михайлова и др., т. 3. Автоматические читающие устройства, ВИНИТИ, М., 1967.

Б. Алгебраические и структурные методы

13. Автоматический анализ сложных изображений, сб. переводов под ред. Э. М. Бравермана, «Мир», М., 1969.
14. Бридинг К. Дж., Язык для описания изображений, в сб. «Автоматический анализ сложных изображений» под ред. Э. М. Бравермана, «Мир», М., 1969, стр. 65—86.
15. Фришкопф Л. С., Хармон Л. Д., Машинное чтение слитного рукописного текста, в сб. «Автоматический анализ сложных изображений» под ред. Э. М. Бравермана, «Мир», М., 1969, стр. 171—187.
16. Файн В. С., Опознавание изображений — основы непрерывно-групповой теории и ее приложения, «Наука», М., 1970.
17. Kulikowski J. L., Niektore problemy strukturalnej analizy obrazov złożonych — Some problems of structural analysis of composite patterns, in Polish, *Archivum Automatyki i Telemechaniki*, **XV**, No. 3 (1970).

18. Kulpa Z., Szydlo Jezyk, PICTURE ALGOL 1204 A. Opis użytkovy — The language PICTURE ALGOL 1204 A. A use manual, in Polish Institute of Automation Polish Academy of Sciences, a report, Warsaw 1971.
19. Lipkin B. S. (ed.), Rosenfeld A., Picture processing and psychopictorics, Acad. Press, New Y. — Lond., 1970.
20. Маркус С., Теоретико-множественные модели языков, «Наука», М., 1970.
21. McCormick B. H., The Illinois pattern recognition computer, (Illiac III) Digital Computer Lab. Un. of Illinois, Rep. No. 148, 1963.
22. McCormick B. H., Illiac III computer system. Brief description and annotated bibliography, Dept. of Comp. Science Un. of Illinois, File No. 841, June 1970.
23. McCormick B. H., Schwebel J. C., Properties of a discrete space preserved by image processing relations, Dept. of Computer Science Univ. of Illinois File No. 769, July 1968.
24. Нарасимхан Р., Лингвистический подход к распознаванию образов, в сб. «Автоматический анализ сложных изображений», «Мир», М., 1969, стр. 22—49.
25. Нарасимхан Р., Программа сканирования фотографий пузырьковой камеры BUBBLE SCAN I, в сб. «Автоматический анализ сложных изображений», «Мир», М., 1969, стр. 111—129.
26. Нарасимхан Р., Метод маркировки и синтаксическое описание изображений, в сб. «Автоматический анализ сложных изображений», «Мир», М., 1969, стр. 87—110.
27. Нарасимхан Р., Синтаксическая интерпретация классов изображений, в сб. «Автоматический анализ сложных изображений», «Мир», М., 1969, стр. 50—64.
28. Нарасимхан Р., Дж. Р. Уитскен, Х. Джонсон, BUBBLE TALK. Структура программы, осуществляющей непосредственный обмен информацией между оператором и системой ILLIAC III, в сб. «Автоматический анализ сложных изображений», «Мир», М., 1969, стр. 130—170.
29. Поляков В. Г., О цифровой технике считывания и анализа контуров, в сб. «Опознание образов. Теория передачи информации», «Наука», М., 1965, стр. 44—71.
30. Read J., «SHOW-and-TELL. An Interactive programming system for image processing». Dept. of Computer Science University of Illinois Report No. 429, February 1971.
31. Rogers J., Tanimoto T., A computer program for classifying plants, *Science*, **132**, 1960.
32. Schwebel J. C., Use of graph transformations to characterize an image: an illustrative example, Dept. of Computer Science Univ. of Illinois, File No. 770, July 1968.
33. Шерман Г., Квазитопологический метод распознавания линейных изображений, в сб. «Автоматический анализ сложных изображений», «Мир», М., 1969, стр. 11—20.
34 Staniszkis M., Porownywanie struktur punktowych — Comparison of point struktures, in Polish, Institute of Automation Polish Academy of Sciences (a Report), Warsaw, 1971.
35. Структурные методы опознавания и автоматическое чтение, сборник статей под общей ред. проф. А. И. Михайлова, ВИНИТИ, М., 1970.

СОДЕРЖАНИЕ

Математические вопросы

Теория распознавания

УВАЖАЕМЫЙ ЧИТАТЕЛЬ!

Ваши замечания о содержании книги, ее оформлении, качестве перевода и другие просим присылать по адресу: 129820, Москва, И-110, ГСП, 1-й Рижский пер., д. 2, издательство «Мир».

ИБ № 947

КИБЕРНЕТИЧЕСКИЙ СБОРНИК № 14

Редактор И. Маховая, Л. Штейнпресс
Художественный редактор В. Шаповалов
Технический редактор И. Кренделева
Корректор А. Шехтер

Сдано в набор 19.11.76. Подписано к печати 7.06.77.
Бумага кн.-журн. 60×90$^{1}/_{16}$=7,25 бум. л. Усл. печ. л. 14,50. Уч.-изд. л. 13,29. Изд. № 1/9137. Цена 1 р. 80 к.
Заказ № 420

Издательство «Мир»
Москва, 1-й Рижский пер., 2

Ордена Трудового Красного Знамени
Ленинградская типография № 2
имени Евгении Соколовой Союзполиграфпрома
при Государственном комитете Совета Министров
СССР по делам издательств, полиграфии
и книжной торговли. 198052, Ленинград,
Л-52, Измайловский проспект, 29