МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

ПРИКЛАДНАЯ ДИСКРЕТНАЯ МАТЕМАТИКА

Научный журнал

2012 №2(16)

Свидетельство о регистрации: ПИ №ФС 77-33762
от 16 октября 2008 г.

ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

Адрес редакции: 634050, г. Томск, пр. Ленина, 36
E-mail: vestnik_pdm@mail.tsu.ru

В журнале публикуются результаты фундаментальных и прикладных научных исследований отечественных и зарубежных ученых, включая студентов и аспирантов, в области дискретной математики и её приложений в криптографии, компьютерной безопасности, кибернетике, информатике, программировании, теории надежности, интеллектуальных системах.

Периодичность выхода журнала: 4 номера в год.

Редактор *Н. И. Шидловская*
Верстка *И. А. Панкратовой*

Подписано к печати 21.05.2012.
Формат $60 \times 84\frac{1}{8}$. Усл. п. л. 13,6. Уч.-изд. л. 15,2. Тираж 300 экз.

Издательство ТГУ. 634029, Томск, ул. Никитина, 4
Отпечатано в типографии ТПУ.

СОДЕРЖАНИЕ

CONTENTS

ТЕОРЕТИЧЕСКИЕ ОСНОВЫ ПРИКЛАДНОЙ ДИСКРЕТНОЙ МАТЕМАТИКИ

DOI 10.17223/20710410/16/1

УДК 519.6

О ПРИМИТИВНЫХ НАБОРАХ НАТУРАЛЬНЫХ ЧИСЕЛ

С. Н. Кяжин, В. М. Фомичев

Национальный исследовательский ядерный университет МИФИ, г. Москва, Россия

E-mail: litota975@mephist.ru, fomichev@nm.ru

Описано строение множества примитивных наборов натуральных чисел и установлены их основные свойства. С использованием понятий тупиковости и k-минимальности построен алгоритм перечисления примитивных наборов чисел, не превышающих заданного числа m. Предложены алгоритмы определения показателя примитивности ориентированного конечного графа с помощью поиска в глубину на графе и возведения в степень матрицы смежности вершин и оценена их вычислительная сложность.

Ключевые слова: *примитивный набор натуральных чисел, примитивный граф, примитивная матрица, экспонент, субэкспонент.*

Введение

Матрица M называется *положительной* (*неотрицательной*), если положительны (неотрицательны) все её элементы; этот факт обозначается $M > 0$ ($M \geqslant 0$). Неотрицательная квадратная *матрица* называется *примитивной*, если $M^t > 0$ при некотором натуральном t, а наименьшее натуральное γ, при котором $M^\gamma > 0$, называется *экспонентом* (*показателем примитивности*) *матрицы* M и обозначается $\exp M$. Если такого t не существует, то $\exp M = \infty$. Если для неотрицательной матрицы M выполнено соотношение $M + M^2 + \ldots + M^t > 0$, то наименьшее такое t называется *субэкспонентом матрицы* M.

Сильносвязный *орграф* Γ называется *примитивным*, если при некотором натуральном m для любой пары вершин (i, j) в Γ существует путь из i в j длины m. Наименьшее такое m называется *экспонентом графа* Γ. Под *субэкспонентом орграфа* Γ понимается субэкспонент матрицы смежности его вершин. Субэкспонент сильносвязного n-вершинного графа не превышает n (он равен диаметру графа).

Одной из важных задач при изучении перемешивающих свойств преобразований является определение экспонентов перемешивающих матриц и соответственно перемешивающих графов [1, 2]. При исследовании экспонентов матриц и графов часто используется эпиморфизм мультипликативного моноида неотрицательных матриц порядка n на моноид n-вершинных орграфов, где умножение орграфов определено как умножение бинарных отношений [3]. Матрица положительна, если и только если соответствующий граф полный. Отсюда следует, что орграф и его матрица смежности M одновременно примитивны или не примитивны, в случае примитивности экспоненты их равны.

Критерий примитивности орграфа определяется длинами его простых контуров [2] (контур простой, если проходит через любую вершину не более одного раза). Если

$C_1, \ldots, C_k$ суть все простые контуры орграфа длин $l_1, \ldots, l_k$ соответственно, то орграф примитивный, если и только если числа $l_1, \ldots, l_k$ взаимно просты.

Набор взаимно простых в совокупности натуральных чисел $A = (a_1, \ldots, a_k)$, где $1 < a_1 < \ldots < a_k$, называется *примитивным*. Заметим, что множество всех примитивных наборов натуральных чисел совпадает с областью определения функции Фробениуса $f(a_1, \ldots, a_k)$ [4].

Распознавание примитивности орграфа может быть выполнено как определение длин всех его простых контуров и распознавание примитивности набора всех различных длин этих контуров. Другой подход заключается в определении показателя примитивности графа с помощью возведения в степень матрицы смежности его вершин.

Распознавание примитивности набора чисел $(a_1, \ldots, a_k)$ можно выполнить, применив $k - 1$ раз алгоритм Евклида к элементам набора. Можно также воспользоваться заранее составленными таблицами примитивных наборов.

В п. 1 работы описана связь экспонента и субэкспонента матрицы и ее полугрупповых характеристик, в п. 2 представлены основные свойства примитивных наборов натуральных чисел, в п. 3 сформулированы и доказаны критерии тупиковости и k-минимальности наборов, в п. 4 описан алгоритм перечисления k-минимальных тупиковых примитивных наборов, в п. 5 оценена вычислительная сложность определения длин простых циклов графа с помощью поиска в глубину, а в п. 6 — вычислительная сложность определения экспонента матрицы (и соответствующего графа) с помощью возведения матрицы в степень.

1. О связи экспонента и субэкспонента матрицы с её полугрупповыми характеристиками

Обозначим через $A = (a_{ij})$ матрицу смежности вершин орграфа Γ и через $A^t = \left(a_{ij}^{(t)}\right)$ — степень матрицы A. *Периодом вершины орграфа* называется наибольший общий делитель (НОД) всех длин контуров, содержащих данную вершину. Заметим, что при подсчёте достаточно учитывать только простые контуры, так как любой контур представляет собой соединение нескольких простых контуров и его длина кратна НОД длин всех составляющих его простых контуров. Если орграф сильно связен (матрица A является неразложимой [2, гл. VI, § 2]), то все его вершины имеют одинаковый период, называемый также *к-периодом матрицы* A (орграфа Γ).

Носителем неотрицательной матрицы $B = (b_{ij})$ называется 0,1-матрица $\nu(B) = (\nu b_{ij})$, где

$$\nu b_{ij} = \begin{cases} 1, \text{ если } b_{ij} > 0, \\ 0, \text{ если } b_{ij} = 0. \end{cases}$$

Множество 0,1-матриц размера $n \times n$ образует полугруппу G_n относительно операции $*$, где $A * B = \nu(AB)$.

Утверждение 1 [2, гл. VI, теорема 2.2]. Пусть к-период сильносвязного орграфа Γ равен $q > 0$. Тогда для любых вершин i, j существует единственное число $r(i, j)$, $0 \leqslant r(i, j) < q$, такое, что

1) если $a_{ij}^{(t)} > 0$, то $t \equiv r(i, j) (\mathrm{mod}\ q)$;

2) существует число $k(i, j) > 0$, такое, что $a_{ij}^{kq+r} > 0$ для любого $k \geqslant k(i, j)$.

Следовательно, множество вершин V графа Γ разбивается на q блоков:

$$V = C_0 \cup \ldots \cup C_{q-1},$$

при этом имеется единственная упорядоченная последовательность этих блоков со свойством: если (i,j) — дуга графа Γ и $i \in C_s$, где $s \in \{0,\dots,q-1\}$, то $j \in C_{(s+1)\bmod q}$. Отсюда $A^{kq+r} + A^{kq+r+1} + \dots + A^{kq+r+q-1} > 0$ для любого $k \geqslant \max\limits_{i,j}\{k(i,j)\}$.

Утверждение 2. Пусть неразложимая матрица $A = (a_{ij})$ с к-периодом q имеет тип (d,τ) в полугруппе G_n, где d — циклическая глубина и τ — период матрицы A. Тогда

1) $\tau = q$;
2) если матрица A примитивная, то $d = \exp A$ и $\tau = 1$;
3) если A не примитивная, то $\tau > 1$ и субэкспонент матрицы A не превышает $d + \tau - 1$.

Доказательство. Из определения типа матрицы следует

$$A^d = A^{d+\tau}. \tag{1}$$

Если матрица A примитивная, то $q = 1$ в силу критерия примитивности сильносвязного орграфа. Пусть $\gamma = \exp A$, то есть $A^\gamma > 0$, тогда в полугруппе G_n выполнены условия $A^\gamma = A^{\gamma+1}$ и $A^\gamma \neq A^{\gamma-1}$. Следовательно, (1) выполняется при $d = \gamma, \tau = 1$. Субэкспонент матрицы не превышает γ.

Если A не примитивная, то $q > 1$. Из утверждения 1 следует, что в полугруппе G_n имеет место $A^{kq+r} = A^{kq+r+q}$ для любого $k \geqslant \max\limits_{i,j}\{k(i,j)\}$ и $A^{kq+r} \neq A^{kq+r-q}$ при $k < \max\limits_{i,j}\{k(i,j)\}$. Следовательно, (1) выполняется при $\tau = q$ и $d \geqslant k(i,j)q + r$. Из утверждения 1 также следует, что $A^{kq+r} + A^{kq+r+1} + \dots + A^{kq+r+q-1} > 0$ для любого $k \geqslant \max\limits_{i,j}\{k(i,j)\}$. Значит, субэкспонент матрицы A не превышает $d + \tau - 1$. ■

2. Определяющие свойства примитивных наборов натуральных чисел

Исследуем свойства примитивных наборов.

Утверждение 3. Если A — примитивный набор чисел, то примитивен любой набор, полученный из A добавлением любого натурального числа или (при $|A| > 1$) удалением числа a, кратного одному из остальных чисел набора.

Следствие 1. Набор примитивен, если содержит пару взаимно простых чисел.

Определение 1. *Набор* из N^k назовем *приведённым*, если в нём любое число не кратно любому другому числу.

Иначе говоря, приведённый набор $(a_1,\dots,a_k)$ является антицепью в решётке натуральных делителей числа $k_A = \operatorname{lcm}(a_1,\dots,a_k)$, обозначаемой $D(k_A)$, где $\operatorname{lcm}(a_1, \dots, a_k)$ — наименьшее общее кратное чисел $a_1,\dots,a_k$.

Любому примитивному набору A размера $k \geqslant 1$ однозначно соответствует приведённый набор $\pi(A)$ размера l, где $1 \leqslant l \leqslant k$: если в наборе A число a_j кратно a_i, то набор $\pi(A)$ не содержит a_j.

Определение 2. *Примитивный набор* A размера $k \geqslant 1$ назовем *тупиковым*, если $A = (1)$ или при $k > 1$ удаление из набора любого элемента нарушает его примитивность.

Все примитивные наборы длины 2, не содержащие 1, являются тупиковыми, так как при удалении одного из элементов набора остается число, отличное от 1. В соответствии с утверждением 3 примитивный тупиковый набор является приведённым набором.

Определение 3. *Примитивный набор* A размера $k > 1$ назовем *r-примитивным*, где $0 \leqslant r \leqslant k-1$, если после удаления из A любого подмножества порядка r примитивность получившегося набора сохраняется.

Тупиковый набор 0-примитивен, но не 1-примитивен.

Далее полагаем, что $A = (a_1, \ldots, a_k) \in N^k$ — приведённый набор, элементы которого не превышают числа m. Пусть 2^A — булеан множества $\{a_1, \ldots, a_k\}$; $P(A, r)$ — множество всех примитивных наборов (упорядоченных по возрастанию) размера (порядка) r, $P(A) = \bigcup\limits_{r \leqslant k} P(A, r)$. Заметим, что если A — примитивный приведённый набор, то все наборы из $P(A)$ также приведённые.

На множестве $P(A)$ определим отношение частичного порядка: $(b_1, \ldots, b_l) \leqslant (a_1, \ldots, a_k)$, если и только если $l \leqslant k$ и найдется бесповторная упорядоченная выборка $(i_1, \ldots, i_l)$ из $(1, \ldots, k)$, такая, что $i_1 < \ldots < i_l$ и b_j делит a_{i_j}, $j = 1, \ldots, l$.

Определение 4. *Тупиковый набор* $B \in P(A)$ назовем *минимальным* в $P(A)$, если не существует другого набора $B' \in P(A)$, такого, что $B' \leqslant B$.

Для любого набора B из $P(A)$ имеется хотя бы один минимальный тупиковый набор $\Theta(B)$, такой, что $\Theta(B) \leqslant B$.

Определение 5. *Тупиковый набор* $B \in P(A)$ назовем *r-минимальным* в $P(A)$, если не существует другого набора $B' \in P(A)$ размера r, такого, что $B' \leqslant B$.

Утверждение 4. Если A — примитивный приведённый набор, то $\langle P(A), \leqslant \rangle$ — верхняя полурешётка, в которой максимальный элемент есть A и любой минимальный элемент — это тупиковый минимальный набор.

Доказательство. Если наборы $A_1, A_2 \in P(A)$ имеют размеры соответственно l и r, то их верхняя грань $\sup\{A_1, A_2\}$ определена как набор размера t упорядоченных по возрастанию элементов множества $A_1 \cup A_2$, где $\max\{l, r\} \leqslant t \leqslant r + l$. В соответствии с утверждением 3, $\sup\{A_1, A_2\}$ также есть примитивный набор из $P(A)$, то есть $\langle P(A), \leqslant \rangle$ — верхняя полурешётка.

Утверждения о максимальном и минимальных элементах полурешётки вытекают из определений множества $P(A)$ и тупикового минимального набора. ■

Для набора A рассмотрим наибольший общий делитель как функцию, определённую на 2^A. При $B = \{a_{i_1}, \ldots, a_{i_l}\} \in 2^A$ обозначим $\gcd(B) = \gcd(a_{i_1}, \ldots, a_{i_l})$, если $B \neq \varnothing$, и $\gcd(\varnothing) = \operatorname{lcm}(a_1, \ldots, a_k)$, где $\gcd(a_{i_1}, \ldots, a_{i_l})$ — наибольший общий делитель чисел a_{i_1}, …, a_{i_l}; $D(A) = \{\gcd(B) : B \in 2^A\}$. Множество $D(A)$ частично упорядочено по отношению делимости: $\gcd(B) \leqslant \gcd(B')$ для $B, B' \in 2^A$, если и только если $\gcd(B)$ делит $\gcd(B')$.

Утверждение 5. Если A — примитивный тупиковый набор, то $D(A)$ — решётка, антиизоморфная решетке 2^A.

Доказательство. Установим биективность функции gcd: $2^A \to D(A)$, для этого достаточно убедиться в её инъективности.

Предположим, что функция gcd не инъективна, то есть найдутся множества $B_1, B_2 \in 2^A$, $B_1 \neq B_2$, такие, что $\gcd(B_1) = \gcd(B_2) = d$. Заметим, что $\gcd(B_1 \cup B_2)$ делит d в соответствии с определением функции gcd. Вместе с тем d делит каждое из чисел множества $B_1 \cup B_2$. Значит, $\gcd(B_1 \cup B_2) = d$.

Так как $B_1 \neq B_2$, то одно из этих множеств не включено в другое, пусть для определённости $B_2 \setminus B_1 \neq \varnothing$. Обозначим $B_3 = A \setminus (B_1 \cup B_2)$. В соответствии с определением

функции gcd имеем цепочку равенств

$$1 = \gcd(A) = \gcd(B_1 \cup B_2 \cup B_3) = \gcd(\gcd(B_1 \cup B_2), B_3) = \\ = \gcd(d, B_3) = \gcd(\gcd(B_1), B_3) = \gcd(B_1 \cup B_3).$$

Следовательно, множество $B_1 \cup B_3$ примитивное, где $B_1 \cup B_3 \subset A$, так как $B_2 \setminus B_1 \neq \varnothing$. Отсюда получаем противоречие с тупиковостью набора A.

В соответствии с определением функции gcd если $B \subset B'$, то $\gcd(B')$ делит $\gcd(B)$, значит, биекция $2^A \leftrightarrow D(A)$ антиизотонна. ■

3. Критерии тупиковости и k-минимальности наборов натуральных чисел

Обозначим через A_i коатомы решётки 2^A и через μ_i — атомы решётки $D(A)$: $A_i = \{a_1, \ldots, a_k\} \setminus \{a_i\}$, $\mu_i = \gcd(A_i)$, $i = 1, \ldots, k$.

Теорема 1. Набор A примитивный тупиковый, если и только если $(\mu_1, \ldots, \mu_k)$ — набор попарно взаимно простых чисел, отличных от 1. При этом

$$a_i = \frac{c_i \mu_1 \cdot \ldots \cdot \mu_k}{\mu_i}, \tag{2}$$

где $(c_1, \ldots, c_k)$ есть 1-примитивный набор натуральных чисел и $\gcd(c_i, \mu_i) = 1$ для $i = 1, \ldots, k$.

Доказательство. Пусть набор A примитивный тупиковый. Если $\mu_i = 1$ при некотором $i \in \{1, \ldots, k\}$, то множество A_i примитивное, что противоречит тупиковости набора A. Если $\gcd(\mu_i, \mu_j) = d > 1$ при $i \neq j$, то d делит все числа множеств A_i и A_j, значит, d делит все числа набора A, что противоречит его примитивности.

Докажем достаточность. Если набор A не примитивный, то $\gcd(A) = d > 1$. Отсюда d делит μ_i при любом $i = 1, \ldots, k$, значит, числа $\mu_1, \ldots, \mu_k$ не являются попарно взаимно простыми, то есть имеем противоречие. Если набор A не тупиковый, то $\mu_i = 1$ при некотором $i \in \{1, \ldots, k\}$, что противоречит условию.

По определению чисел $\mu_1, \ldots, \mu_k$ число a_i делится на каждое из чисел множества $\{\mu_1, \ldots, \mu_k\} \setminus \{\mu_i\}$, следовательно, для $i = 1, \ldots, k$ верно (2), где $(c_1, \ldots, c_k) \in N^k$.

Заметим, что набор $(c_1, \ldots, c_k)$ примитивный, так как иначе набор A не примитивный в силу (2). Если $\gcd(c_i, \mu_i) = d > 1$ при некотором $i \in \{1, \ldots, k\}$, то d делит все числа набора A в соответствии с (2) и с определением чисел $\mu_1, \ldots, \mu_k$, что противоречит примитивности набора A. Следовательно, $\gcd(c_i, \mu_i) = 1$ при любом $i = 1, \ldots, k$. Из (2) и определения чисел $\mu_1, \ldots, \mu_k$ следует также, что

$$\mu_i = \gcd\left(\{c_1\mu_2 \cdot \ldots \cdot \mu_k, \ldots, c_k\mu_1 \cdot \ldots \cdot \mu_{k-1}\} \setminus \{(c_i\mu_1 \cdot \ldots \cdot \mu_k)/\mu_i\}\right).$$

Значит, $\gcd(\{c_1, \ldots, c_k\} \setminus \{c_i\}) = 1$ для $i = 1, \ldots, k$, и $(c_1, \ldots, c_k)$ есть 1-примитивный набор. ■

Следствие 2. Пусть $B = \{a_{i_1}, \ldots, a_{i_l}\} \in 2^A$ и $\overline{B} = A \setminus B$, тогда

$$\gcd(B) = \gcd(\{c_{i_1}, \ldots, c_{i_l}\}) \prod_{j \in \overline{B}} \mu_j.$$

Доказательство. Если $c_1, \ldots, c_k, x_1, \ldots, x_k \in N$, то $\gcd(c_1x_1, \ldots, c_kx_k)$ делится на $\gcd(\{c_1, \ldots, c_k\})\gcd(x_1, \ldots, x_k)$. Отсюда, положив $x_i = (\mu_1 \cdot \ldots \cdot \mu_k)/\mu_i$, $i = 1, \ldots, k$, в соответствии с (2) получаем, что $\gcd(B)$ делится на $\gcd(\{c_{i_1}, \ldots, c_{i_l}\}) \times$ $\times\gcd(\{x_{i_1}, \ldots, x_{i_l}\})$, где из теоремы 1 следует, что $\gcd(\{x_{i_1}, \ldots, x_{i_l}\}) = \prod_{j \in \overline{B}} \mu_j$.

Без ущерба для общности положим $B = \{a_1, \ldots, a_l\}$, где $1 \leqslant l \leqslant k$, и обозначим $c' = \gcd(c_1, \ldots, c_l)$, $x' = \gcd(x_1, \ldots, x_l)$. В этих условиях множества $C' = \{c_1/c', \ldots, c_l/c'\}$ и $X' = \{x_1/x', \ldots, x_l/x'\}$ примитивные.

Пусть $\gcd(B) = d \cdot c' \cdot x'$ при натуральном $d > 1$, тогда $\gcd(B') = d$, где $B' = \{a_1/c' \cdot x', \ldots, a_l/c' \cdot x'\}$. Следовательно, d делит $c_r x_r / c' x'$ при $r = 1, \ldots, l$, отсюда

$$d = d(c_r)d(x_r), \tag{3}$$

где $d(c_r)$ делит c_r/c' и $d(x_r)$ делит x_r/x'. В силу примитивности множества C' найдется номер $r \in \{1, \ldots, l\}$, такой, что $d(x_r) > 1$. Тогда, учитывая, что $x_i/x' = (\mu_1 \cdot \ldots \cdot \mu_l)/\mu_i$, $i = 1, \ldots, l$, и числа $\mu_1, \ldots, \mu_l$ попарно взаимно простые, найдется номер $j \in \{1, \ldots, l\}$, такой, что $\gcd(d(x_r), \mu_j) = d_{rj} > 1$, значит, d_{rj} делит d. Из того, что $\gcd(x_j/x', \mu_j) = 1$, следует $\gcd(x_j/x', d_{rj}) = 1$, отсюда в силу (3) d_{rj} делит $d(c_j)$, поэтому d_{rj} делит cc_j/c'.

Вместе с тем в соответствии с теоремой 1 $\gcd(c_j, \mu_j) = 1$, тогда $\gcd(d(c_j), d_{rj}) = 1$. Имеем противоречие. Следовательно, $d = 1$. ■

Представление целого числа n произведением степеней простых чисел $n = \varepsilon \cdot p_1^{k_1} \cdot \ldots \cdot p_s^{k_s}$, где $\varepsilon = \pm 1$, $k_i > 0$ — кратность числа p_i, называется *каноническим разложением числа* n, при этом множество чисел $\{p_1, \ldots, p_s\}$ называется *факторной базой числа* n и обозначается $F(n)$.

Определение 6. *Факторной базой набора* $A = (a_1, \ldots, a_k)$ назовем множество чисел $F(A) = F(a_1) \cup \ldots \cup F(a_k)$.

Докажем критерий k-минимальности тупикового примитивного набора.

Следствие 3. Примитивный тупиковый набор A является k-минимальным, если и только если μ_1, …, μ_k — простые числа и $c_i = 1$, $i = 1, \ldots, k$.

Доказательство. Пусть A есть k-минимальный набор и каноническое разложение μ_i при некотором $i \in \{1, \ldots, k\}$ имеет вид $\mu_i = p_1^{k_1} \cdot \ldots \cdot p_s^{k_s}$. Рассмотрим набор B, состоящий из чисел $\mu_1, \ldots, \mu_{i-1}, \mu_{i+1}, \ldots, \mu_k, \mu_i' = p_1^{k_1} \cdot \ldots \cdot p_s^{k_s - 1}$. Согласно (2), $B \leqslant A$, причём его размер равен k. Тогда A не является k-минимальным. Если $c_i > 1$ при некотором $i \in \{1, \ldots, k\}$, то существует набор B', которому соответствуют $c_1, \ldots, c_{i-1}, c_{i+1}, \ldots, c_k, c_i' = 1$. Согласно (2), $B' \leqslant A$, что противоречит k-минимальности A. Следовательно, μ_1, …, μ_k — простые числа и $c_i = 1$, $i = 1, \ldots, k$.

Если набор A не является k-минимальным, то существует набор $A' = (a_1', \ldots, a_k')$, такой, что $A' \leqslant A$. Согласно (2), $a_i' = (c_i' \mu_1' \cdot \ldots \cdot \mu_k')/\mu_i'$, $i = 1, \ldots, k$, причём c_i' делит c_i или μ_j' делит μ_j при некоторых $i, j \in \{1, \ldots, k\}$. Тогда $c_i > 1$ или μ_j составное соответственно. ■

Следствие 4. Факторной базой k-минимального тупикового набора является множество $\{\mu_1, \ldots, \mu_k\}$.

Примеры k-минимальных тупиковых примитивных наборов:

1) 3-минимальные наборы $A = (6, 10, 15)$, $F(A) = \{2, 3, 5\}$; $B = (10, 14, 35)$, $F(B) = \{2, 5, 7\}$;
2) 4-минимальный набор $C = (30, 42, 70, 105)$, $F(C) = \{2, 3, 5, 7\}$.

4. Перечисление k-минимальных примитивных наборов натуральных чисел

По утверждению 3 любой примитивный набор A можно получить из соответствующего тупикового набора A' добавлением любого числа. По следствию 3 любой ту-

пиковый набор A' можно получить из соответствующего k-минимального набора A'' умножением элемента набора a_i на число, взаимно простое с μ_i, $i \in \{1, \ldots, k\}$.

Построим алгоритм перечисления множества всех k-минимальных примитивных наборов, состоящих из чисел, не превышающих m (обозначим его PR_m).

Пусть $P(x)$ — множество простых чисел, не больших x. Известно [5], что $\pi(x) = |P(x)| \approx x/\ln x$. Набор размера 2 является 2-минимальным, если и только если он представляет собой пару различных простых чисел. Число таких наборов равно $\pi(m)(\pi(m)-1)/2$. Задача перечисления 2-минимальных примитивных наборов решается, в частности, с использованием «решета Эратосфена».

При $k > 2$ в соответствии с теоремой 1 и следствием 3 k-минимальный тупиковый примитивный набор $A = (a_1, \ldots, a_k)$ состоит из чисел $a_i = (\mu_1 \cdot \ldots \cdot \mu_k)/\mu_i$, где $(\mu_1, \ldots, \mu_k)$ — набор различных простых чисел. Тогда если $\mu_1 < \ldots < \mu_k$, то достаточно перечислить все наборы $(\mu_1, \ldots, \mu_k)$ со свойством $\mu_2 \cdot \ldots \cdot \mu_k \leqslant m$. Заметим, что при данных ограничениях $\mu_2 < m^{\frac{1}{k-1}}$.

Пусть n-е простое число есть p_n. Тогда $p_1 = 2 \leqslant \mu_1$, $p_2 = 3 \leqslant \mu_2$. Для $k > 2$ обозначим $\Psi_k = p_3 \cdot \ldots \cdot p_k$ и положим $\Psi_2 = 1$. Значения Ψ_k для $k = 3, \ldots, 8$ приведены в табл. 1.

Таблица 1

Значения функции Ψ_k

k	3	4	5	6	7	8
Ψ_k	5	35	385	5005	85085	1616615

Для любого подходящего набора $(\mu_1, \ldots, \mu_k)$ из неравенства $\mu_2 \cdot \ldots \cdot \mu_k \leqslant m$ при $k > 2$ следует, что $p_2 \leqslant \mu_2 \leqslant m^{\frac{1}{k-1}}$ и для $s = 3, \ldots, k$

$$p_s \leqslant \mu_s \leqslant \left(\frac{m}{3\Psi_{s-1}}\right)^{\frac{1}{k-s+1}}, \tag{4}$$

где при $s < k$ неравенство строгое.

Алгоритм перечисления при $k > 2$ состоит в следующем. В качестве μ_k перебираем все простые числа в пределах, указанных двусторонним неравенством (4). При $3 \leqslant s < k$ и каждом фиксированном наборе чисел $(\mu_{s+1}, \ldots, \mu_k)$ в качестве μ_s перебираем все простые числа в пределах, указанных в (4). При каждом фиксированном наборе чисел $(\mu_3, \ldots, \mu_k)$ перебираем все простые числа μ_1 и μ_2, где $2 \leqslant \mu_1 < \mu_2 < m^{\frac{1}{k-1}}$.

Оценим вычислительную сложность алгоритма, измеренную числом построенных наборов различных простых чисел $(\mu_1, \ldots, \mu_k)$. Из алгоритма следует, что при $s = 3, \ldots, k$ число различных значений μ_s не больше $\pi\left(\left(m/\left(3\Psi_{s-1}\right)\right)^{\frac{1}{k-s+1}}\right)$ (при $s < k$ строго меньше). Число различных пар (μ_1, μ_2) не больше $\pi(x)(\pi(x)-1)/2$ при $x = m^{\frac{1}{k-1}}$. Тогда общее количество искомых наборов оценивается величиной

$$\frac{\pi\left(m^{\frac{1}{k-1}}\right)\left(\pi\left(m^{\frac{1}{k-1}}\right)-1\right)}{2} \prod_{s=3}^{k} \pi\left(\left(\frac{m}{3\Psi_{s-1}}\right)^{\frac{1}{k-s+1}}\right).$$

При больших m и k эта величина имеет порядок не более

$$\mathrm{O}\left(\left(km^{\frac{2}{k-1}}(m/3)^{H(k-1)}\right) / \left(\ln^2 m \prod_{j=2}^{k-1} \Psi_j\right)\right),$$

где $H(k-1) = 1 + \frac{1}{2} + \frac{1}{3} + \ldots + \frac{1}{k-2}$ — сумма первых $(k-1)$ членов гармонического ряда. Порядок последней величины не превышает $\mathrm{O}\left(m^{\ln k} / \left(\ln^2 m \prod_{j=2}^{k-1} \Psi_j\right)\right)$. При $k > 2$ для оценки величин Ψ_k можно использовать оценку [6]: $p_k > k \ln k$. Тогда $\Psi_k > \frac{k!}{2} \prod_{j=3}^{k} \ln j$.

Пусть теперь числа $a_1, \ldots, a_k$ не ограничены. Тогда наибольшее число в k-минимальном тупиковом примитивном наборе равно $\max(a_1, \ldots, a_k) = \mu_2 \cdot \ldots \cdot \mu_k$. С использованием точных значений простых чисел вычислены достижимые нижние оценки для наибольших чисел в k-минимальных тупиковых примитивных наборах при $k = 3, \ldots, 8$ (табл. 2).

Таблица 2

Числовые границы для наборов длины k

Размер набора k	3	4	5	6	7	8
Нижняя граница $\max(a_1, \ldots, a_k)$	15	105	1155	15015	255255	4849845

5. Определение длин простых циклов с помощью поиска в глубину

Для определения длин простых циклов используем известный алгоритм поиска в глубину [7].

Пусть дан граф $G = (V, E)$, где V — множество вершин графа, а E — множество его неориентированных рёбер либо ориентированных дуг. Опишем алгоритм обхода всех рёбер графа. В качестве начальной выбираем произвольную вершину и двигаемся по рёбрам, пока не встретится тупик (вершина, не имеющая исходящих рёбер, ведущих в непосещённые вершины). После попадания в тупик возвращаемся назад по пройденному пути, пока не встретится вершина, у которой есть исходящие ребра, ведущие в непосещённые вершины, и из неё двигаемся дальше по одному из таких рёбер. Алгоритм обхода рёбер завершает работу, когда встречается начальная вершина и все её соседние вершины уже посещены. Если после этого остаются непосещённые вершины, то повторяется поиск из одной из них в соответствии с вышеописанным алгоритмом. Алгоритм завершается, когда обнаружены все вершины графа.

Для наглядности считаем, что в ходе работы алгоритма вершины графа могут быть белыми, серыми и чёрными. Изначально все вершины помечены белым цветом. Впервые обнаружив вершину v, красим её серым цветом. По окончании обработки всех исходящих рёбер красим вершину v в чёрный цвет. Таким образом, белый цвет соответствует тому, что вершина ещё не обнаружена, серый — что вершина обнаружена, но просмотрены ещё не все исходящие из неё ребра, чёрный — что вершина обнаружена и все исходящие из неё рёбра просмотрены.

Оценим вычислительную сложность реализации алгоритма на однопроцессорной системе. В качестве элементарных операций выберем прохождение ребра графа и сравнение двух чисел. Так как каждое ребро проходится не более одного раза в прямом и обратном направлении, то время работы алгоритма поиска в глубину оценивается величиной $\mathrm{O}(n^2)$.

Модифицируем данный алгоритм поиска в глубину с целью определения длин всех простых циклов. В частности, переходя в вершину u из вершины v по ребру (v, u),

будем запоминать предшественника u, записывая $p[u] = v$. Для вершин, у которых предшественников нет, положим $p[u] = -1$.

Рёбра ориентированного графа можно разделить на несколько категорий в зависимости от их роли при поиске в глубину. *Рёбра деревьев* — это ребра, входящие в лес поиска в глубину. *Обратные рёбра* — это рёбра, соединяющие вершину с её предком в дереве поиска в глубину. *Прямые ребра* — это рёбра, соединяющие вершину с её потомком, но не входящие в лес поиска в глубину. *Перекрёстные ребра* — все остальные. Они могут соединять две вершины из одного дерева поиска в глубину, если ни одна их этих вершин не является предком другой, или же вершины из разных деревьев.

Тип ребра (v, u) можно определить по цвету вершины u в момент, когда ребро проходится в первый раз: белый цвет означает ребро дерева ((v, u) войдёт в лес поиска в глубину); серый (u является предком v) — обратное ребро; чёрный (ни одна из вершин не является предком другой) — прямое или перекрёстное ребро.

Для каждой вершины v в процессе поиска в глубину запомним ещё два параметра: в $d[v]$ запишем «время» i-го попадания в вершину, а в $f[v]$ — «время» $(i+1)$-го попадания. Здесь под «временем» понимается номер шага алгоритма. Если вершина u серая, то это означает, что последнее ребро — обратное и получен цикл, содержащий u. Если $d[u] \neq 0$, то данный цикл является простым, так как вершина u встретилась второй раз. Длина цикла равна разности $f[u] - d[u]$.

Вычисление длин простых циклов реализуется в ходе алгоритма поиска в глубину и имеет порядок временной сложности $\mathrm{O}(n)$.

Из полученного набора длин циклов необходимо выделить множество всех различных длин с помощью упорядочивания чисел [8]. Сложность увеличится не более чем в $\mathrm{O}(\log n)$ раз, то есть вычислительная сложность алгоритма определения длин всех простых циклов графа оценивается величиной $\mathrm{O}(n^2 \log n)$.

Ёмкостная сложность алгоритма определяется размером памяти, необходимым для хранения матрицы смежности вершин графа, то есть составляет $\mathrm{O}(n^2)$ битов.

6. Определение экспонента графа с помощью возведения в степень матрицы смежности вершин

Определение экспонента графа связано с возведением в степень матрицы M смежности вершин графа и с проверкой положительности её элементов.

Известна [9, 10] достижимая оценка экспонента матрицы $\exp M \leqslant n^2 - 2n + 2$, где n — порядок матрицы. То есть если матрица M^t имеет при $t > n^2 - 2n + 2$ хотя бы один нулевой элемент, то соответствующий граф не примитивен. Если $M^t > 0$, то матрица и граф примитивны и $\exp M = \exp \Gamma \leqslant t$.

Для оценки вычислительной сложности алгоритма рассмотрим однопроцессорную вычислительную модель. Элементарными операциями считаем сложение и умножение в кольце целых чисел. Рассмотрим операцию умножения в полугруппе G_n. Размер необходимой памяти оценим в битах.

Сложность умножения квадратных матриц размера n имеет порядок $\mathrm{O}(n^3)$. При этом для распознавания примитивности достаточно возвести матрицу в степень не выше 2^r, где $r = \lceil \log_2(n^2 - 2n + 2) \rceil$. С помощью алгоритма быстрого возведения в степень [8] потребуется $\mathrm{O}(rn^3) = \mathrm{O}(n^3 \log_2 n)$ операций для определения примитивности матрицы.

Опишем подробнее алгоритм быстрого возведения в степень для точного вычисления экспонента матрицы. Возведем матрицу M в квадрат. Полученную после первого возведения матрицу M^2 ещё раз возведем в квадрат, получим M^4 и т. д. Пусть матрица

M^k положительна. Тогда вернемся к матрице $M^{k/2}$ и умножим её на матрицу $M^{k/4}$, и далее будем делить пополам отрезок, содержащий значение $\exp M$, пока не определится наименьшая степень t, при которой матрица M^t положительна.

Оценим ёмкостную сложность алгоритма. В памяти достаточно хранить матрицы M^t, где $t = 2^0, 2^1, \ldots, 2^{r-1}$. Таким образом, ёмкостная сложность алгоритма составляет $\mathrm{O}(n^2 \log_2 n)$.

Сравним эти значения с оценками сложности алгоритма распознавания примитивности n-вершинного ориентированного графа с помощью поиска в глубину, полученными в п. 5. Результаты сравнения приведены в табл. 3.

Таблица 3

Сложность алгоритмов распознавания примитивности графа

Алгоритм	Временная сложность	Ёмкостная сложность
Поиск в глубину	$\mathrm{O}(n^2 \log n)$ обращений к памяти	$\mathrm{O}(n^2)$ бит
Возведение в степень матрицы смежности	$\mathrm{O}(n^3 \log n)$ сложений и умножений единиц и нулей	$\mathrm{O}(n^2 \log_2 n)$ бит

Отметим, что, в отличие от первого алгоритма, второй определяет значение показателя примитивности.

ЛИТЕРАТУРА

1. *Фомичев В. М.* Оценки экспонентов примитивных графов // Прикладная дискретная математика. 2011. № 2(11). С. 101–112.
2. *Сачков В. Н., Тараканов В. Е.* Комбинаторика неотрицательных матриц. М.: ТВП, 2000.
3. *Биркгоф Г.* Теория решеток. М.: Наука, 1984.
4. *Арнольд В. И.* Экспериментальное наблюдение математических фактов. М.: МЦНМО, 2006.
5. *Коблиц Н.* Курс теории чисел и криптографии. М.: ТВП, 2001.
6. *Rosser B.* The n-th prime is greater than $n \log n$ // Proc. London Math. Soc. 1939. V. 45. P. 21–44.
7. *Лахно А. П.* Поиск в глубину и его применение // Московские олимпиады по информатике. М.: МЦНМО, 2006.
8. *Порублев И. Н., Ставровский А. Б.* Алгоритмы и программы. Решение олимпиадных задач. М.: Вильямс, 2007
9. *Wielandt H.* Unzerlegbare nicht negative Matrizen // Math. Zeitschr. 1950. No. 52. S. 642–648.
10. *Сачков В. Н., Ошкин И. Б.* Экспоненты классов неотрицательных матриц // Дискретная математика. 1993. № 2. С. 150–159.

МАТЕМАТИЧЕСКИЕ МЕТОДЫ КРИПТОГРАФИИ

DOI 10.17223/20710410/16/2

УДК 512.62

ДИОФАНТОВА КРИПТОГРАФИЯ НА БЕСКОНЕЧНЫХ ГРУППАХ

В. А. Романьков

Омский государственный университет им. Ф. М. Достоевского, г. Омск, Россия

E-mail: romankov48@mail.ru

В работе даётся краткое представление криптографии, основанной на группах («group-based cryptography»), — современного направления, главными объектами в котором являются абстрактные бесконечные группы, а основной целью — построение на них криптографических примитивов, систем и протоколов. Исследования по этому направлению ведутся методами теории групп, теории сложности и теории вычислений. Обращается внимание на использование неразрешимых и трудноразрешимых алгоритмических проблем теории групп в качестве основы обозначенного построения. Обсуждаются аспекты сложности алгоритмических проблем и связанных с ними проблем поиска. Объясняется универсальность диофантова языка в криптографии. Отмечается его объединяющая роль.

В качестве возможных платформ для криптографических систем и протоколов предлагается использовать свободные метабелевы группы. Приводятся основания для такого использования, в том числе алгоритмическая разрешимость в этих группах проблемы равенства и наличие нормальных форм записи элементов группы. Ещё одним основанием является алгоритмическая неразрешимость в таких группах проблемы существования решений у групповых уравнений и алгоритмическая неразрешимость проблемы эндоморфной сводимости, вытекающих из неразрешимости 10-й Проблемы Гильберта. Предполагается, что в последующей работе автора совместно с С. Ю. Ерофеевым материал данной работы будет использован для построения на свободных метабелевых группах возможно односторонних функций и протоколов аутентификации с нулевым разглашением.

Ключевые слова: *криптография, основанная на группах, алгоритмическая проблема, проблема поиска, генерическая сложность, диофантов язык, диофантова криптография, свободная метабелева группа, проблема эндоморфной сводимости.*

Введение

Понятие «криптография, основанная на группах» (англ. вариант — *group-based cryptography*) появилось сравнительно недавно — на рубеже XX и XXI столетий. Так обозначено направление исследований, основными объектами которого являются абстрактные бесконечные группы, а основной целью — построение на них криптографических систем и протоколов. Исследования ведутся методами теории групп, теории сложности и теории вычислений.

Данному направлению посвящены монографии [1, 2], а также множество статей, затрагивающих самые разные вопросы. Часть из этих работ представляет криптографические примитивы, другая описывает криптографические системы и протоколы,

третья исследует вопросы сложности и т. д. Некоторые работы посвящена вскрытию слабостей представленных систем и протоколов. Имеются и практические разработки, в том числе компьютерные программы, для использования полученных результатов на практике. Обширный список работ по данному направлению можно найти в [3].

В настоящей работе даётся краткое представление о криптографии, основанной на группах, обсуждается использование неразрешимых и трудноразрешимых алгоритмических проблем теории групп в качестве основы для построения криптографических систем и протоколов. Для обоснования возможности такого использования мы обращаем внимание на следующие аспекты:

— Как должна быть задана и какими свойствами должна обладать группа, претендующая на роль платформы для построения криптографических систем и протоколов?
— Какой должна быть алгоритмическая проблема, претендующая на роль основы для построения криптографических систем и протоколов?
— Каковы общие принципы обозначенных построений и чем должна обуславливаться их криптостойкость?

Подобные вопросы уже неоднократно рассматривались в литературе. После того как вышла пионерская в данном направлении работа М. Аншеля и др. [4], в которой представлена схема генерации общего ключа, известная теперь как *протокол Аншель — Аншеля — Голдфельда*, появилось множество работ, эксплуатирующих те или иные алгоритмические проблемы для построения систем и протоколов. В [4] в качестве платформы берётся группа кос Артина B_n на n нитях для достаточно большого n. В качестве алгоритмической в ней рассматривается проблема сопряжённости в группе B_n двух наборов элементов $\bar{g} = (g_1, \ldots, g_k)$ и $\bar{f} = (f_1, \ldots, f_k)$. Алгоритм может быть представлен в любой группе. Конкретный её выбор обуславливается многими факторами. Во-первых, группа, выбранная в качестве платформы, должна быть удобной для реализации алгоритма. В то же время необходимо обеспечить криптостойкость алгоритма. Об этом более подробно говорится в дальнейшем. Опишем сам алгоритм в общем виде.

Пусть G — конечно порождённая группа с множеством порождающих элементов $\{x_1, \ldots, x_n\}$, которое можно считать фиксированным. Любой элемент группы G записывается в виде группового слова от порождающих элементов. *Групповым* называется слово, записанное от букв $x_1, \ldots, x_n$ и формальных обратных $x_1^{-1}, \ldots, x_n^{-1}$. Конечно, такая запись элемента неоднозначна. Во-первых, возможны сокращения подслов вида $x_i^{\varepsilon} x_i^{-\varepsilon}$ для $\varepsilon = \pm 1$. Если таких подслов нет, то слово называется *редуцированным* или *несократимым*. Если $G = F_n$ — свободная группа с базисом $\{x_1, \ldots, x_n\}$, то запись элемента в виде несократимого слова однозначна. В группе, не являющейся свободной, существуют нетривиальные слова, записывающие наравне с пустым словом тривиальный элемент. Они называются *соотношениями* группы.

Предположим, что в группе G существует нормальная форма записи её элементов. Это означает, что любой элемент может быть однозначно записан в виде канонического слова от порождающих элементов. Переход от произвольной записи элемента $g = g(x_1, \ldots, x_n)$ к его записи в нормальной форме нф(g) предполагается эффективным. Обычно он осуществляется через переписывающий процесс, получающий на входе произвольное слово от порождающих элементов и выдающий на выходе запись соответствующего элемента в нормальной форме. В свободной группе с базисом $\{x_1, \ldots, x_n\}$ это обычный процесс сокращения подслов вида $x_i^{\varepsilon} x_i^{-\varepsilon}$, о которых гово-

рилось выше. Известно, что результат — несократимое слово — не зависит от выбора порядка сокращения.

В протоколе Аншель — Аншеля — Голдфельда корреспонденты **A** и **B**, часто именуемые как Алиса и Боб, выбирают наборы элементов $\bar{g} = (g_1, \ldots, g_k)$ и $\bar{f} = (f_1, \ldots, f_k)$ группы G, причём **A** выбирает набор $\bar{g}$, а **B** — набор $\bar{f}$. Эти наборы считаются известными (*public*). Затем **A** выбирает секретное (*private*) слово $u = u(f_1, \ldots, f_k)$, а **B** — секретное слово $v = v(g_1, \ldots, g_k)$. Они могут это сделать, так как наборы известны. Далее **A** выполняет сопряжение набора $\bar{f}$ элементом u, получая набор $\bar{f}^u = (f_1^u, \ldots, f_k^u)$. Запись вида a^b означает сопряжение bab^{-1} элемента a элементом b. В дальнейшем через $[a, b]$ обозначается *коммутатор* $= aba^{-1}b^{-1}$ элементов a и b. Корреспондент **A** вычисляет и публикует нормальную форму $\text{нф}(\bar{f}^u) = (\text{нф}(f_1^u), \ldots, \text{нф}(f_k^u))$. Подобным образом корреспондент **B** вычисляет и публикует набор $\text{нф}(\bar{g}^v) = (\text{нф}(g_1^v), \ldots, \text{нф}(g_k^v))$.

Предполагается, что в группе G вычисление по опубликованным нормальным формам сопряжённых наборов элементов u и v — трудная задача. На этом основывается криптостойкость протокола. После выхода [4] появилось множество работ с анализом этой криптостойкости, где подчёркивается важность выбора параметров, ключей и т. п. Появились соответствующие рекомендации и т. п.

На выходе протокола корреспонденты получают секретный ключ. Делается это следующим образом. Корреспондент **A** вычисляет элемент

$$u\,(\text{нф}(g_1^v), \ldots, \text{нф}(g_k^v))\;u^{-1} = [v, u].$$

Корреспондент **B** вычисляет тот же самый элемент $[v, u]$ из равенства

$$v \cdot v(\text{нф}(f_1^u), \ldots, \text{нф}(f_k^u))^{-1} = [v, u].$$

Таким образом пользователи **A** и **B** генерируют общий известный только им ключ $K = \text{нф}[v, u]$.

Мы здесь не приводим и не обсуждаем конкретные свойства групп кос Артина B_n, выбранных авторами [4] в качестве платформы своего протокола. Относительно этих свойств см. классическую работу А. А. Маркова [5], монографию П. Дехорноя [6], обзор В. Я. Лина [7] или ещё какую-нибудь монографию, посвящённую группам кос Артина.

О криптографии на группах кос Артина, включая подробное описание протокола Аншель — Аншеля — Голдфельда, см. обзоры П. Дехорноя [8] и К. Мальбурга [9].

Нам важно выделить некоторые основные, с нашей точки зрения, свойства группы кос Артина, позволяющие рассматривать приведённые построения как заслуживающие внимания. Эти свойства следующие:

— Группы B_n при любом n являются конечно определёнными, то есть порождаются конечными множествами порождающих элементов, все соотношения между которыми следуют из конечного множества определяющих соотношений.
— Группы B_n обладают нормальными формами однозначной записи элементов, переход к которым от записей в виде групповых слов эффективен.
— В группах кос нахождение сопрягающего элемента u по элементу g и нормальной форме $\text{нф}(g^u)$ сопряжённого к нему элемента является трудноразрешимой задачей. Более того, она остаётся трудноразрешимой при замене одного элемента на набор элементов.

Отмеченные свойства так или иначе присутствуют в большинстве работ. Хочется ещё заметить, что в работе [4] фактически впервые существенно использовалась

некоммутативность группы. Более того, представленный протокол не являлся переносом известных протоколов теоретико-числового характера. Подобные переносы, например, в матричные группы уже были известны, но не дали толчка для последующего развития.

В последующей работе автора совместно с С. Ю. Ерофеевым «О построении возможно односторонних функций на основе алгоритмической неразрешимости проблемы эндоморфной сводимости в свободных метабелевых группах» мы перейдём к конкретным предложениям. В качестве платформы для построения криптографических примитивов, систем и протоколов выбираем свободные метабелевы группы. Свободная метабелева группа M_n ранга n, где n — натуральное число, определяется как факторгруппа F_n/F_n'' свободной группы F_n ранга n по второму коммутанту F_n''. Напомним, что коммутантом G' произвольной группы G называется её подгруппа, порождённая всеми коммутаторами $[g, f]$ элементов группы G. Подгруппа G' нормальна в группе G, фактор-группа G/G' по ней абелева. Второй коммутант G'' определяется как коммутант от коммутанта $(G')'$. Он также нормален в группе G. Группа G называется *метабелевой*, если G'' тривиален. В этом случае коммутант G' абелев, а группа G является расширением абелевой нормальной подгруппы G' с помощью абелевой факторгруппы G/G'. Отсюда её название.

Группа M_n называется *свободной метабелевой группой ранга* n, потому что в ней есть *базис* $X_n = \{x_1, \ldots, x_n\}$, состоящий из n элементов, такой, что любое отображение этого базиса $X_n \to G$ в произвольную метабелеву группу G однозначно продолжается до гомоморфизма $M_n \to G$. Говорят также, что группа M_n — *свободная группа ранга* n *многообразия всех метабелевых групп* $\mathcal{A}^2$. Базис X_n называют *множеством свободных порождающих* группы M_n. О многообразиях групп см. монографию Х. Нейман [10].

В качестве алгоритмической проблемы возьмём проблему $E(M_n)$ эндоморфной сводимости в группе M_n. Известно [11, 12], что она алгоритмически неразрешима при достаточно большом $n \geqslant n_0$. Основываясь на её алгоритмической неразрешимости, укажем метод построения функции $f_n : M_n \to M_n$, претендующей на роль односторонней функции. Наконец, используя платформу M_n и функцию f_n, предложим протокол аутентификации с нулевым разглашением пользователя в системе. Подобный протокол уже предлагался в [13], причём также со ссылкой на работу автора [12]. Однако так просто воспользоваться группами и функциями из [12] в данном случае нельзя. Мы покажем, что для криптостойкости протокола аутентификации необходима алгоритмическая неразрешимость более сильной проблемы *двукратной эндоморфной сводимости*.

Важно отметить, что алгоритмическая неразрешимость проблемы эндоморфной сводимости $E(M_n)$ при $n \geqslant n_0$, установленная в [12], базируется на алгоритмической неразрешимости 10-й Проблемы Гильберта о существовании алгоритма, определяющего по произвольному уравнению вида $d(\zeta_1, \ldots, \zeta_k) = 0$, где $d(\zeta_1, \ldots, \zeta_k)$ — многочлен с целыми коэффициентами, имеет ли это уравнение решение в целых числах. Алгоритмическая неразрешимость 10-й Проблемы Гильберта установлена Ю. В. Матиясевичем в [14] (полное доказательство в [15], см. также [16]).

В данной работе мы показываем, что диофантов язык является достаточно универсальным. На нем записываются функции и уравнения, фигурирующие во многих криптографических системах и протоколах, в том числе в системе RSA и протоколах, основанных на понятии дискретного логарифма в мультипликативных группах конеч-

ных полей. Диофантов язык позволяет объединять эти системы и протоколы в единое целое, что даёт возможность ввести в рассмотрение *диофантову криптографию*.

В заключение отметим, что построение криптографических систем и протоколов, основанных на неразрешимых и трудноразрешимых проблемах, осуществлялось многими авторами (см., например, монографии [1, 2], обзор [8], статьи [17 – 21]).

1. Бесконечные группы и алгоритмические проблемы

Рассмотрим постановку алгоритмических проблем только для групп, хотя аналоги этих проблем легко формулируются и для других алгебраических и не только алгебраических систем. Заметим, что истоки этих проблем лежат в топологии.

1.1. Постановка алгоритмических проблем

В самых общих чертах алгоритмическая проблема выглядит следующим образом. Имеется теоретико-групповое свойство $\mathbf{P}$, которое может относиться как к отдельным элементам, так и к наборам элементов, к подгруппам или подмножествам группы, к различным группам и т. п. Требуется определить, обладают ли свойством $\mathbf{P}$ указанные объекты. Более точно проблема формулируется как следующий вопрос: *существует ли алгоритм, определяющий за конечное число шагов, удовлетворяет объект $\mathcal{O}$ свойству $\mathbf{P}$ или нет*?

При постановке проблемы упоминание алгоритма часто опускается. Говорят, что проблема *алгоритмически разрешима* (или просто *разрешима*), если такой алгоритм существует, и *неразрешима* в противном случае.

При постановке алгоритмической проблемы предполагается, что группа, её элементы, подгруппы, подмножества, словом, объекты, для которых ставится проблема, заданы каким-либо эффективным образом. Способов эффективного задания существует довольно много. Далее опишем некоторые из них.

Классические алгоритмические проблемы теории групп сформулированы в начале XX столетия Максом Деном. Они ставились для класса конечно определённых групп. Это означает, что группа G, для которой ставится проблема, задана своим *конечным представлением* вида

$$\mathcal{P}(G) = \langle x_1, \ldots, x_n \mid r_1, \ldots, r_m \rangle. \tag{1}$$

Иначе говоря, группа G является фактор-группой F_n/R свободной группы F_n с базисом (множеством свободных порождающих элементов) $\{x_1, \ldots, x_n\}$ относительно *нормального замыкания* $R = \text{нз}(r_1, \ldots, r_m)$, то есть минимальной нормальной подгруппы группы F_n, содержащей элементы $r_1, \ldots, r_m$. Элементы $r_1, \ldots, r_m$ записываются в виде групповых слов от порождающих $x_1, \ldots, x_n$. Напомним, что групповое слово записывается как слово от элементов $x_1^{\pm 1}, \ldots, x_n^{\pm 1}$. Элементы $r_1, \ldots, r_m$ называются *определяющими словами*. Иногда представление (1) записывают в виде

$$\mathcal{P}(G) = \langle x_1, \ldots, x_n \mid r_1 = 1, \ldots, r_m = 1 \rangle. \tag{2}$$

Смысл задания остается тем же самым, равенства $r_i = 1$ для $i = 1, \ldots, m$ называют *определяющими соотношениями* группы G.

Через H' обозначим *коммутант* группы H, то есть подгруппу, порождённую в группе H всеми коммутаторами её элементов. Коммутант H' является нормальной подгруппой группы H, фактор-группа H/H' по которой абелева. Более того, коммутант — наименьшая подгруппа с этим свойством, то есть если фактор-группа H/N по какой-либо нормальной подгруппе N группы H абелева, то N обязательно содержит H'.

Элементы нормального замыкания $R = \text{нз}(r_1, \ldots, r_m)$ допускают описание как групповые слова следующего вида:

$$u = \prod_{j=1}^{k} (r_{i_j}^{g_j})^{\varepsilon_j}, \tag{3}$$

где $i_j \in \{1, \ldots, m\}$; $g_j \in F_n$; $\varepsilon_j = \pm 1$ для $j = 1, \ldots, k$.

Естественно определяется *канонический гомоморфизм* $F_n \to G$, переводящий произвольный элемент g группы F_n в элемент gR группы G. Группа G имеет каноническое множество порождающих элементов $y_i = x_i R$ для $i = 1, \ldots, n$. Любой элемент g группы G можно поэтому записать в виде группового слова $g = g(y_1, \ldots, y_n)$. Однако часто порождающие y_i группы G обозначают теми же буквами x_i, что и их прообразы. Элемент g записывается в виде $g(x_1, \ldots, x_n)$.

Классические алгоритмические проблемы Дена формулируются следующим образом.

1. **Проблема равенства.** Определить по двум групповым словам $g = g(x_1, \ldots, x_n)$ и $f = f(x_1, \ldots, x_n)$, записывают ли они один и тот же элемент группы G, заданной своим конечным представлением (1) или (2). Другими словами, верно ли, что в группе G справедливо равенство $g = f$. Иногда, чтобы указать группу, пишут $g =_G f$.

Рассмотрение проблемы равенства может быть сведено к случаю, когда один из элементов равен 1. Действительно, равенство $g =_G f$ выполнено тогда и только тогда, когда $gf^{-1} =_G 1$. Оно может быть также перенесено в группу F_n, поскольку равенство $g =_G 1$ равносильно тому, что слово $g = g(x_1, \ldots, x_n)$ как элемент группы F_n принадлежит нормальной подгруппе R.

Первые примеры конечно определённых групп с неразрешимой проблемой равенства были построены в 50-е годы XX столетия П. С. Новиковым в [22, 23]. Впоследствии таких примеров стало достаточно много.

2. **Проблема сопряжённости.** Определить, задают ли два слова $g = g(x_1, \ldots, x_n)$ и $f = f(x_1, \ldots, x_n)$ сопряжённые элементы группы G. Другими словами, существует ли элемент h группы G, такой, что $g^h =_G f$.

Разрешимость проблемы сопряжённости в группе G, очевидно, влечет разрешимость в G проблемы равенства. Действительно, элемент g группы G равен 1 тогда и только тогда, когда g сопряжён с 1. Обратное утверждение в общем случае неверно. Существуют конечно определённые группы, в которых проблема равенства разрешима, а проблема сопряжённости неразрешима. Первые примеры групп с неразрешимой проблемой сопряжённости, некоторые из которых имеют разрешимую проблему равенства, построены в [23].

3. **Проблема изоморфизма.** Определить по двум представлениям конечно определённых групп G и H, изоморфны эти группы или нет.

Неразрешимость проблемы изоморфизма в классе конечно определённых групп установлена С. И. Адяном [24].

Отметим ещё одну проблему, которая хотя и не была явно сформулирована Деном, но впоследствии стала одной из основных.

4. **Проблема вхождения.** Определить для произвольного элемента g данной конечно определённой группы G и произвольной её конечно порождённой подгруппы H, принадлежит g подгруппе H или нет.

Проблему вхождения часто называют *обобщённой проблемой равенства*. Очевидно, что её разрешимость влечет разрешимость проблемы равенства, равносильной про-

блеме вхождения в тривиальную подгруппу. Выделяют также проблему вхождения в фиксированную конечно порождённую подгруппу.

Известные результаты и открытые проблемы алгоритмического характера в теории групп освещены в обзорах [25 – 27].

В некоторых важных классах групп классические алгоритмические проблемы разрешимы. Например, все они разрешимы в классах свободных и конечно порождённых абелевых групп. Многие проблемы разрешимы в классах нильпотентных и полициклических групп. Имеются важные результаты о разрешимости алгоритмических проблем в матричных группах. Большое внимание уделено разработке практических алгоритмов решения алгоритмических проблем в группах (см. по этому поводу монографии [28, 29] и обзорную статью [30]).

Таким образом, первоначально при постановке алгоритмических проблем рассматривались только конечно определённые группы, элементы которых записываются в виде групповых слов. Впоследствии класс групп, для которых ставятся алгоритмические проблемы, был расширен как за счёт включения в него *рекурсивно определённых* групп, в которых множество порождающих элементов конечно, а множество определяющих соотношений может быть бесконечным рекурсивно перечислимым множеством, так и за счёт конечно порождённых подгрупп каких-нибудь известных хорошо заданных групп. Например, группа может быть задана своим конечным порождающим множеством в матричной группе над конструктивным полем или более общо — над кольцом. Группа может также быть задана как конструктивный объект в смысле теории моделей. Она может определяться и другими эффективными способами.

Класс конечно порождённых матричных групп над конструктивными кольцами представляет особый интерес. Проблема равенства в таких группах, очевидно, разрешима. Однако другие проблемы даже в достаточно просто устроенных матричных группах могут быть неразрешимыми.

Одним из самых известных является пример К. А. Михайловой матричной группы с неразрешимой проблемой вхождения, описанный в работе [31]. Эта группа есть прямое произведение $G = F_2 \times F_2$ двух копий свободной группы ранга 2. Она допускает точное представление матрицами порядка 4 над кольцом целых чисел $\mathbb{Z}$. Опишем одно из возможных таких представлений. Хорошо известно (см., например, [32, 33]), что представление (оно называется *представлением Санова*) группы F_2 матрицами порядка 2 над $\mathbb{Z}$, заданное на порождающих элементах x_1, x_2 группы F_2 следующим отображением, точное:

$$x_1 \mapsto \begin{pmatrix} 1 & 2 \\ 0 & 1 \end{pmatrix}, \quad x_2 \mapsto \begin{pmatrix} 1 & 0 \\ 2 & 1 \end{pmatrix}.$$

Точное представление группы G матрицами 4-го порядка соответствует схеме

$$\begin{pmatrix} F_2 & 0 \\ 0 & F_2 \end{pmatrix},$$

согласно которой первая копия группы F_2 представляется верхней левой клеткой, а вторая — нижней правой (остальные матричные элементы в представлении множителей группы G, как у единичной матрицы).

Неразрешимость проблемы вхождения в группе G основывается на существовании 2-порождённой конечно определённой группы с неразрешимой проблемой равенства. Пусть такая группа K задана представлением $\mathcal{P}(K) = \langle x_1, x_2 \mid r_1, \ldots, r_m \rangle$. Рассмотрим

в группе G подгруппу H, порождённую элементами вида $f_1 = (x_1, x_1)$, $f_2 = (x_2, x_2)$, $h_i = (r_i, 1)$ для $i = 1, \ldots, m$. Легко показать, что элемент $g = (g', 1)$ принадлежит H тогда и только тогда, когда g' принадлежит нормальной подгруппе $R = \text{нз}(r_1, \ldots, r_m)$ группы F_2. Неразрешимость проблемы равенства в группе $K = F_2/R$ влечет неразрешимость проблемы вхождения в подгруппу H группы G.

2. Неразрешимые и трудноразрешимые алгоритмические проблемы как основа для построения криптографических систем и протоколов

Классические алгоритмические проблемы Дена уже неоднократно предлагались в качестве основы для построения на группах криптографических примитивов, систем и протоколов. Традиционно наибольшее внимание привлекает в этой связи проблема сопряжённости. В уже упоминавшейся работе М. Аншеля и др. [4] основой криптостойкости протокола служит трудноразрешимость проблемы сопряжённости в группах кос Артина. Заметим, что проблема сопряжённости в них разрешима. Это установлено в работе Гарсайда [34]. Тем не менее, так как эффективный алгоритм для её решения до сих пор не найден, эта задача продолжает считаться трудноразрешимой (см. публикации [19, 35, 36]). Проблема равенства фигурирует в работах [37 – 39]; проблема вхождения — в работе [18]. Этот список — только малая часть работ, обсуждающих использование алгоритмических проблем в криптографии.

2.1. Проблемы поиска

Абсолютное большинство работ в криптографии, основанной на группах, в которых рассматриваются трудноразрешимые и неразрешимые проблемы, связано с решением так называемых *проблем поиска* (англ. вариант — *search problems*). Например, если в основе криптографического примитива лежит проблема сопряжённости, то обычно, как это происходит, например, в протоколе Аншель — Аншеля— Голдфельда, известно, что данные элементы или наборы элементов, если речь идёт о наборах, сопряжены. Задача заключается в эффективном нахождении сопрягающего элемента. В общих чертах, если алгоритмическая проблема ставится для объекта $\mathcal{O}$ относительно свойства $\mathbf{P}$, то проблема поиска выясняет в случае, если $\mathcal{O}$ обладает свойством $\mathbf{P}$, доказательство, или ещё говорят — *свидетельство* этого, справедливость которого легко проверить. Например, если известно, что элемент f группы G принадлежит подгруппе H, порождённой элементами $h_1, \ldots, h_k$, то требуется найти запись f в виде группового слова от этих элементов. Для проблемы равенства, аналогично, требуется найти выражение слова $u = u(x_1, \ldots, x_n)$ от порождающих элементов $x_1, \ldots, x_n$ свободной группы F_n, записывающего тривиальный элемент группы $G = F_n/R$, где $R = \text{нз}(r_1, \ldots, r_m)$ — его выражение в виде (3). Это, конечно, можно сделать простым перебором, но сейчас речь идёт о реальных вычислениях, когда такой перебор уже не может рассматриваться как эффективный.

Если алгоритмическая проблема, взятая за основу криптографического примитива, не допускает полиномиального алгоритма, то не существует также полиномиального ограничения длины входа от длины наблюдаемого выхода, значит, нельзя организовать полиномиальный перебор возможных входов, основываясь на такой оценке. Если же основная алгоритмическая проблема неразрешима, то невозможно вообще дать какую-либо оценку длины входа, зная длину выхода. В этом случае неприменим метод «грубой силы», то есть полного перебора.

Это рассуждение носит, конечно, самый общий характер. В теории сложности подобные вопросы получают строгое обоснование. Этому в основном посвящена монография [2].

3. О сложности алгоритмических проблем и соответствующих им проблем поиска

Современная теория сложности зародилась в 70-е годы XX столетия. Для нас важное значение имеет прежде всего понятие временной сложности. Действительно, криптография, основанная на группах, как, впрочем, и всякая другая область, ориентированная на практическое использование, должна заботиться о реальном времени вычислений в разрабатываемых ею протоколах. Значит, нам небезразлично, сколь долго будет работать алгоритм. В то же время при практическом использовании разрабатываемых протоколов различные входящие в них параметры, в том числе ключи, выбираются случайным образом. Значит, необходимо заботиться не только о сложности в худшем случае, когда проблему нельзя эффективно решить при каких-то специфических данных, но и о сложности, проявляющейся при случайном выборе данных. Здесь разрабатываются два основных подхода, связанные с определением понятий *сложности в среднем* и *генерической сложности*. Перейдём к общему схематическому описанию, отсылая за деталями к монографиям [1, 2, 40], сборнику [41] и статьям [42 – 45].

Итак, рассматриваем три основных вида сложности:

— сложность по худшему случаю;
— сложность в среднем;
— генерическую сложность.

О классическом понятии сложности по худшему случаю см., например, монографию [40]. Класс сложности $\mathcal{C}$ определяется спецификацией модели вычислений (для нас это многоленточная машина Тьюринга), типом вычислений (то есть использованием либо детерминированной, либо вероятностной машины Тьюринга), а также ресурсами, объём которых необходимо контролировать (обычно это время работы алгоритма, пространство, занимаемое данными, или же то и другое). Данные спецификации позволяют определить функцию сложности $f(n)$, где n — размер входа, оценивающую объём необходимых ресурсов для вычисления соответствующего ему выхода. Мы не приводим точного определения, замечая только, что оно позволяет говорить о линейной, полиномиальной или экспоненциальной сложности алгоритма. Как правило, считается, что линейные, квадратичные и в иных случаях полиномиальные для малых степеней алгоритмы достаточно быстры, а экспоненциальные медленны. Конечно, это все относительно и требует конкретизации в каждом отдельном случае.

Кратко остановимся на понятии *сложности в среднем*. Для её определения необходимо, чтобы на пространстве всех возможных входов была задана функция распределения вероятностей или хотя бы какая-нибудь аддитивная неотрицательная функция. При её задании сложность в среднем чаще всего оценивается математическим ожиданием объёма ресурсов, необходимым для работы алгоритма на случайно выбранном входе. Опять же можно говорить о линейной, полиномиальной и экспоненциальной сложности в среднем. На конечных множествах, как правило, задают равномерную функцию распределения. В бесконечном случае вопрос о задании такой функции становится более тонким. Например, задача определения «случайного» элемента группы рассматривалась многократно. Даже для конечных групп этот вопрос решается далеко не очевидным путём. Действительно, что считать случайным выбором? Уже давно стало понятным, что при учёте алгебраической структуры группы её элементы как бы становятся «неравноправными» (см. по этому поводу [46]).

В работе [47] предложен следующий возможный общий подход к построению обсуждаемого распределения на бесконечной конечно порождённой группе. Пусть груп-

па G наделена функцией натуральнозначной длины $l : G \to \mathbb{N}$, такой, что множество $\mathbb{S}_r$ всех элементов g группы G длины $l(g) = r$ для любого натурального числа r конечно. Считаем также, что $\mathbb{S}_0 = \{1\}$. Функция l в конкретных случаях может называться функцией размера, сложности и т. п. Множество $\mathbb{S}_r$ естественно называть *сферой радиуса* r. Аналогичным образом определяется *шар* $\mathbb{B}_r$ радиуса r, состоящий из всех элементов g группы G, для которых $l(g) \leqslant r$. Затем берётся одна из функций распределения $f : \mathbb{N} \cup \{0\} \to \mathbb{R}$, определённая на множестве натуральных чисел с нулем. Например, это может быть функция Пуассона, биномиального или экспоненциального распределения. Предполагается только её невырожденность, т. е. что для любого $r \in \mathbb{N}$ выполняется $f(r) \neq 0$. Для задания функции распределения вероятностей $p : G \to [0, 1]$ для любого r полагаем $p(\mathbb{S}_r) = f(r)$. Далее для любого $g \in \mathbb{S}_r$ полагаем $p(g) = 1/f(r)$, т. е. все элементы сферы $\mathbb{S}_r$ считаются равновероятными. Это определяет функцию распределения вероятностей на всей группе G, что, в свою очередь, даёт возможность говорить о сложности в среднем алгоритма.

Обычно в качестве значения $l(g)$ для элемента g конечно порождённой группы G с фиксированным конечным множеством X порождающих элементов берётся длина кратчайшей записи элемента g в виде группового слова от этих порождающих. *Расстоянием* $d(g, f)$ между элементами g и f группы G считается значение $l(gf^{-1})$. Группа G таким образом превращается в метрическое пространство, изоморфное графу Кэли, соответствующему выбранной системе порождающих элементов. Данная метрика называется *словарной*; длина элемента относительно неё есть его расстояние от 1.

Понятие сложности вычислений в среднем относительно возможных практических приложений в криптографии, основанной на группах, обладает рядом недостатков. Во-первых, возникает вопрос адекватного выбора функции распределения $f : \mathbb{N} \cup \{0\} \to [0, 1]$. Во-вторых, алгоритм может оказаться таким, что он работает чрезвычайно долго только на малой доле возможных входов, а на остальных входах он достаточно быстр. Усреднённое время его работы будет в этом случае не показательным, так как при случайном выборе «плохие» входы будут встречаться крайне редко. Хочется привести в этой связи аналогию с симплекс-методом. В работе [48] показано, что «плохие» входы для него очень специфичны, поэтому их пренебрежимо мало. Это объясняет тот факт, что на практике симплекс-метод вполне хорошо себя зарекомендовал; он широко используется, в то время как знаменитый полиномиальный алгоритм Хачияна [49] имеет в основном теоретическое значение. Более точно, А. М. Вершик и П. В. Спорышев в [50] и независимо С. Смейл в [51] показали, что симплекс-алгоритм работает с линейной сложностью на множестве полной меры.

По описанным выше причинам представляется особенно важным понятие генерической сложности. Для его введения необходимо, чтобы на множестве всех входов была определена мера со значениями в $[0, 1]$. Это не обязательно должна быть функция распределения вероятностей. *Генерическим* называется множество полной меры, дополнение к которому имеет меру 0. Работа [45] представляет точное определение генерической сложности. В ней же установлено, что для широкого класса конечно порождённых групп классические алгоритмические проблемы равенства, сопряжённости и вхождения имеют линейную сложность при их ограничении на некоторое генерическое подмножество (см. также работу [52]).

Перейдём к определениям. Рассмотрим множество всех слов (включая пустое слово) Σ^* в конечном алфавите Σ, состоящем не менее чем из двух букв. Это множество является свободным моноидом со множеством свободных порождающих Σ. Так как на элементах Σ^* естественно определено понятие длины, можно ввести для любого неот-

рицательного целого числа r понятия сферы $\mathbb{S}_r$ и шара $\mathbb{B}_r$ радиуса r, как это объяснено выше. Очевидно, что эти множества конечны и непусты.

Пусть V — произвольное подмножество моноида Σ^*. *Относительной плотностью множества V в сфере* $\mathbb{S}_r$ называется отношение

$$\rho_{\mathbb{S}_r}(V) = \frac{|V \cap \mathbb{S}_r|}{|\mathbb{S}_r|}, \tag{4}$$

где $|\cdot|$ означает число элементов.

Аналогично вводится понятие *относительной плотности множества V в шаре* $\mathbb{B}_r$:

$$\rho_{\mathbb{B}_r}(V) = \frac{|V \cap \mathbb{B}_r|}{|\mathbb{B}_r|}. \tag{5}$$

Будем использовать для определения асимптотической плотности функцию (5), хотя совершенно аналогично можно дать определение, основываясь на функции (4).

Определение 1. *Асимптотической плотностью* подмножества V моноида Σ^* называется верхний предел

$$\rho(V) = \overline{\lim_{r\to\infty}} \rho_{\mathbb{B}_r}(V). \tag{6}$$

Если в (6) существует предел, то он обозначается через $\tilde{\rho}(V)$ и называется *строгой асимптотической плотностью* множества V. В этом случае нас интересует скорость сходимости к пределу $\tilde{\rho}(V)$ последовательности $\{\rho_{B_r}(V)\}_{r\in\mathbb{N}}$. Будем говорить, что сходимость последовательности *экспоненциально быстрая*, если существуют числа $0 \leqslant \sigma < 1$ и $C \geqslant 0$, такие, что для любого r имеет место оценка

$$|\tilde{\rho}(V) - \rho_{B_r}(V)| \leqslant C\sigma^r.$$

Определение 2. Подмножество V множества Σ^* называется *генерическим*, если $\tilde{\rho}(V) = 1$, и *строго генерическим*, если сходимость $\rho_{B_r}(V) \underset{r\to\infty}{\rightrightarrows} \tilde{\rho}(V)$ экспоненциально быстрая.

Если V — генерическое множество, то дополнение к нему $V' = \Sigma^* \setminus V$ называется *пренебрежимым*. В этом случае $\tilde{\rho}(V') = 0$.

Определения 1 и 2 легко распространяются на случай, когда вместо множества Σ^* рассматривается множество $(\Sigma^*)^k$ наборов из k элементов этого множества для $k \geqslant 2$. *Длиной* набора $\bar{g} = (g_1, \ldots, g_k)$ считаем сумму длин его компонент: $l(\bar{g}) = \sum\limits_{i=1}^{k} l(g_i)$.

Можно использовать также другое определение, согласно которому

$$l(\bar{g}) = \max\{l(g_i) : i = 1, \ldots, k\}.$$

Часто алгоритмические проблемы в группах трактуют как подмножества вида $D \subseteq (\Sigma^*)^k$ для некоторого алфавита Σ и натурального числа k. Например, если в качестве Σ взять множество символов, обозначающих порождающие элементы группы G и формальные обратные к ним, то элементы моноида Σ^* могут рассматриваться как групповые слова от порождающих элементов группы G. Рассмотрим подмножество $D_1(G) \subseteq \Sigma^*$, определяющее в группе G тривиальный элемент. Проблема равенства в группе G — это вопрос о принадлежности произвольного слова $u \in \Sigma^*$ подмножеству $D_1(G)$. Проблема сопряжённости на этом языке записывается как $D\ (G) \subseteq (\Sigma^*)^2$ и состоит из таких пар (u, v) слов в алфавите Σ, которые определяют сопряженные

в группе G элементы. Проблема вхождения относительно подгруппы H, порождённой элементами $h_1, \ldots, h_{k-1}$, имеет вид $D_H(G) \subseteq (\Sigma^*)^k$, $D_H = \{(u, h_1, \ldots, h_{k-1})\}$, где u определяет элемент подгруппы H. При этом подгруппа H считается фиксированной. Можно считать также, что фиксировано множество $\{h_1, \ldots, h_{k-1}\}$ порождающих её элементов. В этом случае на вход работы алгоритма должно подаваться слово u. Если рассматривать проблему вхождения в группе G, то элементы $h_1, \ldots, h_{k-1}$ уже не должны считаться фиксированными. При этом, поскольку число k в общем случае не ограничено, проблему вхождения необходимо рассматривать как подмножество бесконечной степени $(\Sigma^*)^\infty$.

Аналогичным образом можно записать широкий круг алгоритмических проблем относительно конечно порождённых групп.

Определение 3. Алгоритм $\mathcal{A}$ решает алгоритмическую проблему $D \subseteq (\Sigma^*)^k$ с *генерической сложностью* $\mathcal{C}$, если существует генерическое подмножество $V \subseteq (\Sigma^*)^k$, такое, что на любом входе из V алгоритм $\mathcal{A}$ работает со сложностью $\mathcal{C}$. Если множество V можно выбрать строго генерическим, то говорят, что алгоритм $\mathcal{A}$ решает проблему D со *строго генерической* сложностью $\mathcal{C}$.

Обращаем внимание на тот факт, что алгоритм $\mathcal{A}$ может быть частично определённым, то есть он может оказаться неопределённым на некоторых входах, которые можно включить в дополнение V' генерического множества V из определения 3. Может так получиться, что проблема D в целом на группе G алгоритмически неразрешима, а в то же время она генерически разрешима с приемлемой сложностью $\mathcal{C}$. Опыт показывает, что это случается довольно часто и для широкого круга алгоритмических проблем (см. по этому поводу [53, 54]). Опять же можно привести аналогию с симплекс-методом из линейного программирования, который на обсуждаемом языке оказывается генерически быстрым.

В практических приложениях выбор данных осуществляется, как правило, случайным образом. Если генерическая сложность какого-либо алгоритма незначительна, то алгоритм практически всегда применим и достаточно быстр, и может быть использован в практических приложениях.

4. Диофантова криптография

Диофантовым уравнением от n переменных называется выражение вида

$$d(\zeta_1, \ldots, \zeta_n) = 0, \tag{7}$$

где $d(\zeta_1, \ldots, \zeta_n)$ — многочлен с целыми коэффициентами от обозначенных независимых коммутирующих переменных. Множество всех таких многочленов составляет кольцо $\Lambda_n = \mathbb{Z}[\zeta_1, \ldots, \zeta_n]$. Кольцо Λ_m при $m \leqslant n$ естественно вложено в кольцо Λ_n. Объединение всех таких колец обозначим через $\Lambda = \mathbb{Z}[\zeta_1, \ldots, \zeta_n, \ldots]$.

На Втором математическом конгрессе, состоявшемся в 1900 г. в Париже, выдающийся математик Д. Гильберт изложил свои знаменитые 23 математические проблемы для математиков XX столетия. Впоследствии их стали называть *Проблемами Гильберта*. Среди этих проблем присутствовала 10-я Проблема о существовании эффективной процедуры, определяющей за конечное число шагов, имеет ли произвольное диофантово уравнение целочисленные корни. Говоря современным языком, 10-ю Проблему Гильберта можно перефразировать следующим образом: существует ли алгоритм, определяющий по произвольному диофантову уравнению (7) его разрешимость в целых числах.

Алгоритмическая неразрешимость 10-й Проблемы Гильберта установлена Ю. В. Матиясевичем в работах [14, 15]. Тем самым им были успешно завершены усилия многих математиков, из которых наиболее весомый вклад в решение проблемы внесли Д. Робинсон, М. Девис и Х. Путнам.

Вернемся к криптографии. Обычно криптографическая система основывается на трудноразрешимой математической проблеме, часто теоретико-числового характера. Приведём список наиболее популярных проблем такого сорта и покажем, что с каждой из таких проблем можно связать диофантово уравнение таким образом, что любое решение этого уравнения даёт возможность эффективно выписать решение соответствующей проблемы и наоборот, решение проблемы приводит к эффективному решению соответствующего диофантова уравнения. Так как любое конечное множество диофантовых уравнений, более того, любое множество диофантовых уравнений от ограниченного числа переменных, которое, как известно из теоремы Гильберта, эквивалентно своей конечной подсистеме, равносильно одному диофантову уравнению, которое легко получается из левых частей уравнений вида (7), если взять квадраты этих левых частей и приравнять их сумму к 0, то можно эффективно сопоставлять любому конечному множеству проблем, о которых говорилось выше, одну равносильную им проблему — о разрешимости в целых числах полученного диофантова уравнения.

Приведённое рассуждение показывает, что диофантов язык является универсальным в определённом смысле. Он может быть применён для криптографических функций многих известных систем шифрования, в том числе для функции шифрования системы RSA, функции дискретного логарифма и т. п. Перечислим некоторые из упомянутых проблем и покажем, как записать соответствующие им системы диофантовых уравнений.

1. **Разложение на множители.** Для данного составного числа n найти натуральные числа $p, q \geqslant 2$, такие, что $n = p \cdot q$.

Во многих приложениях p и q — различные нечётные простые числа. Пусть

$$\mathbb{Z}^{(2)} = \{n = pq : p, q \text{ различные нечётные простые числа}\}.$$

Так как любое натуральное число, согласно теореме Лагранжа, допускает представление в виде суммы квадратов четырёх неотрицательных целых чисел, проблема разложения на множители числа $n \in \mathbb{Z}^{(2)}$, да и любого нечётного составного числа, равносильна разрешимости в целых числах диофантова уравнения

$$(\zeta_1^2 + \zeta_2^2 + \zeta_3^2 + \zeta_4^2 + 3)(\zeta_5^2 + \zeta_6^2 + \zeta_7^2 + \zeta_8^2 + 3) = n.$$

Проблема разложения на множители изучается уже сотни лет. Разрабатываются различные методы разложения, такие, как метод квадратичного решета и метод, использующий эллиптические кривые. Имеются впечатляющие разложения больших составных чисел, осуществлённые с помощью мощных параллельных вычислений. Однако до сих пор не найден эффективный алгоритм для её решения в целом. Это даёт основание считать, что проблема действительно трудна.

2. **Проблема расшифрования в RSA.** Пусть $n = pq \in \mathbb{Z}^{(2)}$. Кольцо вычетов $\mathbb{Z}_n$ используется в системе шифрования RSA в качестве платформы шифрования. Мультипликативная группа $\mathbb{Z}_n^*$ кольца $\mathbb{Z}_n$ имеет порядок $\varphi(n) = (p-1)(q-1)$, где $\varphi(n)$ обозначает функцию Эйлера. Предполагается, что натуральное число e, взаимно простое с $\varphi(n)$, выбрано как ключ шифрования. Проблема расшифрования заключается в нахождении вычета $x \in \mathbb{Z}_n$, кодирующего исходный текст, по его зашифрованному

виду $c = x^e \bmod n$. Эта проблема равносильна разрешимости диофантова уравнения $x^e = c + ny$ относительно неизвестных x и y.

Проблема расшифрования относительно системы RSA изучается последние три десятка лет, но эффективный алгоритм для её решения пока не найден.

3. **Проблема квадратичного вычета.** Пусть $n = pq \in \mathbb{Z}^{(2)}$. Для произвольного вычета $a \in \mathbb{Z}_n$ определить, существует ли вычет $x \in \mathbb{Z}_n$, такой, что справедливо сравнение $x^2 = a(\mathrm{mod} n)$.

Проблема квадратичного вычета равносильна разрешимости диофантова уравнения $x^2 = a + ny$.

Проблема квадратичного вычета лежит в основе системы шифрования Гольдвассер — Микали, на ней базируется семантическая секретность систем Накаче — Штерна и Бекалоха.

Вычисление квадратичного корня по модулю числа $n \in \mathbb{Z}^{(2)}$ при знании разложения $n = pq$ осуществляется за полиномиальное время. В общем случае проблема так же трудна, как и проблема разложения n на множители.

4. **Дискретный логарифм в простом конечном поле.** Пусть p — простое число, тогда $\mathbb{Z}_p$ — простое конечное поле характеристики p. Мультипликативная группа $\mathbb{Z}_p^*$ циклическая. Пусть g — порождающий элемент этой группы. *Дискретным логарифмом* элемента $f \in \mathbb{Z}_p^*$ называется число x, для которого $g^x = f(\mathrm{mod} p)$.

Число x определяется по модулю $(p-1)$ — порядка группы $\mathbb{Z}_p^*$. *Проблемой дискретного логарифма* в поле $\mathbb{Z}_p$ называется вопрос о вычислении x по случайно выбранному элементу f; g считается известным. Пишут $x = \log_g(f)$ и называют его *дискретным логарифмом* элемента f по основанию g. Для однозначности вычисления x накладывается дополнительное ограничение $0 \leqslant x \leqslant p-2$. Проблема дискретного логарифма в простом конечном поле $\mathbb{Z}_p$ равносильна разрешимости *экспоненциального диофантова уравнения*

$$g^x = f + py. \tag{8}$$

Проблема дискретного логарифма может рассматриваться для любого конечного поля, где также может быть записана эквивалентным экспоненциальным диофантовым уравнением. Здесь мы не говорим об этом более подробно.

Из результатов Ю. В. Матиясевича [14, 15] следует, что множество E всех решений x, y уравнения (8) является диофантовым. В общем случае *диофантовым* называется множество наборов $\bar{a} = (a_1, \ldots, a_k)$ целых чисел, для которого существует диофантов многочлен $d(\zeta_1, \ldots, \zeta_k, \varkappa_1, \ldots, \varkappa_n)$, такой, что набор целых чисел $\bar{a} = (a_1, \ldots, a_k)$ принадлежит множеству E в том и только в том случае, если найдутся целые значения $b_1, \ldots, b_n$ для $\varkappa_1, \ldots, \varkappa_n$, такие, что $d(a_1, \ldots, a_k, b_1, \ldots, b_n) = 0$, то есть соответствующее диофантово уравнение разрешимо в целых числах. По теореме Матиясевича любое рекурсивно перечислимое множество E наборов из k целых чисел является диофантовым множеством. Существование такой характеризации рекурсивно перечислимых множеств даёт ещё большее основание говорить о диофантовом языке как об универсальном и рассматривать его в качестве средства построения криптографических систем и протоколов. О явном виде диофантова уравнения, равносильного уравнению вида (8), см. в [55, 56].

Трудность вычисления дискретного логарифма в произвольном случае лежит в основе многочисленного семейства систем шифрования и протоколов, таких, как система ЭльГамаля, протоколы Масси — Омуры и Диффи — Хеллмана, стандарты цифровой подписи DSS и ГОСТ Р 34.10-94, основанные на базовом протоколе ЭльГамаля, и т. д.

Есть ещё одно важное обстоятельство, говорящее в пользу использования неразрешимости 10-й Проблемы Гильберта в качестве основы для построения криптографических примитивов, систем и протоколов. Как недавно показал А. Н. Рыбалов [59], 10-я Проблема Гильберта остаётся неразрешимой на любом строго генерическом множестве диофантовых уравнений.

Каждый диофантов многочлен $d(\zeta_1, \ldots, \zeta_k)$ может рассматриваться как функция из $\mathbb{Z}^k$ в $\mathbb{Z}$. Ю. В. Матиясевич [14, 15] доказал, что существует такой диофантов многочлен $d_0(\zeta_1, \ldots, \zeta_k)$, что не существует алгоритма нахождения решения в целых числах уже в классе уравнений вида $d_0(\zeta_1, \ldots, \zeta_k) = c$, где $c \in \mathbb{Z}$. Таким образом можно определить функцию $\mathbb{Z}^k \to \mathbb{Z}$, которая может рассматриваться в качестве кандидата на одностороннюю функцию. Действительно, значение данного диофантова многочлена вычисляется за полиномиальное время, в то время как не существует полиномиального ограничения на нахождение аргумента по значению функции $d_0(\zeta_1, \ldots, \zeta_k)$.

Односторонней называется функция, эффективно вычислимая за полиномиальное время на детерминированной машине Тьюринга, для которой не существует вероятностной машины Тьюринга, вычисляющей за полиномиальное время по значению функции один из соответствующих этому значению аргументов с существенной вероятностью. Формального определения здесь не приводим, тем более что часто в определении присутствуют дополнительные требования (см. по этому поводу [57]).

Односторонние функции являются неотъемлемой частью криптографических систем и протоколов. Теоретически их существование ещё не доказано. В то же время ряд функций, которые претендуют на то, чтобы считаться односторонними, широко используются в криптографии. Нам представляется, что диофантовы функции дают широкие возможности для построения возможно односторонних функций, тем более что, как показано выше, ряд кандидатов на роль односторонних функций могут трактоваться как диофантовы.

С. Ю. Ерофеев в работе [58] рассмотрел различные варианты построения не только возможно односторонних, но и возможно двушагово односторонних функций. *Двушагово односторонней* называется композиция двух односторонних функций, которая сама является односторонней. Более того, по значению композиции и известному аргументу второй из функций аргумент самой композиции не является эффективно вычислимым. Необходимость построения таких функций объясняется тем, что в ряде предлагаемых протоколов, например в протоколе аутентификации с нулевым разглашением из [13], наличия обычных односторонних функций недостаточно.

5. Свободные метабелевы группы и проблема эндоморфной сводимости

Этот раздел связан с базовыми понятиями теории групп (о них см., например, [32, 33] и другие книги, излагающие основы теории групп). Элементы теории групповых многообразий см., например, в [10].

5.1. Свободные метабелевы группы

Напомним, что для произвольного натурального числа n через F_n обозначается свободная группа ранга n. Её фактор-группа по коммутанту $A_n = F_n/F_n'$ является свободной абелевой группой ранга n, а фактор-группа по второму коммутанту $M_n = F_n/F_n''$ — свободной метабелевой группой ранга n. При этом второй коммутант G'' определяется как коммутант от коммутанта $(G')'$. Он также является нормальной подгруппой группы G. Фактор-группа G/G'' по второму коммутанту является метабелевой группой. В общем случае *метабелевой* называется группа, в которой существует цепочка

нормальных подгрупп

$$1 \leqslant A \leqslant G \tag{9}$$

с абелевыми факторами. Другими словами, в группе G есть абелева нормальная подгруппа A, фактор-группа по которой G/A также абелева. Легко проверяется, что группа G метабелева в том и только в том случае, если её коммутант G' абелев. В определении (9) коммутант G' играет роль A.

Заметим, что фактор-группа M_n/M_n' также изоморфна свободной абелевой группе A_n.

Зафиксируем канонические гомоморфизмы $\pi_n' : F_n \to A_n$, $\pi_n'' : F_n \to M_n$, $\pi_n : M_n \to A_n$. Если $(f_1, \ldots, f_n)$ — *базис*, то есть *множество свободных порождающих* группы F_n, то $(a_1, \ldots, a_n) = (f_1F_n', \ldots, f_nF_n')$ — соответствующий базис группы A_n и $(x_1, \ldots, x_n) = (f_1F_n'', \ldots, f_nF_n'')$ — базис группы M_n. Имеем также $(a_1, \ldots, a_n) = (\pi_n(x_1), \ldots, \pi_n(x_n))$.

Группы A_n и M_n являются свободными группами многообразия $\mathcal{A}$ всех абелевых групп и многообразия $\mathcal{A}^2$ всех метабелевых групп соответственно. Любое отображение базиса группы A_n в произвольную абелеву группу A и любое отображение базиса группы M_n в произвольную метабелеву группу M однозначно продолжается до гомоморфизма $A_n \to A$ и $M_n \to M$ соответственно.

Перейдём к описанию структуры свободной метабелевой группы M_n. Как уже отмечалось, фактор-группа M_n/M_n' изоморфна свободной абелевой группе A_n. Это означает, что любой элемент группы M_n однозначно представим в виде

$$g = \prod_{i=1}^{n} x_i^{k_i} u, \tag{10}$$

где $k_i \in \mathbb{Z}$ для $i = 1, \ldots, n$; элемент $u = u(g)$ принадлежит коммутанту M_n'. Чтобы получить из (10) нормальную форму записи элемента g, достаточно построить такую форму для элемента u.

Так как M_n' — нормальная абелева подгруппа группы M_n, её можно рассматривать как модуль над групповым кольцом $\mathbb{Z}A_n$. Групповая операция в этом модуле — это умножение в группе M_n. Действие элемента $a \in A_n$ на элемент $v \in M_n'$ определяется следующим образом. Берём произвольный прообраз $\bar{a}$ элемента a в группе M_n относительно канонического гомоморфизма π_n и полагаем

$$v^a = v^{\bar{a}}. \tag{11}$$

Так как все возможные прообразы $\bar{a}$ отличаются друг от друга на элементы из M_n', сопряжение в (11) не зависит от выбора $\bar{a}$. Следовательно, приведённое определение корректно. Продолжение действия на групповое кольцо $\mathbb{Z}A_n$ осуществляется по линейности, а именно: для любого набора элементов $b_j \in A_n$ и любого набора целых чисел $l_j \in \mathbb{Z}$ $(j = 1, \ldots, p)$ полагаем

$$v^{\sum\limits_{j=1}^{p} l_j b_j} = \prod_{j=1}^{p} (v^{b_j})^{l_j}, \tag{12}$$

причём правая часть в (12) не зависит от порядка сомножителей, что обеспечивает корректность определения.

Коммутант M_n' более естественно рассматривать как модуль над $\mathbb{Z}A_n$ по следующим причинам. Как подгруппа, M_n' является свободной абелевой группой, ранг которой бесконечен. Следовательно, такое представление не является конечным. Как

модуль, M'_n конечно порождён. В качестве его порождающих элементов можно взять набор коммутаторов вида

$$e_{ij} = [x_i, x_j] \text{ для } i > j;\ i, j = 1, \ldots, n. \tag{13}$$

Покажем, как произвольное слово $v = v(x_1, \ldots, x_n)$, записывающее элемент коммутанта M'_n, представляется через порождающие модуля (13). Сначала переставляем влево все вхождения $x_1^{\pm 1}$, пользуясь формулами

$$x_i x_1 = [x_i, x_1] x_1 x_i, \quad x_i^{-1} x_1 = [x_i^{-1}, x_1] x_1 x_i^{-1} = [x_i, x_1]^{-a_i^{-1}} x_1 x_i^{-1}, \quad x_i x_1^{-1} = [x_i, x_1^{-1}] x_1^{-1} x_i =$$
$$= [x_i, x_1]^{-a_1^{-1}} x_1^{-1} x_i, \quad x_i^{-1} x_1^{-1} = [x_i^{-1}, x_1^{-1}] x_1^{-1} x_i^{-1} = [x_i, x_1]^{a_1^{-1} a_i^{-1}} x_1^{-1} x_i^{-1}.$$

Возникающие при этом коммутаторы $e_{ij} = [x_i, x_1]$ передвигаются вправо согласно формуле $[x_i, x_1] f = f [x_i, x_1]^{f^{-1}}$, справедливой при любом $i = 2, \ldots, n$ и любом $f \in M_n$. После того как будут переставлены все вхождения $x_1^{\pm 1}$, произойдёт сокращение этих степеней и таких вхождений уже не будет. Продолжим процесс переписки, переводя влево $x_2^{\pm 1}$, и т. д. В результате получим выражение вида

$$v = \prod_{\substack{i>j,\\ i,j=1,\ldots,n}} e_{ij}^{\alpha_{ij}} \tag{14}$$

для некоторых элементов α_{ij} группового кольца $\mathbb{Z}A_n$.

При $l < t$ считаем, что e_{lt} обозначает коммутатор $[x_l, x_t] = [x_t, x_l]^{-1} = e_{tl}^{-1}$. Модуль M'_n не является свободным. Относительно порождающих (13) все его определяющие соотношения следуют из следующих соотношений Якоби:

$$e_{ij}^{a_k - 1} e_{jk}^{a_i - 1} e_{ki}^{a_j - 1} = 1. \tag{15}$$

Ввиду этих соотношений форма (14) записи элемента v не является однозначной. Для получения однозначной (нормальной) формы записи элемента $v \in M'_n$ воспользуемся следующими соображениями. Для любых $m < n$ будем считать группу A_m естественно вложенной подгруппой группы A_n.

Утверждение 1. Любой элемент коммутанта M'_n однозначно представим в виде

$$v = \prod_{\substack{i>j,\\ i,j=1,\ldots,n}} e_{ij}^{\alpha_{ij}}, \tag{16}$$

где $\alpha_{ij} \in \mathbb{Z}A_i$ для $i > j; i, j = 1, \ldots, n$.

Доказательство. Пусть элемент v записан в форме (14). Покажем сначала, как можно исключить в этой записи элементы $a_i^{\pm 1}$ для $i > 2$ из показателя модульного порождающего e_{21}. Начинаем с исключения $a_3^{\pm 1}$. Достаточно рассмотреть случай, когда в показателе стоит элемент группы A_n вида $a_3^k h$, где h не зависит от a_3. Пусть $k > 0$. Тогда $a_3^k h = (a_3 - 1) a_3^{k-1} h + a_3^{k-1} h$. Отсюда и из соотношений (15) получаем равенство

$$e_{21}^{a_3^k h} = e_{21}^{a_3^{k-1} h} e_{31}^{(a_2 - 1) a_3^{k-1} h} e_{32}^{-(a_3 - 1) a_3^{k-1} h}.$$

Далее продолжаем процесс указанным способом до полного исключения a_3^k.

Пусть $k < 0$. Равенства (15) остаются справедливыми, если в них заменить x_3 и a_3 соответственно на x_3^{-1} и на a_3^{-1}. Следовательно, можем исключить из рассматриваемого показателя a_3^k, как это делалось выше, а затем использовать равенства

$$[x_3^{-1}, x_i] = [x_3, x_i]^{-a_3^{-1}} \text{ для } i = 1, 2.$$

Подобным образом исключим из показателя элемента e_{21} все элементы вида $a_i^{\pm 1}$ для $i = 3, \ldots, n$. Оставшийся показатель принадлежит групповому кольцу $\mathbb{Z}A_2$. Далее исключаем подобным образом $a_i^{\pm 1}$ для $i \geqslant 4$ из показателей степеней при e_{31} и e_{32}. В конце концов получим запись элемента v в форме (16). Докажем, что полученная запись единственная. Сначала применим к (16) гомоморфизм специализации группы M_n, определённый на базисных элементах как $x_i \mapsto x_i$ для $i = 1, 2$ и $x_j \mapsto 1$ для $j \geqslant 3$. Образом элемента v будет $e_{21}^{\alpha_{21}}$. Так как все соотношения в модуле M_n' следуют из соотношений Якоби, подмодуль, порождённый в M_n' любым из элементов e_{ij}, свободен. Значит, показатель α_{21} определяется однозначно.

Далее рассматриваем элемент $v' = e_{21}^{-\alpha_{21}} v$ и применяем к нему гомоморфизм специализации, определенный отображением $x_i \mapsto x_i$ для $i = 1, 2, 3$, и $x_j \mapsto 1$ для $j \geqslant 4$. Образом элемента v' будет $e_{31}^{\alpha_{31}} e_{32}^{\alpha_{32}}$. Так как все соотношения в модуле M_n' следуют из соотношений Якоби, подмодуль, порождённый в M_n' набором элементов $e_{i1}, \ldots, e_{ii-1}$, является свободным на этих порождающих. Значит, показатели α_{31} и α_{32} определяются по элементу v однозначно. Продолжая доказательство подобным образом, установим однозначность записи (16). ■

Группа M_n допускает также точное представление матрицами над полем. Оно получается из следующего знаменитого точного матричного представления, известного как *вложение Магнуса*. Относительно этого представления см., например, монографию [60].

Пусть T_n обозначает свободный модуль ранга n над групповым кольцом $\mathbb{Z}A_n$ с базисом $t_1, \ldots, t_n$. Рассмотрим группу матриц $M(A_n, T_n)$ вида

$$\begin{pmatrix} a & \sum\limits_{i=1}^{n} \alpha_i t_i \\ 0 & 1 \end{pmatrix}, \tag{17}$$

где $a \in A_n, \alpha_i \in \mathbb{Z}A_n$ для $i = 1, \ldots, n$.

Группа $M(A_n, T_n)$ является прямым сплетением группы A_n на себя, то есть $M(A_n, T_n) = A_n \wr A_n$. Относительно определения и свойств конструкции сплетения групп см., например, [32]. Легко проверяется, что группа $M(A_n, T_n)$ метабелева. Все матрицы вида (17) с 1 в левом верхнем углу образуют в ней абелеву нормальную подгруппу N, фактор-группа по которой изоморфна группе A_n. Группа N является модулем над групповым кольцом $\mathbb{Z}A_n$ с базисом $t_1, \ldots, t_n$. Можно заметить, что модульная операция индуцируется сопряжениями элементами группы $M(A_n, T_n)$ аналогично тому, как это объяснялось выше относительно модуля M_n'.

Вложение Магнуса группы M_n в группу $M(A_n, T_n)$ определяется следующим отображением базиса:

$$\mu : x_i \mapsto \begin{pmatrix} a_i & t_i \\ 0 & 1 \end{pmatrix} \text{ для } i = 1, \ldots, n. \tag{18}$$

Вложение Магнуса также позволяет ввести нормальную форму элементов группы M_n как матриц вида (18). Важно отметить для этого, что матрица вида (17) принадлежит образу $\mu(M_n)$ тогда и только тогда, когда выполнено равенство

$$a - 1 = \sum_{i=1}^{n} \alpha_i (a_i - 1). \tag{19}$$

Для того чтобы считать группу $M(A_n, T_n)$ матричной над полем, возьмем поле частных $\mathbb{F}$ группового кольца $\mathbb{Z}A_n$ и его чисто трансцендентное расширение

$\bar{\mathbb{F}} = \mathbb{F}(t_1, \ldots, t_n)$. Тогда группа $M(A_n, T_n)$ оказывается подгруппой полной матричной группы $GL_2(\bar{\mathbb{F}})$.

Вложение Магнуса и его обобщения широко используются в теории групп для доказательства важных утверждений не только о свободных метабелевых группах, но и о группах из широкого класса групп вида F/R', то есть фактор-групп по коммутантам нормальных подгрупп свободных групп F. Имеется прямая связь с другим важным понятием теории групп — свободными дифференцированиями Фокса (см., например, монографию [60]).

Например, с помощью вложения Магнуса легко доказать, что моноид X_n^*, порождённый в группе M_n множеством свободных порождающих $X_n = \{x_1, \ldots, x_n\}$, свободен. Это позволяет, например, записывать элементами этого моноида слова в произвольном алфавите из n букв. Можно использовать вместо X_n^* его автоморфную копию или произвести еще какие-нибудь операции, позволяющие скрыть вид слов. Это даёт возможности для построения криптографических примитивов, а затем на их основе систем и протоколов.

Есть ещё ряд обстоятельств, говорящих в пользу групп M_n как возможных платформ для криптографических систем. Во-первых, в группе M_n при любом n проблема равенства разрешима за полиномиальное время. Более точно, в [61] показано, что проблема равенства в группе M_n решается за время $\mathrm{O}(nm \log_2 m)$, где m — длина слова. Там же указан соответствующий алгоритм. В то же время близкая по постановке алгоритмическая проблема вычисления геодезической длины элемента, по которой для данного слова группы M_n нужно найти длину его кратчайшей записи от элементов базиса, является NP-полной. В [62] установлено, что проблема сопряжённости в группе M_n решается за время $\mathrm{O}(nm^8)$, где m обозначает сумму длин двух слов, для которых проверяется сопряжённость записываемых ими элементов группы M_n. Доказательство в [62] конструктивно, что позволяет дать точно такую же оценку времени решения соответствующей проблемы поиска сопряжённого элемента. Приведённые здесь результаты говорят о том, что можно эффективно работать с элементами групп M_n, их нормальными формами и сопряжёнными элементами. Известны также [63] эффективные алгоритмы, решающие в группах M_n проблему вхождения.

Важно также отметить, что группа M_n при $n \geqslant 2$ имеет экспоненциальный рост. Это означает, что функция роста количества элементов длины $\leqslant r$ относительно словарной метрики группы M_n экспоненциальна. Значит, если рассматривать группу M_n при $n \geqslant 2$ в качестве источника параметров, ключей и т. п., соответствующее пространство будет достаточно обширно, чтобы его нельзя было атаковать методом полного перебора.

В заключение отметим, что классические алгоритмические проблемы равенства, сопряжённости и вхождения разрешимы в любой конечно порождённой метабелевой группе. Проблема равенства решена Е. И. Тимошенко [64]. Разработанный им алгоритм, сводящий проблему к решению системы линейных уравнений над групповым кольцом $\mathbb{Z}A_n$ свободной абелевой группы, вполне пригоден для практического использования. Проблема сопряжённости решена Г. А. Носковым [65], но представленный им алгоритм довольно сложен. Он опирается на структурную теорию коммутативных колец и, по-видимому, в такой форме вряд ли может использоваться практически. В работе [66] описан алгоритм, решающий в конечно порождённой метабелевой группе более общую, чем проблема сопряжённости, проблему скрученной сопряжённости. По представлению автора, такой более общий подход даёт возможность его практического использования, по крайней мере, на генерическом множестве. Проблема вхож-

дения решена Н. С. Романовским [63]. На практичность она не исследована, но, по-видимому, может быть реализована для такой цели. Разрешимы и многие другие алгоритмические проблемы [67] (см. соответствующий обзор в [25]). Правда, с точки зрения теории сложности общий случай ещё исследован недостаточно. В классе всех конечно порождённых метабелевых групп разрешима проблема изоморфизма данной свободной метабелевой группе M_n [68]. Разрешимость проблемы изоморфизма в этом классе остается открытым вопросом (см. по этому поводу [69]).

5.2. Уравнения в группах

Итак, классические алгоритмические проблемы в свободных метабелевых группах разрешимы. Однако некоторые другие естественные по своей постановке алгоритмические проблемы в этих группах неразрешимы. Первые такие примеры указаны автором в [11, 12]. Это алгоритмические проблемы разрешимости произвольного уравнения в группе M_n при $n \geqslant 2$ и разрешимость проблемы эндоморфной сводимости в группе M_n достаточно большого ранга $n \geqslant n_0$. Для постановки этих проблем и объяснения их неразрешимости необходимо провести специальную подготовку, к которой мы сейчас переходим.

Пусть G — группа. *Нижним центральным рядом* группы G называется убывающая последовательность нормальных подгрупп

$$G = \gamma_1 G \geqslant \gamma_2 G \geqslant \dots \gamma_k G \geqslant \dots, \tag{20}$$

в которой $\gamma_{i+1}G = [\gamma_i G, G]$ — подгруппа, порождённая всеми коммутаторами вида $[g, f]$, где $g \in \gamma_i G, f \in G$. Ряд (20) называется *центральным*, так как любой его фактор $\gamma_i G/\gamma_{i+1}G$ лежит в центре фактора $G/\gamma_{i+1}G$. Группа G называется *нильпотентной*, если для некоторого i в ней $\gamma_{i+1}G = 1$. Наименьшее число i с этим свойством называется *ступенью нильпотентности* группы G. Считается, что тривиальная группа имеет ступень нильпотентности 0, нетривиальная абелева группа — ступень нильпотентности 1 и т. д.

Выпишем известные (см., например, [32]) коммутаторные тождества, справедливые в любой группе G:

$$\begin{gathered}[xy, z] = [y, z]^x [x, z] = [[y, z], x]^{-1} [y, z][x, z], \\ [x^{-1}, z] = [x, z]^{-1} [[x, z], x^{-1}], \\ [z, x^{-1}] = [[x, z], x^{-1}]^{-1} [x, z].\end{gathered} \tag{21}$$

Одним из следствий тождеств (21) является следующая относительная дистрибутивность:

$$[g_1^{k_1}, \dots, g_l^{k_l}] = [g_1, \dots, g_l]^{\prod_{i=1}^{l} k_i} (\mathrm{mod}\, \gamma_{l+1} G), \tag{22}$$

где $g_1, \dots, g_l$ — произвольные элементы группы G, а $k_1, \dots, k_l$ — целые числа.

Предполагаем, что в левой части стоит *простой* коммутатор, т. е. коммутатор вида $[\dots[[a_1, a_2], a_3], \dots, a_l]$, в котором скобки стоят слева направо. Такие коммутаторы ещё называют *левонормированными*. Однако равенство (22) остаётся верным при любой расстановке скобок, важно только, чтобы все операции с символами были коммутаторными.

Известно [10, 32], что в метабелевых группах выполняется тождество

$$[f, h, g_1, g_2, \dots, g_l] = [f, h, g_{\sigma(1)}, \dots, g_{\sigma(l)}],$$

где σ — произвольная перестановка символов $1, \ldots, l$.

Для свободных порождающих $x_1, \ldots, x_n$ группы M_n определяются *базисные коммутаторы* — простые коммутаторы от элементов базиса $x_1, \ldots, x_n$ вида

$$[x_{i_1}, x_{i_2}, x_{i_3}, \ldots, x_{i_l}], \tag{23}$$

где $l \geqslant 2$ называется *весом* коммутатора, и выполняются неравенства $i_1 > i_2$; $i_2 \leqslant i_3 \leqslant \ldots \leqslant i_l$. Сами порождающие $x_1, \ldots, x_n$ также считаются базисными коммутаторами веса 1. Известно [70, 71], что образы базисных коммутаторов веса l относительно канонического гомоморфизма $M_n \to G/\gamma_{i+1}M_n$ образуют базис свободной абелевой группы $\gamma_i M_n/\gamma_{i+1}M_n$.

Упорядочим все базисные коммутаторы группы M_n по возрастанию весов. Продолжим этот частичный порядок до полного, упорядочив между собой базисные коммутаторы одного веса произвольным образом. Пусть $c_1, \ldots, c_t$, $t = t(i)$ — полный список всех базисных коммутаторов веса не больше чем i в заданном порядке. Обычно считают, что $c_j = x_j$ для $j = 1, \ldots, n$, то есть что порядок на порождающих элементах как базисных коммутаторах веса 1 соответствует порядку на индексах. Тогда любой элемент группы M_n при $n \geqslant 2$ для любого $i \geqslant 1$ однозначно записывается в виде

$$g = \prod_{j=1}^{t} c_j^{k_j} (\mathrm{mod} \gamma_{i+1}M_n) \text{ для некоторых } k_j \in \mathbb{Z}. \tag{24}$$

Таким образом, по модулю $\gamma_{i+1}M_n$ элементы группы M_n кодируются наборами целых чисел $(k_1, \ldots, k_t)$, $t = t(i)$. Компоненты набора будем называть *координатами* элемента g по модулю $\gamma_{i+1}M_n$. Легко видеть, что координаты при различных i соответствуют друг другу в очевидном смысле.

Любое отображение группы M_n на себя, рассматриваемое по модулю $\gamma_{i+1}M_n$, определяет отображение $\mathbb{Z}^t \to \mathbb{Z}^t$, и наоборот. Можно ввести в рассмотрение свободную метабелеву нильпотентную ступени $i+1$ группу $M_{n,i} = M_n/\gamma_{i+1}M_n$ и говорить о нормальной форме её элементов, соответствующей (24), её отображениях и т. п. Группа $M_{n,i}$ является свободной группой ранга n многообразия всех метабелевых нильпотентных ступени $\leqslant i$ групп, которое есть пересечение многообразия $\mathcal{A}^2$ всех метабелевых групп и многообразия $\mathcal{N}_i$ всех нильпотентных групп ступени $\leqslant i$.

Пусть два элемента g и f группы M_n записаны в виде (24), причем элемент g имеет координаты $(k_1, \ldots, k_t)$, а элемент f — координаты $(q_1, \ldots, q_t)$. Тогда существуют многочлены $p_j = p_j(\zeta_1, \ldots, \zeta_{t(j)}, \varkappa_1, \ldots, \varkappa_{t(j)})$ для $j = 1, \ldots, t$, такие, что координаты $(r_1, \ldots, r_t)$ произведения $h = gf$ вычисляются по формулам

$$r_j = p_j(k_1, \ldots, k_{t(j)}, q_1, \ldots, q_{t(j)}). \tag{25}$$

Напомним, что $t(j)$ обозначает количество всех базисных коммутаторов веса не больше чем j.

Аналогично, существуют многочлены $u_j(\zeta_1, \ldots, \zeta_t)$ для $j = 1, \ldots, t$, вычисляющие по координатам элемента g координаты обратного к нему элемента g^{-1}. Теория базисных коммутаторов введена в рассмотрение Ф. Холлом более 50 лет назад и получила широкое применение в теории групп (см. лекции Ф. Холла [70] и монографию [71]).

Важно отметить, что в рассматриваемом случае групповые операции переписываются через диофантовы функции. Это позволяет переходить от групповых уравнений к диофантовым. Вопрос о разрешимости группового уравнения таким образом сводится к решению соответствующего диофантова уравнения. Здесь также проявляется универсальность диофантова языка. Рассмотрим этот вопрос более подробно.

Уравнением в группе G (ещё говорят «над группой G»), или *групповым уравнением*, называется выражение вида

$$w = w(z_1, \ldots, z_r) = 1, \tag{26}$$

где w — групповое слово от неизвестных $z_1, \ldots, z_r$ и элементов группы G. Если ввести в рассмотрение свободную группу F с базисом $\{z_1, \ldots, z_r, \ldots\}$, то w может рассматриваться как элемент свободного произведения $G[z_1, \ldots, z_r, \ldots] = F * G$. *Решением* уравнения (26) называется набор элементов $g_1, \ldots, g_r$ группы G, для которого $w(g_1, \ldots, g_r) = 1$. Решению $g_1, \ldots, g_r$ соответствует гомоморфизм $\varphi : G[z_1, \ldots, z_r, \ldots] \to G$, при котором элементы группы G отображаются сами на себя, на группе F гомоморфизм определяется своими значениями $z_i \mapsto g_i$ при $i = 1, \ldots, r$, и произвольными значениями остальных порождающих z_j для $j \geqslant r+1$. Наоборот, каждому гомоморфизму φ группы $G[z_1, \ldots, z_r, \ldots]$ в группу G, тождественному на элементах группы G, отвечает решение $g_i = \varphi(z_i)$, $(i = 1, \ldots, r)$. Уравнение (26) называется *разрешимым*, если для него существует хотя бы одно решение, и *неразрешимым* в противном случае. Соответственно ставится следующая алгоритмическая проблема.

Проблема разрешимости уравнений. Разрешимо ли произвольное уравнение (26) в данной группе G.

Проблема разрешимости уравнений может ставиться также для класса групп $\mathcal{L}$ — вопрос о существовании алгоритма, который по произвольной группе G из данного класса $\mathcal{L}$ и произвольному уравнению (26) определяет его разрешимость в G. Можно рассматривать более широкую проблему *разрешимости систем уравнений* в группе или в классе групп. Можно, напротив, ограничивать класс рассматриваемых уравнений. Например, брать уравнения от ограниченного числа переменных или уравнения специального вида. Важным классом являются так называемые *бескоэффициентные* уравнения вида

$$w(z_1, \ldots, z_r) = f, \tag{27}$$

где левая часть не зависит от элементов группы G (*коэффициентов*), а в правой части стоит элемент f группы G.

Для любого натурального числа i сопоставим уравнению (26) *относительное уравнение*

$$w(z_1, \ldots, z_r) = 1 (\mathrm{mod} \gamma_{i+1} G), \tag{28}$$

для которого естественно определяется понятие *разрешимости*. Ясно, что разрешимость уравнения (26) влечет разрешимость уравнения (28) для любого i. Обратное утверждение в общем случае неверно. Разрешимость уравнения (28) в группе G равносильно разрешимости его гомоморфного образа в группе $G/\gamma_{i+1}G$. Под *гомоморфным образом* понимается уравнение, полученное из исходного уравнения заменой всех вхождений элементов из G на их канонические гомоморфные образы в фактор-группе $G/\gamma_{i+1}G$.

Итак, разрешимость относительных уравнений в группе равносильна разрешимости уравнений в фактор-группе. Выше рассмотрены относительные уравнения по модулю членов нижнего центрального ряда $\gamma_{i+1}G$ группы G. Однако ясно, что можно брать относительные уравнения по модулю любой нормальной подгруппы N группы G и связывать с ними уравнения в фактор-группе G/N.

Свободные метабелевы группы обладают так называемым свойством *эллиптичности*, или *конечности ширины* вербальных подгрупп, которое позволяет, в частности, относительным уравнениям (28) в группе $G = M_n$ сопоставлять равносильные им

уравнения в самой группе M_n. Напомним, что *вербальной* подгруппой $v(G)$ группы G, соответствующей групповому слову $v = v(z_1, \ldots, z_r)$, называется подгруппа, порождённая всеми *значениями* $v(g_1, \ldots, g_r)$ слова v в группе G. Подгруппа $v(G)$ имеет *конечную ширину* $l = \mathrm{width}(v(G))$, если любой элемент $u \in v(G)$ представим как произведение не более l значений слова v или обратного к нему v^{-1} в группе G, и l — минимальное число с этим свойством. Если такого числа l не существует, то говорят, что подгруппа $v(G)$ имеет *бесконечную ширину*.

Известно [72] (доказательство можно найти также в [73]), что любая вербальная подгруппа свободной метабелевой группы M_n при любом n имеет конечную ширину. Следовательно, любой член нижнего центрального ряда $\gamma_i M_n$ имеет конечную ширину $l_{n,i} = \mathrm{width}(\gamma_i M_n)$ относительно слова $v_i = v_i(z_1, \ldots, z_i) = [z_1, z_2, \ldots, z_i]$. Более точно, в работе [74] показано, что $l_{n,2} = [n/2]$ при $n \geqslant 2$ и $l_{n,i} = n$ при $i \geqslant 3$.

Это означает, что если взять слово

$$V_{n,i} = \prod_{j=1}^{l_{n,i}} [z_{j,1}^{(1)}, z_{j,2}^{(1)}, \ldots, z_{j,i}^{(1)}][z_{j,1}^{(2)}, z_{j,2}^{(2)}, \ldots, z_{j,i}^{(2)}]^{-1}$$

от независимых переменных $z_{j,k}^{(t)}$ $(j = 1, \ldots, l_{n,i};\ k = 1, \ldots, i;\ t = 1, 2)$, то любой элемент $u \in \gamma_i M_n$ представляется как значение этого слова в группе M_n. Значит, уравнение

$$u = \prod_{j=1}^{l_{n,i}} [z_{j,1}^{(1)}, z_{j,2}^{(1)}, \ldots, z_{j,i}^{(1)}][z_{j,1}^{(2)}, z_{j,2}^{(2)}, \ldots, z_{j,i}^{(2)}]$$

разрешимо в группе M_n тогда и только тогда, когда элемент u принадлежит подгруппе $\gamma_i M_n$. Отсюда получаем, что относительное уравнение (28) разрешимо в группе M_n тогда и только тогда, когда в M_n разрешимо уравнение

$$w(z_1, \ldots, z_r) V_{n,i} = 1.$$

Разрешимость относительного бескоэффициентного уравнения (27) равносильна разрешимости бескоэффициентного уравнения

$$w(z_1, \ldots, z_r) V_{n,i} = f.$$

Рассмотрим относительное уравнение

$$w(z_1, \ldots, z_r) = 1 (\mathrm{mod}\, \gamma_{i+1} G). \tag{29}$$

Перепишем в нормальной форме (24) все константы, входящие в запись (29), и все неизвестные $z_1, \ldots, z_r$, полагая $z_k = \prod_{j=1}^{t} c_j^{\zeta_{k,j}} (\mathrm{mod}\, \gamma_{i+1})$, с неизвестными показателями степеней $\zeta_{k,j}$ $(k = 1, \ldots, r;\ j = 1, \ldots, t)$, где $t = t(i)$. Другими словами, запишем константы и неизвестные в координатной форме. Затем приведём левую часть уравнения к нормальному виду (24), используя многочлены (25). Решению уравнения (29) соответствует тривиальная нормальная форма. Значит, все показатели полученной нормальной формы должны равняться нулю. Разрешимость уравнения (29) сводится таким образом к разрешимости системы из $t = t(i)$ диофантовых уравнений. В свою очередь, любая конечная система диофантовых уравнений равносильна одному диофантову уравнению.

Равносильность относительных и обычных уравнений в группе M_n позволяет получать те же системы диофантовых уравнений и для уравнений в группе M_n.

Остаётся установить, что класс получающихся диофантовых уравнений достаточно широк с точки зрения его алгоритмической разрешимости, а именно, что этот класс алгоритмически неразрешим. Это сделано в работах автора [11, 12]. А именно, в [12] показано, что по любому диофантову уравнению (7) можно явно указать такое групповое слово $w(z_1, \ldots, z_r)$, не зависящее от констант, и такой элемент f группы M_n для произвольного $n \geqslant 2$, что уравнение (27) разрешимо в группе M_n тогда и только тогда, когда диофантово уравнение (7) разрешимо в целых числах. Более того, можно зафиксировать левую часть уравнения (27), а элемент f выбирать из фиксированного смежного класса по циклической подгруппе $\mathrm{гр}(h)$ группы M_n. Отсюда и из неразрешимости 10-й Проблемы Гильберта следует неразрешимость проблемы разрешимости уравнений в любой свободной метабелевой нециклической группе. Более того, эта проблема неразрешима уже для класса бескоэффициентных уравнений с фиксированной левой частью, правая часть которых берется из фиксированного смежного класса по циклической подгруппе, как это объяснено выше.

Перейдем теперь к проблеме эндоморфной сводимости. Для произвольной группы G она ставится следующим образом.

Проблема эндоморфной сводимости. Определить по произвольным элементам g и f группы G, является ли элемент f эндоморфным образом элемента g при каком-нибудь эндоморфизме группы G.

Если рассмотреть свободную метабелеву группу M бесконечного счётного ранга с базисом $\{x_1, \ldots, x_r, \ldots\}$, то неразрешимость проблемы эндоморфной сводимости в ней вытекает из неразрешимости проблемы разрешимости бескоэффициентных уравнений в M_2. Действительно, для произвольных элементов $g = g(x_1, \ldots, x_r)$ и $f = f(x_1, \ldots, x_r)$ группы M элемент f является эндоморфным образом элемента g тогда и только тогда, когда он является таковым, если рассматривать группу M_r с базисом $\{x_1, \ldots, x_r\}$. В свою очередь, это равносильно разрешимости в M_r уравнения

$$g(z_1, \ldots, z_r) = f.$$

Как отмечено выше, эта проблема неразрешима уже в группе M_2. Число неизвестных также можно ограничить, поскольку известно [75], что 10-я Проблема Гильберта неразрешима уже для класса диофантовых уравнений от 9 переменных. Значит, для достаточно большого r проблема эндоморфной сводимости в группе M_r алгоритмически неразрешима.

Есть несколько возможностей построения диофантовых функций на свободных метабелевых группах. Произвольный эндоморфизм μ группы M_n однозначно определяется своими значениями на элементах базиса. Можно вести рассуждения по модулю $\gamma_{i+1}M_n$. Тогда этим значениям $\mu(x_i)$ для $i = 1, \ldots, n$ взаимно однозначно соответствуют наборы целых чисел из (24). Наоборот, любой набор из n таких наборов определяет некоторый эндоморфизм группы M_n, рассматриваемый по модулю $\gamma_{i+1}M_n$. Эндоморфизм, в свою очередь, является, с одной стороны, отображением группы M_n в себя, с другой стороны, если его рассматривать по модулю $\gamma_{i+1}M_n$, может считаться отображением множества $\mathbb{Z}^t$ в себя. Это отображение в свете существования диофантовых функций (25) является диофантовым отображением. Семейство таких отображений, среди которых, как известно, есть такие, что проблема нахождения для них прообразов по данному значению алгоритмически неразрешима, даёт богатую возможность

построения функций, претендующих на роль односторонних. Другие возможности связаны с рассмотрением других нормальных форм элементов группы M_n, о которых говорилось выше. В следующей работе мы приведём детали подобных построений.

ЛИТЕРАТУРА

1. *Myasnikov A., Shpilrain V., and Ushakov A.* Group-based cryptography. (Advances courses in Math., CRM, Barselona). Basel, Berlin, New York: Birkhäuser Verlag, 2008. 183 p.
2. *Myasnikov A., Shpilrain V., and Ushakov A.* Non-commutative cryptography and complexity of group-theoretic problems. (Amer. Math. Soc. Surveys and Monographs). Providence, RI: Amer. Math. Soc., 2011. 385 p.
3. `www.grouptheory.info/PersonalPages/Shpilrain,Vladimir/CryptologyePrintArchive`
4. *Anshel I., Anshel M., and Goldfeld D.* An algebraic method for public-key cryptography // Math. Res. Lett. 1999. V. 6. P. 287–291.
5. *Марков А. А.* Основы алгебраической теории кос // Труды Матем. ин-та АН СССР. 1945. Т. 16. С. 3–54.
6. *Dehornoy P.* Braids and self-distributivity. (Progress in Math. 192). Basel, Berlin, New York: Birkhäuser Verlag, 2000. 623 p.
7. *Лин В. Я.* Косы Артина и связанные с ними группы и пространства // Итоги науки и техн. Алгебра. Геометрия. Топология. Т. 17. М.: ВИНИТИ, 1983. С. 159–227.
8. *Dehornoy P.* Braid-based cryptography // Group theory, statistics and cryptography. Contemp. Math. V. 360. Providence, RI: Amer. Math. Soc., 2004. P. 5–33.
9. *Mahlburg K.* An overview of braid group cryptography. 2004. `www.math.wisc.edu/~boston/mahlburg.pdf`
10. *Нейман Х.* Многообразия групп. М.: Мир, 1974. 264 с.
11. *Романьков В. А.* О неразрешимости проблемы эндоморфной сводимости в свободных нильпотентных группах и в свободных кольцах // Алгебра и логика. 1977. Т. 16. № 4. С. 457–471.
12. *Романьков В. А.* Об уравнениях в свободных метабелевых группах // Сиб. матем. ж. 1979. Т. 40. № 3. С. 671–673.
13. *Grigoriev D. and Shpilrain V.* Zero-knowledge autentification schemes from actions on graphs, groups and rings // Ann. Pure Appl. Logic. 2010. V. 162. P. 194–200.
14. *Матиясевич Ю. В.* Диофантовость перечислимых множеств // Докл. АН СССР. 1970. Т. 191. № 2. С. 279–282.
15. *Матиясевич Ю. В.* Диофантово представление перечислимых предикатов // Изв. АН СССР. Сер. матем. 1971. Т. 35. № 1. С. 3–30.
16. *Матиясевич Ю. В.* Десятая проблема Гильберта. М.: Наука, 1993. 223 с.
17. *Shpilrain V. and Zapata G.* Using decision problems in public key cryptography // Groups. Complexity. Cryptology. 2009. V. 1. P. 33–40.
18. *Shpilrain V. and Zapata G.* Using the subgroup membership problem in public key cryptography // Contemp. Math. V. 418. Providence, RI: Amer. Math. Soc., 2006. P. 169–179.
19. *Shpilrain V. and Zapata G.* Combinatorial group theory and public key cryptography // Applicable Algebra in Engineering Communication and Computing. 2006. V. 17. P. 291–302.
20. *Kurt Y.* A new key exchange primitive based on the triple decomposition problem // Preprint: `http://eprint.iacr.org/2006/378`
21. *Birget J.-C., Magliveras S., and Sramka M.* On public-key cryptosystems based on combinatorial group theory // Tatra Mountains Math. Publ. 2006. V. 33. P. 137–148.

22. *Новиков П. С.* Об алгоритмической неразрешимости проблемы тождества слов в теории групп // Труды Матем. ин-та АН СССР. 1955. Т. 44. С. 3–143.
23. *Новиков П. С.* Неразрешимость проблемы сопряженности в теории групп // Изв. АН СССР. Сер. матем. 1954. Т. 18. № 6. С. 485–524.
24. *Адян С. И.* Неразрешимость некоторых алгоритмических проблем в теории групп // Труды Моск. матем. общества. 1957. Т. 6. С. 231–298.
25. *Ремесленников В. Н., Романьков В. А.* Теоретико-модельные и алгоритмические вопросы теории групп // Итоги науки и техн. Алгебра. Геометрия. Топология. Т. 21. М.: ВИНИТИ, 1983. С. 3–79.
26. *Адян С. И., Дурнев В. Г.* Алгоритмические проблемы для групп и полугрупп // Успехи матем. наук. 2000. Т. 55. С. 3–94.
27. *Miller III C. F.* Decision problems for groups — survey and reflections // Algorithms and Classification in Combinatorial Group Theory. Berlin, Heidelberg, New York: Springer Verlag, 1992. P. 1–60.
28. *Sims C. C.* Computation with finitely presented groups. Cambridge: Cambridge Univ. Press, 1994. 604 p.
29. *Holt D. F., Eick B., and O'Brien E. A.* Handbook of computational group theory. London: Chapman & Hall/CRC, 2005. 414 p.
30. *Detinko A., Eick B., and Flannery D.* Computing with matrix groups // London Math. Soc. Lect. Notes Ser. 2011. V. 387. P. 256–270.
31. *Михайлова К. А.* Проблема вхождения для прямых произведений групп // Докл. АН СССР. 1958. Т. 119. С. 1103–1105.
32. *Каргаполов М. И., Мерзляков Ю. И.* Основы теории групп. М.: Лань, 2009. 288 с.
33. *Курош А. Г.* Теория групп. М.: Физматгиз, 2008. 808 с.
34. *Garside F. A.* The braid group and other groups // Quart. J. Math. 1969. V. 20. No. 78. P. 235–254.
35. *Gebhardt V.* Conjugacy search in braid groups from a braid-based cryptography point of view // Applicable Algebra in Engineering Communication and Computing. 2006. V. 17. P. 219–238.
36. *Gebhardt V.* A new approach to the conjugacy problem in Garside groups // J. Algebra. 2005. V. 292. P. 282–302.
37. *Petrides G.* Cryptoanalisis of the public key cryptosystem based on the word problem on the Grigorchuk groups // LNCS. 2003. V. 2898. P. 234–244.
38. *Magyarik M. R. and Wagner N. P.* A public key cryptosystem based on the word problem // LNCS. 1985. V. 196. P. 19–36.
39. *Fine B., Habeeb M., Kahrobaei D., and Rosenberger G.* Survey and open problems in non-commutative cryptography // JP J. Algebra, Number Theory and Applications. 2011. V. 21. P. 1–40.
40. *Papadimitriou C.* Computation complexity. Boston: Addison-Wesley, 1994. 523 p.
41. Computational complexity theory / eds. S. Rudich and A. Wigderson. Amer. Math. Soc. Institute for Advanced Study, IAS/Park City Math. Series. V. 10. Providence, RI: Amer. Math. Soc., 2004. 389 p.
42. *Gurevich Y.* Average case comleteness // J. Comput. Syst. Sci. 1991. V. 42. P. 346–398.
43. *Levin L.* Average case complete problems // SIAM J. Comput. 1986. V. 15. P. 285–286.
44. *Kapovich I., Myasnikov A., Shupp P., and Shpilrain V.* Average-case complexity and decision problems in group theory // Adv. Math. 2005. V. 190. P. 343–359.

45. *Kapovich I., Myasnikov A., Shupp P., and Shpilrain V.* Generic-case complexity, decision problems in group theory and random walks // J. Algebra. 2003. V. 264. P. 665–694.
46. *Celler F., Leedham-Green C. R., Murray S. H., et al.* Generating random elements of a finite group // Comm. Algebra. 1995. V. 23. No. 3. P. 4931–4948.
47. *Borovik A., Myasnikov A., and Shpilrain V.* Measuring sets in infinite groups // Contemp. Math. V. 298. Providence, RI: Amer. Math. Soc., 2002. P. 21–42.
48. *Klee V. and Minty G.* How good is the simplex algorithm? // Inequalities. Proc. Third. Symp., Univ. California. California: Academic Press, 1972. P. 159–175.
49. *Хачиян Л. А.* Полиномиальный алгоритм в линейном программировании // Докл. АН СССР. 1979. Т. 244. № 5. С. 1093–1096.
50. *Вершик А. М., Спорышев П. В.* Ограничение среднего числа шагов в симплекс-методе и проблемы асимптотической интегральной геометрии // Докл. АН СССР. Т. 271. № 5. С. 1044–1048.
51. *Smale S.* On the average number of steps of the simplex method of linear programming // Math. Programming. 1983. V. 27. P. 241–262.
52. *Gilman R., Myasnikov A. G., Miasnikov A. D., and Ushakov A.* Report on generic case complexity // Вестник Омского университета. Спец. вып.: Комбинаторные методы алгебры и сложность вычислений. 2007. С. 103–110.
53. *Cook S. A. and Mitchell D. G.* Finding hard instances of the satisfiability problem: A survey // Satisfiability problem: Theory and Applications. V. 35. Providence, RI: Amer. Math. Soc., 1997. P. 1–17.
54. *Kellerer H., Pferschy U., and Pisinger D.* Knapsack Problems. Berlin, New York: Springer Verlag, 2004. 546 p.
55. *Ерофеев С. Ю.* Диофантовость дискретного логарифма // Прикладная дискретная математика. Приложение. 2011. № 4. С. 31–32.
56. *Ерофеев С. Ю.* Диофантовость дискретного логарифма // Вестник Омского университета. 2010. № 4. С. 13–15.
57. *Goldreich O.* Foundations of cryptography. Cambridge: Cambridge Univ. Press., 2001. V 1. 451 p.; 2004. V. 2. 798 p.
58. *Ерофеев С. Ю.* Схемы построения двушагово односторонних функций // Вестник Омского университета. 2011. № 4. С. 15–18.
59. *Рыбалов А. Н.* О генерической неразрешимости десятой проблемы Гильберта // Вестник Омского университета. 2011. № 4. С. 19–22.
60. *Тимошенко Е. И.* Эндоморфизмы и универсальные теории разрешимых групп. Новосибирск: Изд-во НГТУ, 2011. 327 с.
61. *Myasnikov A., Roman'kov V., Ushakov A., and Vershik A.* The word and geodesic problems in free solvable groups // Trans. Amer. Math. Soc. 2010. V. 362. P. 4655–4682.
62. *Vassileva S.* Polynomial time conjugacy in wreath products and free solvable groups // Groups. Complexity. Cryptology. 2011. V. 3. P. 105–120.
63. *Романовский Н. С.* О некоторых алгоритмических проблемах для разрешимых групп // Алгебра и логика. 1974. Т. 13. № 1. С. 26–34.
64. *Тимошенко Е. И.* Алгоритмические проблемы для метабелевых групп // Алгебра и логика. 1973. Т. 12. № 2. С. 232–240.
65. *Носков Г. А.* О сопряженности в метабелевых группах // Математические заметки. 1982. Т. 31. № 4. С. 495–507.
66. *Вентура Э., Романьков В. А.* Проблема скрученной сопряженности для эндоморфизмов метабелевых групп // Алгебра и логика. 2009. Т. 48. № 2. С. 157–173.

67. *Baumslag G., Cannonito F., and Robinson D. J. S.* The algorithmic theory of finitely generated metabelian groups // Trans. Amer. Math. Soc. 1994. V. 344. P. 629–648.
68. *Groves J. R. J. and Miller III C. F.* Recognizing free metabelian groups // Illinois J. Math. 1986. V. 30. No. 2. P. 246–254.
69. *Baumslag G., Mikhailov R., and Orr K. E.* A new look at finitely generated metabelian groups // arXiv math. Group Theory. 2012. No. 1203.5431. 17 p.
70. *Hall P.* Nilpotent groups // Canad. Math. Cong. Summer Sem. Vankuver: University of Alberta, 1957. P. 12–30.
71. *Холл М.* Теория групп. М.: ИЛ, 1962. 467 с.
72. *Stroud P.* Ph. D. Thesis. Cambridge, 1966. 121 p.
73. *Segal D.* Words: notes on verbal width in groups // London Math. Soc. Lect. Notes. V. 361. Cambridge: Cambridge Univ. Press, 2009. 215 p.
74. *Алламбергенов Х. С., Романьков В. А.* Произведения коммутаторов в группах // Докл. АН Уз. ССР. 1984. Т. 4. С. 14–15.
75. *Jones J.* Universal diophantine equation // J. Symbolic Logic. 1982. V. 47. No. 3. P. 549–571.

2012 Математические методы криптографии №2(16)

DOI 10.17223/20710410/16/3

УДК 681.326; 531.19

МАТЕМАТИЧЕСКАЯ МОДЕЛЬ ГЕНЕРАТОРА СЛУЧАЙНЫХ ЧИСЕЛ НА ОСНОВЕ ТРЁХЗНАЧНОЙ ЛОГИКИ

Е. Л. Столов

Казанский федеральный университет, г. Казань, Россия

E-mail: yevgeni.stolov@ksu.ru

Предложена математическая модель физического генератора случайных чисел, реализованного в виде кольцевого соединения комбинационных схем, работающих в трёхзначной логике. Доказана равномерность распределения сигнала, снимаемого с любого выхода.

Ключевые слова: *асинхронный генератор, трёхзначная логика, марковский процесс.*

Введение

Увеличение производительности цифровых устройств является одной из основных задач современной схемотехники. Достигнутая тактовая частота уже близка к предельной, распараллеливание пригодно не для любой задачи. В этой связи внимание исследователей снова обращается к схемам, использующим трёхзначную логику. В последнее время созданы элементарные физические устройства, реализующие указанную логику (см., например, работу [1]), где говорится о разработке технологии, позволяющей реализовать произвольную комбинационную схему, работающую в троичной логике. Физические генераторы случайных последовательностей дают примеры устройств, применение в которых k-значных логик позволяет увеличить их производительность, поскольку в любой момент времени снимаемый сигнал имеет большую длину по сравнению с аналогичной схемой, работающей в двоичной логике. Упомянутые генераторы используются для создания криптографических ключей. В работах [2, 3] рассмотрен цифровой стохастический генератор бинарных последовательностей, составленный из сумматоров по модулю 2 с обратными связями. В предлагаемой работе рассматриваются аналогичные устройства, только вместо сумматора используются блоки с двумя входами и одним выходом, реализующие комбинационную схему, работающую в трёхзначной логике. Пример подобного генератора представлен на рис. 1. Здесь символом F обозначен упомянутый блок, а сигнал снимается с выхода любого из блоков.

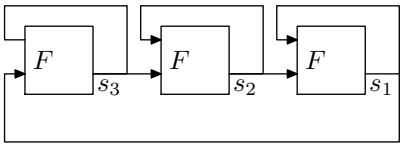


Рис. 1. Пример генератора тернарных последовательностей

При создании указанных генераторов возникают следующие проблемы:

1) возможность перехода генератора в стационарное состояние;
2) доказательство независимости сгенерированных символов.

После перехода генератора в стационарное состояние снимаемый с его выхода сигнал не меняется во времени. Ниже показано, что при надлежащем выборе блока F генератор не имеет стационарных состояний. Что касается независимости символов генерируемой последовательности, то о ней можно говорить лишь после того, как будут выбраны математическая модель функционирования отдельных блоков и способ их соединения. Кроме того, потребуем равномерности распределения выходного сигнала на выходе блока. Это требование является естественным, поскольку появляется возможность преобразования выходного сигнала в сигнал с произвольным распределением.

1. Математическая модель генератора

Общий подход к исследованию генератора, составленного из нелинейных блоков, представлен в работе [4], однако применение трёхзначной логики вносит некоторые упрощения в описание ситуации. Каждый из блоков реализует некоторую функцию $c = F(a, b)$, $a, b, c \in \{0, 1, 2\}$. При изменении входных сигналов блок срабатывает, реализуя функцию F. Относительно срабатывания блоков сделаны следующие предположения:

1) время срабатывания блока является случайной величиной с экспоненциальным распределением, то есть вероятность того, что после поступления изменённого сигнала на вход блока время изменения сигнала на выходе меньше T, равна $1 - \exp(-T)$;
2) срабатывания отдельных блоков являются независимыми событиями, причём никакие два блока не могут сработать одновременно.

Определение 1. Перенумеруем блоки генератора. Состоянием генератора в момент времени t называется вектор $\mathbf{s}(t)$, компонента которого $\mathbf{s}(t)[k], k = 1, \ldots, N$ есть сигнал на выходе блока с номером k в момент времени t.

В дальнейшем будем пользоваться векторным обозначением состояния в форме

$$\mathbf{s} = \langle s_1, s_2, \ldots, s_N \rangle.$$

Если генератор содержит N блоков, то число его состояний $M = 3^N$. Соединения блоков друг с другом задаются списком Con. Элемент списка $Con[k] = [i_k, j_k]$, где i_k, j_k — номера блоков, с выходов которых сигналы поступают на вход блока с номером k. Например, структура генератора, представленного на рис. 1, задается списком $Con = ([1, 2]; [2, 3]; [3, 1])$.

Наложенные ограничения аналогичны предположениям, накладываемым на системы массового обслуживания, благодаря чему функционирование генератора описывается уравнениями Эрланга [5]. Выпишем систему дифференциальных уравнений Эрланга, описывающих динамику генератора. Перенумеруем все состояния генератора числами от 1 до M. Пусть $\mathbf{s}_n$ — состояние с номером n, а $P_n(t)$ — вероятность того, что в момент времени t генератор находится в состоянии $\mathbf{s}_n$. Обозначим через $i_1, i_2, \ldots, i_m$ все номера состояний, свои для каждого n, из которых можно попасть в состояние $\mathbf{s}_n$ в результате срабатывания только одного блока. Вероятность того, что через момент Δt генератор окажется в состоянии $\mathbf{s}_n$, складывается из вероятности того, что генератор находился в этом состоянии прежде и ни один из N блоков не сработал, и из вероятностей перейти в состояние $\mathbf{s}_n$ из одного из состояний с номерами $i_1, i_2, \ldots, i_m$ в результате срабатывания только одного блока. Имеем

$$\frac{dP_n(t)}{dt} = -NP_n(t) + \sum_{k=1}^{m} P_{i_k}(t). \tag{1}$$

2. Выбор функции F и способа соединения блоков

При выводе уравнения (1) не делалось никаких предположений о виде функции F и способе соединения блоков между собой. Как отмечалось выше, нужно гарантировать отсутствие стационарных состояний генератора. Предположим, что функция F обладает следующим свойством:

$$c = F(a, b), \quad \forall a, b \quad (c \neq a, c \neq b). \tag{2}$$

Из условия (2) следует, что при срабатывании любого блока состояние генератора изменится при любом соединении блоков между собой. Таким образом, данное условие исключает наличие стационарных состояний. Перечислим все функции, обладающие свойством (2). Значение $F(a, b)$ определено однозначно, если $a \neq b$, поэтому $F(a, b) = F(b, a)$. Это означает, что достаточно определить функцию лишь для совпадающих аргументов. Если σ — произвольная перестановка чисел $0, 1, 2$, то функции $F(a, b)$ и $\sigma(F(\sigma(a), \sigma(b))$ будем считать неразличимыми. Отсюда следует, что существуют лишь две существенно разные функции, обладающие свойством (2):

	$F(0,0)$	$F(1,1)$	$F(2,2)$
F_1	1	2	0
F_2	1	2	1

Изучаемый генератор предназначен для генерации последовательностей, в которых каждый из элементов появляется с одной и той же вероятностью. В этой связи далее в качестве функции F будем использовать только F_1.

Остановимся на выборе соединения блоков друг с другом. Рассмотрим соединение, определенное списком $Con = ([1, 2], [2, 3], \ldots, [N - 1, N], [N, 1])$. На рис. 1 представлен указанный вид соединения для $N = 3$. Достоинством указанной схемы является равноправность всех выходов блоков и одинаковая электрическая нагрузка на выходе каждого блока. Далее показано, что схема обладает свойством «забывания» начального состояния, поэтому в силу отмеченной симметрии на выходе любого блока при снятии сигнала через достаточно большой интервал времени T_0 генерируется тернарная последовательность, в которой каждый символ появляется с одной и той же вероятностью.

3. Матрица переходов генератора

Обозначим через A матрицу размера $M \times M$ переходов генератора. Эта матрица состоит из нулей и единиц, причём $A[i, k] = 1$ тогда и только тогда, когда при срабатывании какого-либо блока генератор переходит из состояния с номером i в состояние с номером k. Изучим особенности матрицы переходов исследуемого генератора. Рассмотрим произвольное состояние

$$\mathbf{s} = \langle s_1, \ldots, s_k, \ldots, s_N \rangle. \tag{3}$$

В силу свойства (2) в результате срабатывания любого из N блоков состояние генератора изменится, причем все получившиеся состояния будут разными. Это означает, что каждая строка матрицы A имеет ровно N единиц.

Утверждение 1. Состояния (3), в которых все сигналы равны между собой, при указанных выборе функции F и способе соединения блоков недостижимы из других состояний генератора.

Доказательство. Рассмотрим состояние $\mathbf{s_0} = \langle 0, \ldots, 0\rangle$. Если из некоторого состояния $\mathbf{s_1}$ возможен переход в состояние $\mathbf{s_0}$ в результате срабатывания одного блока, то все компоненты вектора $\mathbf{s_1}$, кроме одной, равны 0. Легко видеть, что после срабатывания любого блока, в силу (2), переход в состояние $\mathbf{s_0}$ невозможен. Поскольку в рассматриваемой схеме все троичные значения равноправны, аналогичные утверждения справедливы для векторов $\langle 1, \ldots, 1\rangle$ и $\langle 2, \ldots, 2\rangle$. ■

Из доказанного утверждения следует, что столбцы с номерами, отвечающими трем недостижимым состояниям, являются нулевыми. Удалим из A эти три столбца и три столбца с теми же номерами и обозначим через A' получившуюся матрицу.

Напомним некоторые известные факты из теории матриц.

Определение 2 [6]. Матрица B с неотрицательными элементами называется разложимой, если существует матрица перестановки P, такая, что

$$P^{\mathrm{T}} \cdot B \cdot P = \begin{pmatrix} B_{11} & 0 \\ B_{21} & B_{22} \end{pmatrix}.$$

В противном случае матрица называется неразложимой.

Согласно [6, с. 166–168], неотрицательная матрица B неразложима тогда и только тогда, когда для любой пары индексов i, j существует натуральное число k, такое, что $B^k[i, j] > 0$. Кроме того, наибольшее характеристическое число r неразложимой матрицы является простым корнем характеристического многочлена этой матрицы.

Свойства генерируемой последовательности базируются на следующей теореме.

Теорема 1. Матрица A' является неразложимой.

Доказательство. Под множеством состояний будем понимать множество всех состояний генератора, кроме трёх отмеченных недостижимых состояний. Достаточно доказать, что из любого состояния можно перейти в любое другое. Пусть $\mathbf{s_1} = \langle 1, 0, \ldots, 0\rangle$. Покажем, что из $\mathbf{s_1}$ можно перейти в любое другое состояние. После двукратного срабатывания второго блока генератор перейдет в состояние $\langle 1, 1, 0, \ldots, 0\rangle$, а затем — в состояние $\langle 1, 2, 0, \ldots, 0\rangle$. Итак, из $\mathbf{s_1}$ можно перейти в состояние $\langle 1, a, 0, \ldots, 0\rangle$, где a — любой элемент множества $L = \{0, 1, 2\}$. При срабатывании первого блока переходим из состояния $\mathbf{s_1}$ в состояние $\langle 2, 0, 0, \ldots, 0\rangle$. После этого, как и выше, доказывается, что достигается любое состояние $\langle 2, a, 0, \ldots, 0\rangle$, где $a \in L$. Предположим, что уже доказана достижимость из состояния $\mathbf{s_1}$ любого состояния вида $\langle 1, a_2, a_3, \ldots, a_k, 0, \ldots, 0\rangle$ или $\langle 2, a_2, a_3, \ldots, a_k, 0, \ldots, 0\rangle$, где $a_2, a_2, \ldots, a_k$ — любые элементы множества L. Если $k < N-1$, то при срабатывании блока с номером $k+1$ из состояния $\langle 1, a_2, a_3, \ldots, a_k, 0, 0, \ldots, 0\rangle$ переходим в состояние $\langle 1, a_2, a_3, \ldots, a_k, 1, 0, \ldots, 0\rangle$, а затем — в $\langle 1, a_2, a_3, \ldots, a_k, 2, 0, \ldots, 0\rangle$. Если $k = N-1$, то при срабатывании блока с номером N из состояния $\langle 1, a_2, a_3, \ldots, a_k, 0\rangle$ переходим в состояние $\langle 1, a_2, a_3, \ldots, a_k, 2\rangle$, а из состояния $\langle 2, a_3, \ldots, a_k, 0\rangle$ — в $\langle 2, a_2, a_3, \ldots, a_k, 1\rangle$. После срабатывания первого блока переходим в состояние $\langle 2, 0, a_3, \ldots, a_k, 0\rangle$ и после срабатывания блока с номером N — в состояние $\langle 2, 0, a_3, \ldots, a_k, 1\rangle$. Наконец, покажем достижимость состояния вида $\langle 1, \ldots, 1, b_p, \ldots, b_{N-1}, 1\rangle$, где $b_p \neq 1$. Согласно предположению, достижимо состояние $\langle 2, 0, \ldots, 0, b_p, \ldots, b_{N-1}, 1\rangle$. После последовательного срабатывания блоков с номерами $1, 2, \ldots, p-1$ получим состояние $\langle 1, \ldots, 1, b_p, \ldots, b_{N-1}, 1\rangle$, поскольку $b_p = 0$ или $b_p = 2$.

Теперь покажем, что из любого состояния $\mathbf{s}$ можно перейти в состояние $\mathbf{s_1}$. При срабатывании первого блока происходит переход из состояния $\langle 0, a_2, \ldots, a_N\rangle$ в со-

стояние $\langle b, a_2, \ldots, a_N\rangle$, где $b = 1$ или $b = 2$. Это означает, что можем ограничиться состояниями, начинающимися с 1 или 2. Из состояния $\langle 1, 0, \ldots, 0, 1\rangle$ после двукратного срабатывания блока с номером N получаем состояния $\langle 1, 0, \ldots, 0, 2\rangle$, $\langle 1, 0, \ldots, 0, 0\rangle$. Из состояния $\langle 2, 0, \ldots, 0, 0\rangle$ после срабатывания блока с номером N получается состояние $\langle 2, 0, \ldots, 0, 1\rangle$, а после срабатывания первого блока — $\langle 1, 0, \ldots, 0, 1\rangle$. Из состояния $\langle 2, 0, \ldots, 0, 2\rangle$ после срабатывания блока с номером N получается состояние $\langle 2, 0, \ldots, 0, 0\rangle$. Это означает, что состояние $\mathbf{s_1}$ достижимо из состояний вида $\langle 1, 0, \ldots, 0, a\rangle$, $\langle 2, 0, \ldots, 0, a\rangle$ для любых $a \in L$. Пусть уже доказана достижимость $\mathbf{s_1}$ из состояний вида $\langle 1, 0, \ldots, 0, b_p, \ldots, b_N\rangle$ для любых $b_i \in L$. Рассмотрим произвольное состояние вида $\langle 1, 0, \ldots, 0, b_{p-1}, b_p, \ldots, b_N\rangle$, $b_{p-1} \neq 0$. Возможны следующие наборы значений b_{p-1}, b_p : (1,2), (2,1), (2,2), когда после срабатывания блока с номером $p-1$ получаем 0 в позиции $p-1$; (1,1), когда после двукратного срабатывания блока с номером $p-1$ получаем 0 в позиции $p-1$; (2,0), (1,0) — последняя ситуация требует дополнительного рассмотрения. Рассмотрим состояние $\langle 1, 0, \ldots, 0, 1, 0, b_{p+1}, \ldots, b_N\rangle$. При срабатывании блока с номером p в этой позиции появится 1 или 2, что сводит эту ситуацию к предыдущему случаю. Для завершения доказательства достаточно поменять местами значения 1 и 2. ■

4. Статистические свойства генерируемой последовательности

Введем обозначение $B = A^{\mathrm{T}}$. Согласно определению матрицы A, каждая строка матрицы B с номером i содержит 1 в позиции j тогда и только тогда, когда возможен переход из состояния с номером j в состояние с номером i в результате срабатывания одного блока. Рассмотрим более подробно систему уравнений (1). Положим $\mathbf{p}(t) = (P_1(t), \ldots, P_M(t))$. Указанная система может быть переписана в виде

$$\frac{d\mathbf{p}(t)}{dt} = (B - N \cdot I)\mathbf{p}(t) = D\mathbf{p}(t). \tag{4}$$

Решение данной системы имеет вид

$$\mathbf{p}(t) = \exp(Dt)\mathbf{e},$$

где $\mathbf{e}$ — произвольный стохастический вектор, определяющий начальное состояние. Матрица B имеет три нулевых строки; пусть это строки с номерами 1, 2, 3. Из (4) следует, что

$$P_k(t) = e_k \exp(-Nt), \quad k \in \{1, 2, 3\}, \tag{5}$$

где $0 \leqslant e_k \leqslant 1$. Отсюда вытекает, что $P_k(t) \to 0$ при $t \to \infty$, $k \in \{1, 2, 3\}$. Исключим из векторов компоненты с индексами 1, 2, 3, а из матриц — строки и столбцы с этими номерами. В результате получим векторы $\mathbf{e}'$, $\mathbf{p}'(t)$, $D' = A'^{\mathrm{T}} - N \cdot I'$, а решение системы (4) сведется к решению системы

$$\mathbf{p}'(t) = \exp(D't)\mathbf{e}'.$$

Теорема 2. Пусть $C(t) = \exp(Dt)$. Тогда $C(t) \to C_0$ при $t \to \infty$, где C_0 — матрица с одинаковыми столбцами, равными стохастическому вектору $\mathbf{d}$, такому, что $\mathbf{d} = (0, 0, 0, \mathbf{d}')^{\mathrm{T}}$, $A'^{\mathrm{T}}\mathbf{d}' = N\mathbf{d}'$.

Доказательство. Из (5) вытекает, что первые три строки матрицы C_0 нулевые. Сумма элементов в каждой строке матрицы A равна N, и при выбранной нумерации состояний её первые три столбца нулевые. Таким образом, матрица A'/N

стохастическая, её максимальное характеристическое число равно 1, все остальные характеристические числа имеют модули, не превосходящие 1 (см. [6, с. 200]), а из неразложимости этой матрицы вытекает, что 1 является простым корнем. Другими словами, если $\xi_1, \ldots, \xi_{M-3}$ — все характеристические числа матрицы A' и ξ_1 — максимальный по модулю корень, то $\xi_1 = N$, $|\xi_i| \leqslant N, i > 1$. По определению $D' = A'^{\mathrm{T}} - N \cdot I'$, поэтому для характеристических чисел μ_j этой матрицы выполнены условия $\mu_1 = 0$, $\mathrm{real}(\mu_i) < 0$, $i = 2, \ldots, M-3$. Характеристические числа матрицы $C'(t) = \exp(D't)$ равны $\exp(\mu_i t)$, поэтому существует предел $C'_0 = \lim\limits_{t\to\infty} C'(t)$ и матрица C'_0 имеет ранг 1. Если $A'^{\mathrm{T}}\mathbf{d}' = N\mathbf{d}'$, то $D'\mathbf{d}'$ — нулевой вектор, а из представления матрицы $C'(t)$ в виде ряда вытекает, что $C'(t)\mathbf{d}' = \mathbf{d}'$ для любого t. Это означает, что $C'_0\mathbf{d}' = \mathbf{d}'$. Далее, $(1, 1, \ldots, 1)A'^{\mathrm{T}} = N(1, 1, \ldots, 1)$, поэтому $(1, 1, \ldots, 1)D'$ — нулевой вектор и $(1, 1, \ldots, 1)C'_0 = (1, 1, \ldots, 1)$, т. е. сумма элементов в каждом столбце этой матрицы равна 1. ■

Вектор $\mathbf{d}'$ задает финальное распределение вероятностей состояний генератора. Из теоремы 2 следует независимость этого распределения от начального состояния, а финальные вероятности появления 0, 1 и 2 на выходе любого блока равны $1/3$. Следует, однако, заметить, что отсюда нельзя заключить, что финальные вероятности каждого из состояний генератора совпадают между собой.

5. Результаты численных экспериментов

При практическом использовании разработанного генератора возникает вопрос о скорости сходимости решения уравнения (4) к финальному вектору в зависимости от числа N блоков в схеме. В качестве меры близости выбрано среднеквадратическое отклонение δ финального вектора — собственного вектора матрицы D, отвечающего собственному значению 0, — от первого столбца матрицы $\exp(Dt)$. Результаты экспериментов представлены в следующей таблице.

Зависимость δ от N и t

$N\backslash t$	2	4	6	8	10
2	0,0095249	0,0001857	0,0000034	6,288e−08	1,141e−09
3	0,0026750	0,0000398	0,0000005	7,596e−09	1,102e−10
4	0,0014727	0,0000404	0,0000011	3,592e−08	1,113e−09
5	0,0004785	0,0000077	8,981e−08	9,856e−10	3,056e−11
6	0,0001991	0,0000055	0,0000002	6,546e−09	2,868e−10

Из приведённых результатов следует, что при любом N схема практически забывает своё начальное состояние при $t \geqslant 5$. Расчёт проведён для параметра экспоненциального распределения $\lambda = 1$.

Заключение

Рассмотренный в работе физический генератор, составленный из одинаковых блоков, реализующих специальную функцию трёхзначной логики, может использоваться для генерации случайных ключей. Генерация возникает за счёт наличия обратных связей и случайности времени срабатывания. Если время срабатывания блока определяется экспоненциальным распределением, то при одном и том же значении параметра распределения трёхзначный генератор обеспечивает большую производительность по сравнению с бинарной схемой, работающей в том же режиме. Выбранный способ соединения блоков обеспечивает одинаковую электрическую нагрузку на выходах всех устройств. Использование более чем трёх блоков не уменьшает время «забывания» начального состояния генератора.

ЛИТЕРАТУРА

1. *Sheng L., Yong-Bin K., and Lombardi F.* CNTFET-Based Design of Ternary Logic Gates and Arithmetic Circuits //IEEE Trans. Nanotechnology. 2011. V. 10. No. 2. P. 217–225.
2. *Кузнецов В. М., Песошин В. А., Столов Е. Л.* Марковская модель цифрового стохастического генератора //АиТ. 2008. № 9. С. 62–68.
3. *Кузнецов В. М., Песошин В. А., Столов Е. Л.* Стабильные состояния асинхронного генератора //Учёные записки Казанского государственного университета. 2010. Т. 152. Кн. 1. Сер. Физико-математические науки. С. 174–180.
4. *Столов Е. Л.* Генератор случайных чисел, составленный из нелинейных асинхронных элементов // Исследования по прикладной математике. Вып. 26. Казань: Институт информатики КГУ, 2006. С. 101–106.
5. *Хинчин А. Я.* Работы по математической теории массового обслуживания. М.: Физматгиз, 1963. 236 с.
6. *Маркус М., Минк Х.* Обзор по теории матриц и матричных неравенств. М.: Наука, 1972. 232 с.

МАТЕМАТИЧЕСКИЕ ОСНОВЫ КОМПЬЮТЕРНОЙ БЕЗОПАСНОСТИ

DOI 10.17223/20710410/16/4

УДК 004.056.52

СЕМАНТИЧЕСКАЯ РОЛЕВАЯ МОДЕЛЬ УПРАВЛЕНИЯ ДОСТУПОМ

Н. А. Семенова

Московский государственный институт электроники и математики (ТУ), г. Москва, Россия

E-mail: natasha_sem@inbox.ru

Приводится описание расширения ролевой модели $RBAC$, используемой для управления доступом в автоматизированную информационную систему (АИС) предприятия, за счёт введения семантического контекста, позволяющего отразить взаимосвязь между должностными обязанностями сотрудников и ролями, назначенными соответствующим им учётным записям пользователей в системе. Основное отличие модели — возможность автоматизации операций назначения и отзыва ролей. Приведено доказательство безопасности системы ролевого управления доступом, построенной на основе предложенной семантической модели.

Ключевые слова: *ролевое управление доступом, автоматизация управления ролями.*

Введение

Модель ролевого управления доступом $RBAC$ [1] применяется для построения систем управления доступом в АИС. Одной из особенностей классической модели $RBAC$ является то, что все операции по назначению и отзыву ролей выполняются вручную администраторами системы. При большом количестве ролей в АИС нагрузка на администраторов возрастает, что может привести к увеличению времени, необходимому для обеспечения пользователя АИС, соответствующего некоторому сотруднику предприятия, всеми нужными ему для выполнения должностных обязанностей правами доступа [2]. В данной работе предлагается ролевая модель управления доступом, основанная на классической модели и позволяющая автоматизировать назначение и отзыв ролей учётным записям пользователей при наступлении в системе ряда определённых событий.

1. Основные понятия и определения классической модели $RBAC$

Рассмотрим некоторые основные понятия классической модели $RBAC$, которые потребуются для дальнейшего описания [3]:

P — множество возможных прав доступа;

U — множество учётных записей пользователей системы;

$R \subseteq 2^P$ — множество ролей системы;

$UA : U \to 2^R$ — функция, задающая для каждой учетной записи пользователя множество ролей, на которые она может быть авторизована;

$PA : R \to 2^P$ — функция, задающая для каждой роли множество прав доступа. При этом для каждого права доступа $p \in P$ существует роль $r \in R$, такая, что $p \in PA(r)$;

$user : S \to U$ — функция, задающая для каждой сессии учётную запись, от имени которой она авторизована.

Определение 1. Иерархией ролей в классической модели ролевого управления доступом называется заданное на множестве ролей R отношение частичного порядка $\geqslant$. При этом по определению выполняется условие: для учётной записи пользователя $u \in U$ если $r_1, r_2 \in R$, $r_1 \in UA(u)$ и $r_1 \geqslant r_2$, то $r_2 \in UA(u)$.

Зададим $RH \subseteq R \times R$ — иерархическое отношение на множестве ролей — следующим образом: $(r_1 \geqslant r_2) \Leftrightarrow ((r_1, r_2) \in RH)$. Если $r_1 \geqslant r_2$, то r_1 считается старшей по отношению к r_2 (предком r_2), а r_2 — младшей (потомком r_1).

Определение 2. Два права доступа $p_1, p_2 \in P$ называются статически взаимоисключающими, если они не могут входить в состав одной и той же роли одновременно: для любой $r \in R$ выполняется неравенство $\{p_1, p_2\} \cap PA(r) \leqslant 1$.

Определение 3. Две роли $r_1, r_2 \in R$ называются статически взаимоисключающими, если они не могут быть назначены одной учётной записи пользователя одновременно: для любой $u \in U$ выполняется неравенство $\{r_1, r_2\} \cap UA(u) \leqslant 1$.

Определение 4. Множество предварительных условий CR — это множество, включающее в себя ограничения на роли, которыми должна обладать учётная запись пользователя, чтобы быть назначенной на новую роль, а также ограничения взаимного исключения ролей. Пусть $z_r : U \to \{\texttt{false}, \texttt{true}\}$ — функция, такая, что $z_r(u) = \texttt{true}$ тогда и только тогда, когда для $u \in U$ и $r \in R$ выполняется условие $r \in UA(u)$. Пусть $c_r : U \to \{\texttt{false}, \texttt{true}\}$ — функция, такая, что $c_r(u) = c_r(z_{r_1}(u), \ldots, z_{r_k}(u))$, где $u \in U$; $r, r_1, \ldots, r_k \in R$; $r \neq r_i$ для $i = 1, \ldots, k$ и $c_r(y_1, \ldots, y_k)$ — булева функция от k переменных. Тогда $CR = \{c_{r_1}, \ldots, c_{r_t}\}$ — множество функций, определяющих, при каких условиях той или иной учётной записи пользователя может быть назначена некоторая роль $r_j \in R$.

Для обеспечения возможности администрирования множества назначенных учётным записям пользователей ролей используем следующие функции, определённые в стандарте ARBAC'97 [4]:

- AP — множество административных прав доступа;
- $AR \subseteq 2^{AP}$ — множество административных ролей;
- $APA : AR \to 2^{AP}$ — функция, задающая для каждой административной роли множество административных прав доступа; при этом для каждого права доступа $p \in AP$ существует роль $r \in AR$, такая, что $p \in APA(r)$;
- $AUA : U \to 2^{AR}$ — функция, задающая для каждой учётной записи пользователя множество административных ролей, на которые она может быть авторизована;
- $can\_assign : AR \to CR \times 2^R$ — функция, определяющая для каждой административной роли множество ролей, которые могут быть назначены учётной записи пользователя с использованием данной административной роли при выполнении заданных предварительных условий CR;
- $can\_revoke : AR \to 2^R$ — функция, определяющая для каждой административной роли множество ролей, которые могут быть отозваны у учётной записи пользователя с использованием данной административной роли;
- $roles : S \to 2^R \cup AR$ — функция, задающая для учётной записи пользователя множество ролей, на которые она авторизована в данной сессии. При этом в каждый момент времени для каждой сессии $s \in S$ выполняется условие $roles(s) \in \in UA(user(s)) \cup AUA(user(s))$.

Определение 5. Под классической системой ролевого управления доступом будем понимать конечный автомат, заданный кортежем $K = \langle \Gamma, Q, a, \Psi \rangle$, где Γ — множество состояний; Q — множество запросов на доступ к объектам АИС; Ψ — множество действий, переводящих систему из состояния в состояние; $a : \Gamma \times Q \to \{\texttt{true}, \texttt{false}\}$ — функция, определяющая, является ли запрос истинным в заданном состоянии. Состояние системы задаётся кортежем $\gamma = (S, U, UA, user, roles) \in \Gamma$, а правила перехода между состояниями определяются предварительными условиями CR.

Таким образом, классическая модель $RBAC$ не содержит в себе информацию о связи между должностными обязанностями пользователей и ролями, назначенными их учётным записям (о семантическом смысле назначения ролей). Ещё одним важным ограничением классической модели $RBAC$ является то, что роли всегда назначаются и отзываются сессией, инициированной некоторой учётной записью пользователя системы, обладающей административными правами, и не могут быть назначены (отозваны) при наступлении заданного события, поскольку модель не содержит в себе средств описания событий и логики реакции на них. Далее описаны предлагаемые расширения классической модели за счёт введения семантического контекста.

2. Семантическая ролевая модель управления доступом

2.1. Иерархия ролей «по предусловию»

В классической модели $RBAC$ считается, что родительская роль содержит в себе все права дочерних ролей, а также более высокопривилегированные права доступа, чем дочерние (например, роль «Начальник отдела» является родительской для роли «Рядовой сотрудник» и содержит как все права доступа, назначенные рядовым сотрудникам, так и права, которыми обладают только руководители структурной единицы предприятия). В реальных АИС иерархические отношения между ролями обычно имеют следующий смысл [5]: меньшая роль описывает некоторые базовые права доступа к объекту, а родительские роли — более функциональные права доступа. В качестве примеров таких АИС можно привести службу каталогов *ActiveDirectory* [5], где базовым правом доступа является включение учётной записи в домен (роль *DomainUsers*), а также СУБД *Oracle* [6], содержащую базовые права доступа $CONNECT$ и $CREATESESSION$, разрешающие создание соединения с базой данных (роль *DatabaseUser*). Таким образом, родительские роли в общем случае не включают в себя права доступа дочерних ролей. В то же время можно показать, что в классической модели $RBAC$ существует неявно выраженная зависимость между предварительными условиями назначения ролей, находящихся в иерархической связи друг с другом. Пусть в классической модели $RBAC$ роль r_1 является старшей для роли r_2. Тогда, по определению 1, для учётной записи пользователя $u \in U$ если $(r_1, r_2) \in RH$, $r_1 \in UA(u)$ и $r_1 \geqslant r_2$, то $r_2 \in UA(u)$. В этом случае можно утверждать, что учётная запись пользователя u удовлетворяет условиям: если $c_{r_1}(u) = \texttt{true}$, то $c_{r_2}(u) = \texttt{true}$, в противном случае роль r_2 не могла бы быть назначена учётной записи пользователя. Таким образом, существует неявным образом выраженная зависимость между предварительными условиями назначения ролей r_1 и r_2.

Переопределим иерархические связи между ролями, используя описание множества предварительных условий CR.

Определение 6. Иерархией ролей «по предусловию» будем называть заданное на множестве ролей R отношение строгого порядка $>$. При этом для учётной записи пользователя $u \in U$ если $(r_1, r_2) \in RH$, $r_1 > r_2$ и $c_{r_1}(u) = \texttt{true}$, то $c_{r_2}(u) = \texttt{true}$ и

$c_{r_1}(u) = c_{r_1}(z_1(u), \ldots, c_{r_2}(u), \ldots, z_k(u))$. Роль r_1 будем называть предком r_2 «по предусловию».

Таким образом, по определению, для возможности авторизации на некоторую роль учётная запись пользователя должна удовлетворять и всем предварительным условиям для дочерней роли.

Предположение 1. В данной работе предполагается, что в модели действуют ограничения статического взаимного исключения прав доступа для ролей, состоящих в иерархических отношениях: для каждого множества ролей, состоящих в иерархии, каждое право доступа может входить в состав только одной роли: $P = PA(r_1) \cup \cdots \cup \cup PA(r_m)$, где $(r_i, r_j) \in RH$; $PA(r_i) \cap PA(r_j) = \varnothing$ для $1 \leqslant i < j \leqslant m$, $r_i, r_j \in R$.

Предположение 2. Будем считать, что если для некоторой административной роли $r_a \in AR$ и роли $r \in R$ выполняется условие $r \in can\_revoke(r_a)$, то для каждой роли $r_0 \in R$, такой, что $r_0 > r$, выполняется условие $r_0 \in can\_revoke(r_a)$.

Сформулируем ряд утверждений, описывающих свойства ролевой модели управления доступом, в которой выполняются условия предположений 1 и 2, а иерархия ролей строится по определению 6.

1) Отзыв у учётной записи пользователя родительской роли не влечёт за собой автоматического отзыва всех дочерних ролей, поскольку множества прав доступа, входящих в родительскую и дочерние роли, не пересекаются (не выполняется требование, справедливое для классической модели: если $r_1 \geqslant r_2$, то $PA(r_1) \supseteq PA(r_2)$).

2) Отзыв дочерней роли приводит к отзыву всех родительских ролей. По определению 6, при удалении у учётной записи пользователя u некоторой роли r для u перестают выполняться предварительные условия для всех ролей, родительских для r. Это означает, что в результате удаления r у учётной записи пользователя одновременно отзываются и все родительские для r роли. Для возможности каскадного удаления ролей у учётной записи пользователя необходимо выполнение условий предположения 2.

3) Каскадное назначение прав на отзыв административным ролям в направлении «снизу вверх» не влечёт за собой превышения прав доступа административных пользователей, поскольку родительские роли не включают в себя права доступа дочерних ролей.

Определение 7. Пусть $INIT : \Psi \to 2^U$ — функция, задающая для каждого действия $\sigma \in \Psi$ множество возможных инициаторов его выполнения. Говорим, что субъекты (учётные записи) из множества значений функции $INIT(\sigma) \subseteq U$ могут быть инициаторами действия $\sigma \in \Psi$, переводящего систему $\langle \Gamma, Q, a, \Psi \rangle$ из состояния $\gamma \in \Gamma$ в состояние $\gamma_g \in \Gamma$, если сессии, функционирующие от имени данных субъектов, могут выполнить действие $\sigma \in \Psi$, приводящее к переходу модели из состояния $\gamma \in \Gamma$ в состояние $\gamma_g \in \Gamma$, т. е. если существует действие $\sigma \in \Psi$, для которого выполняются следующие условия:

— $init(\sigma) \in INIT(\sigma)$, где $init : \Psi \to U$ — функция, такая, что $init(\sigma) = user(s)$, если сессия s выполнила действие σ;
— $\sigma_1(\gamma) = \gamma_g$.

Таким образом, для каждого возможного действия в системе значение функции $INIT$ — это совокупность учётных записей пользователей, обладающих в рамках заданной сессии s административными правами, позволяющими им выполнить данное действие. Будем использовать обозначение $INIT(\sigma_1, \sigma_2, \ldots, \sigma_i) = INIT(\sigma_1) \times \times INIT(\sigma_2) \times \cdots \times INIT(\sigma_i)$.

Пример 1. Множество учётных записей пользователей, обладающих возможностью назначить роль $r \in R$ учётной записи пользователя $u \in U$, задаётся выражением $\{u_a \in U : u_a \in AUA(can\_assign^{-1}(c_r(u), r))\}$, где $c_r \in CR$ — предварительное условие, определяющее, может ли роль r быть назначена учётной записи пользователя u.

2.2. Атрибуты учётной записи пользователя

Дадим определение семантически осмысленной ролевой модели, являющейся расширением классической модели ролевого управления доступом. Введём следующие дополнительные обозначения:

- A — множество атрибутов учётных записей пользователей. Элементами A являются атрибуты, характеризующие положение сотрудника, соответствующего учётной записи пользователя АИС, в организационно-штатной структуре предприятия, его должность и другие признаки, позволяющие определить его служебные обязанности. Например, $A = \{a_1, a_2\} = \{$«Должность», «Отдел»$\}$.
- V — множество допустимых значений атрибутов. Для каждого элемента множества A множество V содержит вектор значений, которые может принимать данный атрибут: $V = \{(v_{ij})\}$, где i — номер атрибута из A; $j = 1, \ldots, N_i$; N_i — число возможных значений атрибута $a_i \in A$.
- $values : A \to 2^V$ — функция, задающая для каждого атрибута множество его допустимых значений. Например, $V = \{values(a_1), values(a_2)\} = \{(v_1), (v_2)\} = \{(v_{11}, v_{12}, v_{13}), (v_{21}, v_{22}, v_{23})\}=\{$(«Специалист», «Старший специалист», «Ведущий специалист»), («Бухгалтерия», «Отдел Кадров», «Разработка»)$\}$.

Для обеспечения возможности администрирования множества атрибутов учётных записей пользователей используем функцию $can\_change\_attr : AR \to 2^U$, определяющую для каждой административной роли множество учётных записей, атрибуты которых могут быть изменены с использованием данной административной роли. Покажем теперь, каким образом определение элементов множества учётных записей пользователей U может быть расширено за счёт данных об атрибутах учётных записей пользователей.

Определение 8. Множество атрибут-пользователей $\hat{U}$ — это множество векторов $(u, ux_1, \ldots, ux_{na})$, где $ux_i \in values(a_i), u \in U$.

Элементы множества $\hat{U}$ описывают учётные записи пользователей АИС в соответствии с должностными обязанностями, выполняемыми связанными с ними сотрудниками. При этом будем считать, что для $\hat{u} = (u, ux_1, \ldots, ux_{na}) \in \hat{U}$ заданы функции, аналогичные функциям для элементов из множества U, следующим образом:

- $c_r(\hat{u}) = c_r(u)$;
- $UA(\hat{u}) = UA(u)$;
- $AUA(\hat{u}) = AUA(u)$.

Пример 2. Элемент множества $\hat{U}$, представляющий учётную запись пользователя АИС, соответствующую сотруднику отдела кадров, занимающему должность старшего специалиста, может быть представлен как $\hat{u} =$ (u, «Старший специалист», «Отдел кадров»), где u — элемент множества U, соответствующий данной учётной записи пользователя в АИС.

Для обеспечения возможности автоматического (без вмешательства администратора) изменения множества авторизованных ролей учетных записей пользователей АИС при изменении атрибутов, описывающих их должностные обязанности, необходимо ввести в модель управления доступом следующие элементы:

— учётную запись пользователя $system$, от имени которой выполняются все автоматические операции по изменению авторизованных ролей учётных записей пользователей АИС;
— учётную запись пользователя $attribute\_source$, от имени которой выполняются все автоматические операции по изменению атрибутов учётных записей пользователей АИС. Учётная запись пользователя $attribute\_source$ обладает административной ролью $r_{ua} \in AR$, дающей право изменения атрибутов учётной записи любого пользователя из $\hat{U}$, но не авторизована ни на одну из других ролей из AR или R: $UA(attribute\_source) = r_{ua}$;
— правила, описывающие взаимосвязь между атрибутами учётных записей пользователей и назначенными им ролями. Будем называть такие правила атрибут-условиями.

Определение 9. Определим множество атрибут-условий CA.

Пусть $t : \hat{U} \to \{\texttt{false}, \texttt{true}\}$ — функция, такая, что $t_i(\hat{u}) = \texttt{true}$ тогда и только тогда, когда для $\hat{u} \in \hat{U}$ выполняется условие $ux_i = t$.

Пусть $ca_r : \hat{U} \to \{\texttt{false}, \texttt{true}\}$ — функция, такая, что $ca_r(\hat{u}) = ca_r(t_1(\hat{u}), \ldots, t_{na}(\hat{u}))$, $\hat{u} \in \hat{U}$, $r \in R$, где $ca_r(y_1, \ldots, y_{na})$ — булева функция от na переменных, $na = 1, \ldots, A$. Тогда $CA = \{ca_{r1}, \ldots, ca_{rt}\}$ — множество функций, определяющих, при каких ограничениях на атрибуты той или иной учётной записи пользователя может быть назначена некоторая роль $r_i \in R$.

Пример 3. Пусть $A = \{a_1, a_2\} = \{$«Должность», «Отдел»$\}$, $V = \{values(a_1), values(a_2)\} = \{$(«Специалист», «Старший специалист», «Ведущий специалист»), («Бухгалтерия», «Отдел Кадров», «Разработка»)$\}$. Тогда условие для роли r, назначаемой учётной записи пользователя $\hat{u}$, соответствующего сотруднику бухгалтерии с должностью «Ведущий специалист», имеет вид $ca_r(\hat{u}) = (t_1(\hat{u}), t_2(\hat{u})) =$(«Ведущий специалист»$(\hat{u})$, «Бухгалтерия»$(\hat{u})$).

Определение 10. Если роль r имеет связанное атрибут-условие $ca_r \in CA$ и назначение её учётной записи пользователя $\hat{u}$ по этому условию инициировано сессией от имени учётной записи пользователя $system$, то такое назначение называется автоматическим, а роль — атрибут-ролью.

Атрибут-роль $r \in R_a$ может быть автоматически назначена учётной записи пользователя $(u, ux_1 \ldots ux_{na}) = \hat{u} \in \hat{U}$ тогда и только тогда, когда $ca_r(\hat{u}) = \texttt{true}$ и выполняются предварительные условия из CR (т. е. $c_r(\hat{u}) = \texttt{true}$).

Определение 11. Роли, для которых не существует ни одного атрибут-условия $ca_r \in CA$, будем называть классическими, а способ их назначения учётным записям пользователей — классическим.

По определению функции can_assign классическая роль $r \in R_k$ может быть назначена учётной записи пользователя $(u, ux_1, \ldots, ux_{na}) = \hat{u} \in \hat{U}$ сессией, функционирующей от имени администратора $\hat{u_2} \in \hat{U}$, тогда и только тогда, когда $r \in \in can\_assign\,(CR, AUA(u_2))$.

Предположение 3. Будем считать, что если для некоторой роли $r \in R$ существует атрибут-условие $ca_r \in CA$, то она всегда назначается автоматически и не может быть назначена классическим способом. В этом случае множество всех ролей R может быть представлено в виде $R = R_k \cup R_a$, $R_k \cap R_a = \varnothing$, где R_k — классические роли и R_a — множество атрибут-ролей.

Предположение 4. Будем считать, что учётная запись *system* авторизована на все административные роли $AR_a \in AR$, такие, что $can\_revoke^{-1}(AR_a) = R_a$ и $can\_assign^{-1}(AR_a) = R_a$.

Утверждение 1. Пусть атрибут-роли $r_1, r_2 \in R_a$ таковы, что $r_1 > r_2$ и атрибут-условие для роли r_2 имеет вид $ca_{r_2}(\hat{u}) = ca_{r_2}(t_1(\hat{u}), \ldots, t_k(\hat{u}))$. Тогда атрибут-условие для роли r_1 имеет вид $ca_{r_1}(\hat{u}) = ca_{r_1}(t_{k+1}(\hat{u}), \ldots, ca_{r_2}(\hat{u}), \ldots, t_{k+p}(\hat{u}))$, $t_i \in values(a_i)$, т. е. для возможности авторизации на некоторую роль учётная запись пользователя должна удовлетворять и всем атрибут-условиям для дочерних ролей.

Доказательство. Пусть $r_1 \in UA(\hat{u})$, $\hat{u} \in \hat{U}$, $r_1 \in R_a$. Отсюда следует, что $ca_{r_1}(\hat{u}) = \texttt{true}$. По определению иерархии, $r_2 \in UA(\hat{u})$ и $ca_{r_2}(\hat{u}) = \texttt{true}$. Таким образом, из $ca_{r_1}(\hat{u}) = \texttt{true}$ всегда следует $ca_{r_2}(\hat{u}) = \texttt{true}$. Следовательно, $ca_{r_2}(\hat{u}) = \texttt{true}$ является необходимым условием для $ca_{r_1}(\hat{u}) = \texttt{true}$, и $ca_{r_1}(\hat{u}) = ca_{r_1}(t_1(\hat{u}), \ldots, ca_{r_2}(\hat{u}), \ldots, t_k(\hat{u}))$. ■

Определение 12. Функция ограничения действий над учётной записью пользователя — это функция $H : \hat{U} \times \Gamma \times \Psi \to 2^{\Psi}$, значение которой для каждого действия $\sigma_i \in \Psi$, выполненного над учётной записью пользователя $\hat{u} \in \hat{U}$ и приведшего систему в текущее состояние $\gamma \in \Gamma$, равно множеству действий, которые могут быть выполнены над заданной учётной записью пользователя для перехода в последующее состояние. Множество Ψ всех возможных действий над учётной записью пользователя описано в таблице.

Пример 4. Пусть $H(\hat{u}, \gamma_t, \sigma_1) = \{\sigma_2, \sigma_3, \ldots, \sigma_q\}$. Это означает, что после выполнения над учётной записью пользователя $\hat{u}$ действия σ_1 над данной учётной записью в системе могут быть выполнены только действия из подмножества $\{\sigma_2, \sigma_3, \ldots, \sigma_q\} \subseteq \Psi$.

Определение 13. Траектория переходов $Tr : \hat{U} \times \Gamma \times \Gamma \times H \to 2^{\Psi}$ — это функция, задающая конечную последовательность действий $\gamma_i \in \Psi$, которая должна быть выполнена над учётной записью пользователя $\hat{u} \in \hat{U}$ в системе, находящейся в состоянии $\gamma_0 \in \Gamma$, для перехода системы в некоторое новое состояние $\gamma_1 \in \Gamma$ при условии, что множество возможных действия ограничено значением функции $H(\hat{u}, \gamma_0, \Psi)$.

Траектория используется пользователем *system* для определения последовательности выполнения автоматических действий по отзыву и назначению ролей.

Предположение 5. Администраторы системы могут выполнять действия над учётными записями пользователей в произвольном порядке при условии, что одновременно над этими учётными записями не выполняет никаких действий сессия от имени *system* (т. е. в текущий момент времени траектория действий для данной учетной записи пользователя пуста).

Пусть $Tr(\hat{u}, \gamma_0, \gamma, H(\hat{u}, \gamma_0, \sigma_0)) = \{\sigma_1, \ldots, \sigma_n\}$. Введём обозначения: $tr[0] = \sigma_1$ — первое действие, выполненное в системе над учётной записью $\hat{u}$ из состояния γ_0; $tr[1] = \sigma_2$ — второе действие и т. д. То есть если $\gamma = \sigma_n(\ldots\sigma_2(\sigma_1(\sigma_0(\gamma_0))))$, то $Tr(\hat{u}, \gamma_0, \gamma, H(\hat{u}, \gamma_0, \sigma_0)) = (tr[0], \ldots, tr[n]) = (\sigma_1, \ldots, \sigma_n)$, где $\sigma_i \in H(\hat{u}, \gamma_i, tr[i-1])$. Ограничения на разрешённые для каждого состояния системы действия и порядок их выполнения необходимы, так как операторы не являются коммутативными. Результирующее состояние системы зависит от порядка, в котором были выполнены действия.

Правила преобразования состояний семантической ролевой модели

Правило	Исходное состояние $\gamma_1 = (S, \hat{U}_1, UA_1, user_1, roles_1, H, TR_1)$	Результирующее состояние $\gamma_2 = \left(S, \hat{U}_2, UA_2, user_2, roles_2, H, TR_2\right)$
σ_1=$take\_role(x, r)$ — активация сессией одной из авторизированных ролей	$\sigma_1 \in H(user(x), \gamma_1, \sigma)$, $x_1 \in S$, $r \in R$, $r \in UA(user(x))$	$H(user(x), \gamma_2, \sigma_1) = H(user(x), \gamma_1, \sigma)$, $TR_2 = TR_1$, $S_2 = S_1$, $PA_2 = PA_1$, $user_2 = user_1$, $UA_2 = UA_1$, $roles_2(x) = roles_1(x) \cup \{r\}$, для $x_2 \in S \setminus \{x_1\}$ выполняется $roles_2(x_2) = roles_1(x_2)$
σ_2=$remove\_role(x, r)$ — удаление из сессии одной из ранее активированных ролей	$s_2 \in H(user(x), \gamma_1, \sigma)$, $x_1 \in S$, $r \in R$, $r \in roles(x)$	$TR_2 = TR_1$, $S_2 = S_1$, $PA_2 = PA_1$, $H(user(x), \gamma_2, \sigma_2) = H(user(x), \gamma_1, \sigma)$, $user_2 = user_1$, $UA_2 = UA_1$, $roles_2(x_1) = roles_1(x_1) \setminus \{r\}$, для $x_2 \in S \setminus \{x_1\}$ выполняется $roles_2(x_2) = roles_1(x_2)$
$\sigma_3 = assign\_role(u_1, u_2, r)$ — назначение роли классическим способом	$\sigma_3 \in H(u_1, \gamma_1, \sigma)$, $u_1, u_2 \in \hat{U}$, $r \in R_k$, $r \in can\_assign(CR, AUA(u_2))$	$H(u_1, \gamma_2, \sigma_3) = H(u_1, \gamma_1, \sigma)$, $S_2 = S_1$, $PA_2 = PA_1$, $user_2 = user_1$, $roles_2(x_1) = roles_1(x_1) \setminus \{r\}$, для $u \in \hat{U} \setminus \{u_1\}$ выполняется $UA_2(u) = UA_1(u)$, $UA_2(u_1) = UA_1(u_1) \cup \{r\}$, $roles_2 = roles_1$
σ_4=$revoke\_role(u_1, u_2, r)$ — отзыв роли классическим способом	$\sigma_4 \in H(u_1, \gamma_1, \sigma)$, $x \in S$, $u_1, u_2 \in \hat{U}$, $r \in R_k$, $r \in can\_revoke(AUA(u_2))$; для всех x, таких, что $user(x) = u_1$, выполняется $r \neq roles(x)$	$TR_2 = TR_1$; если для r существуют родительские роли, то $H(u_1, \gamma_2, \sigma_4) = \{\sigma_4\}$ (нельзя выполнять никаких действий, пока родительские роли не будут отозваны); иначе $H(u_1, \gamma_2, \sigma_4) = H(u_1, \gamma_1, \sigma)$; $S_2 = S_1$, $PA_2 = PA_1$, $user_2 = user_1$, для $\hat{u} \in \hat{U} \setminus \{u_1\}$ выполняется $UA_2(u) = UA_1(u)$, $UA_2(u_1) = UA_1(u_1) \setminus \{r\}$, $roles_2(x) = roles_1(x) \setminus \{r\}$
$\sigma_5 = change\_user\_attr(u, i, v)$ — изменение значения i-го атрибута учётной записи пользователя на новое значение v	$tr_1[0] = \sigma_5$, $\sigma_5 \in H(u, \gamma_1, \sigma)$, $x \in S$, $u = (u, ux_1, \ldots, ux_i, \ldots, x_n) \in \hat{U}$, $x_i \neq v$	$tr_2[0] = \sigma_8$, $H(u, \gamma_2, \sigma_5) = \{\sigma_8\}$ (после изменения атрибутов нельзя выполнять никаких действий, кроме перерасчёта ролей), $u = (u, ux_1, \ldots, v, \ldots, ux_n)$, для $\hat{u} \in \hat{U} \setminus \{u\}$ выполняется $\hat{U}_2 = \hat{U}_1$, $S_2 = S_1$, $PA_2 = PA_1$, $user_2 = user_1$, $UA_2 = UA_1$, $roles_2 = roles_1$
$\sigma_6 = auto\_assign\_role(u_1, r)$ — автоматическое назначение роли	$tr_1[0] = \sigma_6$, $x \in S$, $u_1 \in \hat{U}$, $r \in R_a$, $ca_r(u_1) =$ `true`	$H(u_1, \gamma_2, \sigma_6) = \{\sigma_6, \sigma_7\}$, $\hat{U}_2 = \hat{U}_1$, $S_2 = S_1$, $PA_2 = PA_1$, $user_2 = user_1$, для $\hat{u} \in \hat{U} \setminus \{u_1\}$ выполняется $UA_2(\hat{u}) = UA_1(\hat{u})$, $UA_2(u_1) = UA_1(u_1) \cup \{r\}$, $roles_2 = roles_1$
$\sigma_7 = auto\_revoke\_role(u_1, r)$ — автоматический отзыв роли	$tr_1[0] = \sigma_7$, $x \in S$, $u_1 \in \hat{U}$, $r \in UA(u_1)$, $r \in R_a$, $ca_r(u_1) =$ `false`	$H(u_1, \gamma_2, \sigma_7) = \{\sigma_6, \sigma_7\}$, $\hat{U}_2 = \hat{U}_1$, $S_2 = S_1$, $PA_2 = PA_1$, $user_2 = user_1$, для $u_2 \in \hat{U} \setminus \{u_1\}$ выполняется $UA_2(u_2) = UA_1(u_2)$, $UA_2(u_1) = UA_1(u_1) \setminus \{r\}$, $roles_2 = roles_1$
$\sigma_8 = auto\_recalculate(u)$ — построение траектории автоматического отзыва и назначения ролей	$tr_1[0] = \sigma_8$, $u \in \hat{U}$	$H(u, \gamma_2, \sigma_8) = \{\sigma_6, \sigma_7\}$, $TR_2 = TR_1 \cap \{\sigma_{71}, \sigma_{72}, \ldots, \sigma_{7k}, \sigma_{61}, \ldots, \sigma_{6m}\}$, где σ_{7i} — действие по отзыву роли $r_i \in R_a$, σ_{6j} — действие по назначению роли $r_j \in R_a$, $S_2 = S_1$, $PA_2 = PA_1$, $user_2 = user_1$, $UA_2 = UA_1$

Пример 5. Последовательный отзыв всех родительских ролей при отзыве дочерней роли у учётной записи пользователя $\hat{u} \in \hat{U}$ может быть выражен следующей траекторией: $Tr(\hat{u}, \gamma_t, \gamma_{t+1}, H(\hat{u}, \gamma_t, \sigma_{r_1})) = \left(\sigma_{r_2}, \ldots, \sigma_{r_q}\right)$, где σ_{r_1} — действие по отзыву роли r_1 и $r_2, \ldots, r_q$ — родительские роли для r_1. Таким образом, ни одно другое действие над данной учётной записью пользователя в системе не может быть выполнено, пока у неё не будут отозваны все роли, согласно иерархии.

Определение 14. Семантически осмысленная модель ролевого управления доступом задается множествами $\hat{U}$, R, P, $\hat{RH}$, CA, CR, V, A, AR, AP и функциями PA,

UA, AUA, H, Tr, $roles$ и $user$. При этом множества CA, A, V, $\hat{U}$ составляют семантический контекст ролевой модели.

Семантический контекст ролевой модели отражает связь между множествами учётных записей, прав доступа и ролей, организационно-штатной структурой и спецификой деятельности предприятия.

Предположение 6. В рамках данной работы считается, что административные роли AR не имеют семантической составляющей и назначаются всегда классическим способом.

Расширив определение 5 за счет семантического контекста и учёта траектории переходов, введём определение семантической системы ролевого управления доступом.

Определение 15. Семантическая система ролевого управления доступом — это конечный автомат $\Sigma = (\Gamma, Q, a, \Psi)$, состояния которого задаются кортежем $\gamma = \left(S, \hat{U}, UA, user, roles, H, Tr\right) \in \Gamma$, а правила перехода между состояниями определяются предварительными условиями CR, атрибут-условиями CA, функцией ограничения действий H и траекторией Tr.

Предположение 7. Предполагается, что множества $P, R, \hat{RH}, A, V, AR$, а также условия CR и CA не изменяются в процессе функционирования системы.

3. Анализ безопасности семантической системы ролевого управления доступом

3.1. Изменение атрибутов учётной записи и расчёт траектории перехода системы между состояниями

Сформулируем и обоснуем правила, на основании которых определяется порядок следования элементов в траектории Tr.

Утверждение 2. Пусть в результате успешного выполнения действия σ_5 система $\Sigma = \langle \Gamma, Q, a, \Psi \rangle$ перешла в состояние γ_1, где атрибуты учетной записи пользователя $\hat{u} \in \hat{U}$ изменились таким образом, что необходимо отозвать у него атрибут-роли $r_{11}, r_{12}, \ldots, r_{1k} \in R_a$ и назначить ему атрибут-роли $r_{21}, r_{22}, \ldots, r_{2m} \in R_a$, переведя систему в состояние γ_2. Пусть $Tr(\hat{u}, \gamma_1, \gamma_2, H)$ — траектория перехода из состояния γ_1 в состояние γ_2. Переход из состояния γ_1 в состояние γ_2 по траектории Tr возможен только в том случае, если построенная траектория удовлетворяет следующим свойствам:

1) действия σ_7 по отзыву ролей должны следовать в траектории перед действиями по назначению ролей σ_6;

2) если r_i и r_j — такие роли, что $r_i > r_j$, то $\sigma_7(r_j)$ должно следовать в траектории раньше, чем $\sigma_7(r_i)$, и $\sigma_6(r_i)$ должно выполняться раньше, чем $\sigma_6(r_j)$.

Доказательство. Покажем достаточность выполнения условий 1 и 2 для возможности перехода по заданной траектории TR. Невозможность выполнения одного или более действий в траектории перехода может произойти в одном из двух случаев.

Случай 1. Пусть последовательность действий в траектории TR такова, что некоторой учётной записи $\hat{u}$ необходимо автоматически назначить роль r_a, но данное назначение противоречит предварительным условиям CR, так как учётная запись пользователя $\hat{u}$ уже авторизована на роли из множества $\{r_{11}, r_{12}, \ldots, r_{1k}\} \subseteq R_a$, исключающие назначение роли r_a. Данная ситуация противоречит условию 1 утверждения, так как действия по отзыву ролей $r_{11}, r_{12}, \ldots, r_{1k}$ всегда выполняются до выполнения действий по назначению новых ролей.

С л у ч а й 2. Пусть последовательность действий в траектории TR такова, что некоторой учётной записи $\hat{u}$ необходимо автоматически назначить роль r_a, но данное назначение противоречит предварительным условиям CR, так как учётная запись пользователя $\hat{u}$ не авторизована на роли из множества $\{r_{21}, r_{22}, r_{2y-1}, \ldots, r_{2y+1}, \ldots, r_{1m}\} \subseteq R_a$, являющиеся предусловием для назначения роли r_a. По определению иерархии, если роль r_2 является предусловием для роли r_1, то $r_1 > r_2$. Следовательно, данная ситуация противоречит условию 2 утверждения, так как дочерние роли назначаются учётной записи пользователя раньше, чем родительские роли. ■

Замечание. Условия утверждения 2 являются необходимыми, но не достаточными для перехода в состояние γ_2. Невозможность выполнения действий, заданных траекторией, может произойти также в результате некорректного определения предварительных условий, если две или более ролей из множества $\{r_{21}, r_{22}, \ldots, r_{2m}\} \subseteq R_a$ являются взаимоисключающими. Если предварительные условия CR не противоречат атрибут-условиям CA и ни одна из ролей $r_{21}, r_{22}, \ldots, r_{2m} \in R_a$ не исключает других ролей из того же множества, то условия утверждения являются необходимыми и достаточными для перехода в состояние γ_2 по вычисленной траектории Tr.

Утверждение 3. Пусть семантически осмысленная система ролевого управления доступами $\Sigma = \langle \Gamma, Q, a, \Psi \rangle$ содержит только атрибут-роли (т. е. $R = R_a$, $AR_a = AR$) и учётную запись *system*, от имени которой инициируется сессия, выполняющая автоматический перерасчёт назначенных учётной записи пользователя ролей при изменении её атрибутов согласно траектории, удовлетворяющей условиям утверждения 2. Пусть атрибуты учётной записи пользователя полностью описывают его должностные обязанности и все изменения атрибутов передаются в каталог системы ролевого управления доступом. Тогда каждая учётная запись пользователя $\hat{u} \in \hat{U}$ всегда имеет набор ролей, соответствующих его должностным обязанностям.

Доказательство. Пусть должностные обязанности учётной записи пользователя, соответствующей некоторому сотруднику предприятия, характеризуются набором атрибутов $A = a_1, a_2, \ldots, a_N$. Пусть $r_1, r_2 \in R_a$ — атрибут-роли, с которыми связаны различные атрибут-условия ca_{r1} и ca_{r2} соответственно. Пусть первоначально должностные обязанности учётной записи пользователя $\hat{u}$ были таковы, что $ca_{r1}(\hat{u}) = \texttt{true}$ и $ca_{r2}(\hat{u}) = \texttt{false}$. В таком случае учётной записи пользователя назначена только роль r_1 и соответствующие ей права доступа. Пусть после смены должности атрибуты учётной записи пользователя изменяются таким образом, что $ca_{r1}(\hat{u}) = \texttt{false}$ и $ca_{r2}(\hat{u}) = \texttt{true}$. В этом случае, согласно правилам, приведённым в таблице, и определению атрибут-роли, в системе инициируется автоматический отзыв роли r_1 и автоматическое назначение роли r_2. Поскольку, по определению модели $RBAC$, права доступа назначаются учётной записи пользователя только посредством ролей, она будет обладать только правами доступа, соответствующими новой должности связанного с ней сотрудника. ■

3.2. А н а л и з д е й с т в и й п о п е р е х о д у м е ж д у с о с т о я н и я м и

Проведем анализ действия *change_user_attr*, выполняемого сессией от имени учётной записи *attribute_source*. Выполнение действия $change\_user\_attr(u, i, v)$ для заданного состояния системы является успешным только в том случае, если выполняются следующие условия:

— значение некоторого атрибута учётной записи пользователя, соответствующего сотруднику в системе учёта кадров предприятия, не совпадает со значением соот-

ветствующего атрибута учётной записи связанного пользователя $\hat{u} \in \hat{U}$ в каталоге учётных записей системы ролевого управления доступом;

— в данный момент времени над учётной записью пользователя не выполняется никаких действий ($Tr(\hat{u}, \gamma_1, \gamma_2, H) = \varnothing$). В случае успешного выполнения действия над заданным состоянием системы $\gamma_1 = (\hat{U}_1, UA_1, user_1, roles_1)$ новое состояние $\gamma_1 = \left(\hat{U}_2, UA_2, user_2, roles_2\right)$ отличается от исходного множеством значений $\hat{U}$.

Результатом успешного выполнения действия по изменению атрибутов учётной записи $\hat{u_1} = (u, ux_1, \ldots, ux_i, \ldots, ux_n) \in \hat{U}$ является состояние, в котором $\hat{U_2} = \left(\hat{U_1} \setminus \{\hat{u_1}\}\right) \cup$ $\cup\{\hat{u_2}\}$, где $\hat{u_2} = (u, ux_1, \ldots, v, \ldots, ux_n)$, $v \neq ux_i$. В случае неуспешного выполнения действия состояние системы не изменится.

Проведем анализ действий $auto\_assign\_role$ и $auto\_revoke\_role$, выполняемых сессией-инициатором $system$. Выполнение действия $auto\_assign\_role\,(\hat{u}, r)$ для заданного состояния системы является успешным только в том случае, если выполняются следующие условия:

— выполняются атрибут-условия назначения роли $ca_r(\hat{u}) = \texttt{true}$;

— выполняются предварительные условия CR. Поскольку субъект $system$ авторизован на все возможные административные роли для управления назначениями атрибут-ролей, то значение функции $can\_assign\,(CR, AUA\,(system))$ зависит только от аргумента CR;

— σ_6 является текущим значением траектории ($tr[0] = \sigma_6$).

В случае успешного выполнения действия над заданным состоянием системы $\gamma_1 = (\hat{U_1}$, $UA_1, user_1, roles_1)$ новое состояние $\gamma_2 = \left(\hat{U_2}, UA_2, user_2, roles_2\right)$ отличается от исходного значением функции UA. Результатом успешного выполнения действия является состояние, для которого $UA_2(\hat{u_1}) = UA_1(\hat{u_1}) \cup \{r\}$. Выполненное действие удаляется из траектории. В случае неуспешного выполнения действия состояние системы не изменится.

Рассмотрим действие автоматического отзыва роли $auto\_revoke\_role\,(\hat{u}, r)$, где сессия от имени учётной записи пользователя $system$ отзывает у учётной записи пользователя $\hat{u}$ роль $r \in R_a$. Выполнение данного действия для заданного состояния системы является успешным только в том случае, если:

— учётная запись пользователя $\hat{u}$ авторизована на роль r, т. е. $r \in UA\,(u)$;

— значения атрибутов учётной записи пользователя не соответствуют атрибут-условиям назначения для роли r, т. е. $ca_r\,(\hat{u}) = \texttt{false}$;

— σ_7 является текущим значением траектории ($tr[0] = \sigma_7$).

В случае успешного выполнения действия над заданным состоянием системы $\gamma_1 = (\hat{U_1}$, $UA_1, user_1, roles_1)$ новое состояние $\gamma_2 = \left(\hat{U_2}, UA_2, user_2, roles_2\right)$ отличается от исходного значением функции UA. Результатом успешного выполнения действия является состояние, для которого $UA_2(\hat{u_1}) = UA_1(\hat{u_1}) \setminus (\{r\} \cup up\,(r))$, где $up\,(r)$ — множество ролей, больших r. Выполненное действие удаляется из траектории. В случае неуспешного выполнения действия состояние системы не изменится.

Рассмотрим действие $auto\_recalculate\,(\hat{u})$. Условием выполнения данного действия является переход системы в состояние, где σ_8 является единственным возможным действием: $H(\hat{u}, \gamma, \sigma) = \{\sigma_8\}$, $TR[0] = \sigma_8$. Согласно таблице, такая ситуация возможна в результате успешного выполнения действия σ_5. Результатом выполнения действия

является траектория Tr, состоящая из действий σ_6 и σ_7 в соответствии с количеством ролей, которые необходимо назначить или отозвать.

Проведем анализ действий $\sigma_3 = assign\_role$ и $\sigma_4 = revoke\_role$, выполняемых сессией от имени учетной записи администратора системы $admin \in INIT\{\sigma_3, \sigma_4\}$. Выполнение действия $assign\_role(admin, \hat{u}, r)$ для заданного состояния системы γ_1 является успешным только в том случае, если выполняются следующие условия:

— $r \in R_k$ — классическая роль;
— выполняются предварительные условия $CR : c_r(\hat{u}) = \texttt{true}$;
— $r \in can\_assign(CR, AUA(admin))$;
— $\sigma_3 \in H(\hat{u}, \gamma, \sigma)$ и над учётной записью пользователя в данный момент не выполняется никаких других действий — $Tr(\hat{u}, \gamma_1, \gamma_2, H) = \varnothing$.

В случае успешного выполнения действия над заданным состоянием системы $\gamma_1 = (\hat{U_1}, UA_1, user_1, roles_1)$ новое состояние $\gamma_2 = \left(\hat{U_2}, UA_2, user_2, roles_2\right)$ отличается от исходного значением функции UA. Результатом успешного выполнения действия является состояние, для которого $UA_2(\hat{u}) = UA_1(\hat{u}) \cup \{r\}$. В случае неуспешного выполнения действия состояние системы не изменится.

Аналогичные условия выполняются для действия классического отзыва роли $revoke\_role(admin, \hat{u}, r)$, где сессия от имени учётной записи администратора $admin$ отзывает у учётной записи пользователя $\hat{u}$ роль $r \in R_k$. Выполнение данного действия для заданного состояния системы является успешным только в том случае, если:

— $r \in R_k$ — классическая роль;
— $\{r\} \cup up(r) \subseteq can\_revoke(AUA(admin))$, где $up(r)$ — множество ролей, старших r;
— $\sigma_4 \in H(\hat{u}, \gamma, \sigma)$ и над учётной записью пользователя в данный момент не выполняется никаких других действий — $Tr(\hat{u}, \gamma_1, \gamma_2, H) = \varnothing$.

В случае успешного выполнения действия над заданным состоянием системы $\gamma_1 = (\hat{U_1}, UA_1, user_1, roles_1)$ новое состояние $\gamma_2 = \left(\hat{U_2}, UA_2, user_2, roles_2\right)$ отличается от исходного значением функции UA. Результатом успешного выполнения действия является состояние, для которого $UA_2(\hat{u}) = UA_1(\hat{u}) \setminus (\{r\} \cup up(r))$. Выполненное действие удаляется из траектории. В случае неуспешного выполнения действия состояние системы не изменится.

3.3. Критерий безопасности функционирования системы

Сформулируем критерий безопасного функционирования системы семантического ролевого управления доступом и покажем, что в случае, когда управление назначениями атрибут-ролей осуществляется автоматически от лица единственной доверенной учётной записи $system \in INIT(\sigma)$, а классические роли управляются исключительно сессиями, функционирующими от имени учётных записей пользователей из подмножества администраторов $U_a \subseteq INIT(\sigma)$, $U_a \cap \{system\} = \varnothing$, то система является безопасной с точки зрения данного критерия.

Определение 16. Состояние $\gamma \in \Gamma$ системы ролевого управления доступом $\Sigma = \langle \Gamma, Q, a, \Psi \rangle$ называется безопасным, если каждой учётной записи пользователя $\hat{u} \in \hat{U}$ назначены только роли, не противоречащие условиями CR и CA.

Определение 17. Система $\Sigma = \langle \Gamma, Q, a, \Psi \rangle$ называется безопасной, если любое состояние γ_g, достижимое из начального безопасного состояния γ посредством действий, инициированных сессией от имени любой из учётных записей пользователей $\hat{u} \in INIT(\Psi)$, является безопасным.

Теорема 1. Пусть семантически осмысленная система ролевого управления доступом $\Sigma = \langle \Gamma, Q, a, \Psi \rangle$ обладает следующими свойствами:

1) все действия по назначению и отзыву атрибут-ролей $R_a \in R$ инициируются автоматически в порядке, определённом траекторией Tr;
2) множество значений функции $INIT(\sigma_5)$ включает в себя ровно одну учётную запись пользователя $attribute\_source$, т. е. $INIT(\sigma_5) = \{attribute\_source\}$;
3) множество значений функции $INIT(\sigma_6, \sigma_7)$ включает в себя ровно одну учётную запись пользователя $system$, т. е. $INIT(\sigma_6, \sigma_7) = \{system\}$;
4) $INIT(\sigma_3, \sigma_4) = U_a$, при этом $INIT(\sigma_3, \sigma_4) \cap INIT(\sigma_6, \sigma_7) = \varnothing$.

Тогда, если начальное состояние ролевой модели $\gamma_0 = \left(\hat{U}_0, UA_0, user_0, roles_0\right)$ является безопасным, то система Σ является безопасной.

Доказательство. Пусть начальное состояние ролевой модели γ_0 является безопасным. Поскольку при выполнении действий σ_3 и σ_6 по назначению роли учётной записи пользователя осуществляется проверка того, что новое назначение не противоречит предварительным условиям CR, нарушение свойства безопасности системы при переходе из γ_0 в новое состояние γ_g может произойти только в следующих случаях.

С л у ч а й 1. Невозможность автоматического назначения роли в результате назначения взаимоисключающей роли администратором системы: $\sigma_6(\sigma_3(\sigma_5(\gamma_0))) = \gamma_g$. Пусть атрибуты учётной записи пользователя в системе были изменены в результате действия σ_5 и сессией администратора $u_a \in U_a$ инициируется выполнение действия σ_3 по назначению классической роли r_k до того, как сессия $system$ инициирует действие σ_6 по назначению атрибут-роли r_a. После инициализации σ_6 может возникнуть ситуация, когда атрибут-роль r_a должна быть назначена учётной записи пользователя по условиям CA, но не может быть назначена по условиям CR, так как роль r_k является взаимоисключающей с r_a. Данная ситуация противоречит условию 1 теоремы, так как любые другие действия над учётной записью пользователя в системе запрещены до завершения цепочки действий σ_7 и σ_6, выполняемых согласно траектории, рассчитанной при выполнении σ_8. Действие σ_8 является единственным разрешённым действием после успешного выполнения σ_5, и до завершения σ_8 любые другие действия над заданной учётной записью запрещены.

С л у ч а й 2. Невозможность автоматического назначения роли в результате отзыва администратором системы роли, входящей в предусловие: $\sigma_7(\sigma_4(\sigma_5(\gamma_0))) = \gamma_g$. Пусть атрибуты учётной записи пользователя в системе были изменены в результате действия σ_5 и сессия администратора $u_a \in U_a$ инициирует выполнение действия σ_4 по отзыву классической роли r_k до того, как сессия $system$ инициирует действие σ_6 по назначению атрибут-роли r_a. После инициализации σ_4 может возникнуть ситуация, когда роль r_k должна быть назначена учётной записи пользователя по условиям CA, но не может быть назначена ей по условиям CR, так как авторизация на роль r_k является обязательным предусловием назначения r_a. Аналогично случаю 1, данная ситуация противоречит условию 1 теоремы.

С л у ч а й 3. После завершения перерасчёта ролей сессия администратора отзывает роль, дочернюю для автоматически назначенной роли $\sigma_4(\sigma_6(\sigma_5(\gamma_0))) = \gamma_g$. Пусть атрибуты учётной записи пользователя в системе изменены в результате действия σ_5 и атрибут-роль r_a назначена автоматически действием σ_6, но на следующем шаге сессия администратора $u_a \in U_a$ инициирует выполнение действия σ_4 по отзыву классической роли r_k, которая является дочерней (необходимым предусловием) для роли r_a. После инициализации σ_4 может возникнуть ситуация, когда атрибут-роль ra должна быть на-

значена учётной записи пользователя по условиям CA, но не может быть назначена по условиям CR, так как роль r_k является предварительным условием для r_a. Данная ситуация противоречит условиям 3 и 4 теоремы. Если классическая роль r_k обязательно назначается всем учётным записям пользователей, обладающим ролью r_a, то это означает, что $r_k > r_a$. Следовательно, $r_k \in can\_assign\,(CR, AUA\,(u_a))$ и, по определению функции $INIT(\sigma)$, $u_a \in INIT\,(\sigma_6, \sigma_7)$. Но тогда либо $INIT\,(\sigma_3, \sigma_4) \cap INIT\,(\sigma_6, \sigma_7) \neq \varnothing$, либо роль r_k имеет атрибут-условие назначения и не является классической. Противоречие.

С л у ч а й 4. Наличие двух и более администраторов атрибут-ролей.

Пусть $\sigma_7\,(\sigma_6\,(\sigma_5\,(\gamma_0))) = \gamma_g$ и права по администрированию атрибут-ролей назначены одновременно двум и более учётным записям пользователей $system_1$ и $system_2$. Атрибуты учётной записи пользователя в системе были изменены в результате действия σ_5, и сессия $system_1 \in INIT\,(\sigma_6, \sigma_7)$ инициирует выполнение действия σ_6 по назначению атрибут-роли ra_1 до того, как сессия $system_2 \in INIT\,(\sigma_6, \sigma_7)$ инициирует действие σ_7 по отзыву атрибут-роли ra_2, являющейся взаимоисключающей для ra_1. После инициализации σ_6 возможна ситуация, когда роль ra_1 может быть назначена учётной записи пользователя по условиям CA, но её назначение блокируется условиями CR, так как роль ra_2 является взаимоисключающей с ra_1. Данный случай противоречит требованию 3 теоремы о единственности субъекта $system$. ■

Утверждение 4. Необходимым условием для выполнения свойства 4 в теореме 1 является то, что если некоторая роль $r_a \in R$ является атрибут-ролью, то все её старшие роли также являются атрибут-ролями.

Доказательство. От противного. Если атрибут-роль r_a обязательно назначается всем учётным записям пользователей, авторизованным на классическую роль $r_k \in R$, то это означает, что $r_k > r_a$. По предположению 2 в этом случае для некоторой административной роли $r \in AR$, такой, что $r_a \in can\_revoke\,(r)$, выполняется условие $r_k \in can\_revoke\,(r)$. Следовательно, либо $INIT\,(\sigma_3, \sigma_4) \cap INIT\,(\sigma_6, \sigma_7) \neq \varnothing$, либо роль r_k имеет атрибут-условие назначения и не является классической. Получили противоречие. ■

Замечание Свойство распространения атрибут-условий вверх по иерархии доказано в утверждении 1. Поскольку по предположению 7 на протяжении функционирования системы ролевая иерархия $\hat{RH}$ остаётся неизменной, то не может возникнуть ситуации, когда классическая роль включена в иерархию атрибут-ролей администратором системы. По определению множества $INIT(\sigma)$, свойство 3 в теореме эквивалентно требованию $AR = AR_a \cup AR_k$, $AR_a \cap AR_k = \varnothing$, где AR_a — административные роли для атрибут-ролей, AR_k — административные роли для классических ролей. Следовательно, достаточным условием выполнения свойства 3 в теореме 1 является введение ограничения статического взаимного исключения ролей из подмножеств AR_a и AR_k.

Заключение

Дополнение классической модели $RBAC$ за счет семантического контекста позволяет адаптировать её к условиям функционирования реальных АИС предприятий, где каждой учётной записи пользователя АИС соответствует сотрудник предприятия с определённым набором должностных обязанностей, а также автоматизировать операции назначения и отзыва ролей. Ряд элементов классической модели и правила преобразования состояний переопределены для возможности разделения операций по назначению ролей на автоматические и выполняющиеся сессиями, функционирующи-

ми от имени администраторов системы. Кроме того, в семантической модели вводится определение иерархии ролей, отличное от классического, не требующее обязательного включения прав доступа дочерних ролей в родительские, так как это свойство не всегда выполняется в реальных системах. Безопасность системы ролевого управления доступом, построенной на основе предложенной модели, подтверждается проведённым анализом переходов между состояниями.

ЛИТЕРАТУРА

1. `http://csrc.nist.gov/rbac/rbacSTD-ACM.pdf` — National Institute of Standards and Technology, Proposed Standard for Role-Based Access Control.
2. *Ferraiolo D., Kuhn D., and Chandramoul R.* Role Based Access Control. NewYork: ARTECH HOUSE, INC, 2003. 337 с.
3. *Девянин П. Н.* Модели безопасности компьютерных систем. Управление доступом и информационными потоками: учеб. пособие для вузов. М: Горячая линия — Телеком, 2011. 320 с.
4. *Sandhu R. S., Bhamidipati V., and Munawer Q.* The ARBAC97 model for role-based aministration of roles // ACM Trans. Information and Systems Security. No. 2(1). NewYork: ACM Publishing, 1999. P. 105–135.
5. *Нокс Д.* Создание эффективной системы безопасности для Oracle Database 10g. М: Лори, 2007. 576 с.
6. *Джоханссон Дж.* Обеспечение безопасности. Ресурсы Windows Server 2008. СПб.: Русская редакция, 2009. 544 с.

МАТЕМАТИЧЕСКИЕ ОСНОВЫ ИНФОРМАТИКИ И ПРОГРАММИРОВАНИЯ

DOI 10.17223/20710410/16/5

УДК 519.178

FPT-АЛГОРИТМЫ НА ГРАФАХ ОГРАНИЧЕННОЙ ДРЕВОВИДНОЙ ШИРИНЫ

В. В. Быкова

Институт математики Сибирского федерального университета, г. Красноярск

E-mail: bykvalen@mail.ru

Исследован специальный метод конструирования FPT-алгоритмов — метод динамического программирования на основе дерева декомпозиции. Выявлены проблемы, ограничивающие применение этого метода на практике. Предложено проблему памяти решать с помощью бинарного сепараторного дерева декомпозиции, снижающего теоретические и реальные размеры таблиц динамического программирования. Табличная техника описана на языке реляционной алгебры.

Ключевые слова: *алгоритмы на графах, дерево декомпозиции, динамическое программирование.*

Введение

Параметризированный подход к оценке сложности вычислений — естественный способ справиться с трудноразрешимостью задач, которые охарактеризованы как NP-трудные в соответствии с классической дихотомией **P** против **NP** [1]. Основная идея параметризированного подхода состоит в том, чтобы с помощью некоторого числового параметра учесть структуру исходных данных и выделить основной источник трудноразрешимости задачи. Параметризированный подход позволяет исследовать различные параметризации одной и той же задачи, каждая из которых может приводить или не приводить к FPT-алгоритмам. Такие алгоритмы представляют интерес для алгоритмической практики, так как при ограниченном значении параметра время их выполнения полиномиально зависит от длины входа алгоритма [2].

В настоящее время уже известны некоторые специальные методы проектирования FPT-алгоритмов. Наиболее известный из них — метод динамического программирования на основе дерева декомпозиции, автором которого считается Г. Бодлаендер [3]. Этот метод ориентирован на класс задач, решение которых ищется в конечной области. При этом поиск решения подразумевает нахождение одного из допустимых решений (в задачах разрешения и удовлетворения ограничений) или оптимального решения (в задачах комбинаторной оптимизации). Такие задачи в литературе часто называют задачами выбора [4]. Параметризация задачи в данном случае осуществляется по древовидной ширине входного графа. Несмотря на теоретическую привлекательность метода динамического программирования по дереву декомпозиции, практическое его применение ограничивается двумя проблемами, первая из них связана с высокими потребностями в памяти получаемых FPT-алгоритмов, а вторая — с вычислением древовидной ширины и построением дерева декомпозиции. В данной работе предложены

средства решения первой из указанных проблем. Методы решения второй проблемы рассмотрены в работе автора [5].

1. Дерево декомпозиции и древовидная ширина графа

Динамическое программирование по дереву декомпозиции — это сочетание декомпозиционного подхода к решению задачи выбора с декомпозиционным представлением входного графа, когда выделение подзадач и построение их решений осуществляется исходя из этого представления. Дерево декомпозиции графа — это специальная конструкция, которая описывает структуру графа «с точностью до клик и сепараторов» и определяется следующим образом. Пусть задан связный граф $G = (V, E)$, $n = |V| \geqslant 1$ и $m = |E| \geqslant 1$. Дерево декомпозиции графа G — пара (M, T), задающая разбиение множества вершин и множества рёбер этого графа, где $M = \{X_i : i \in I\}$ — семейство подмножеств множества V, называемых «мешками», а $T = (I, W)$ — дерево, узлам которого сопоставлены эти «мешки». Вершины дерева T принято называть узлами, чтобы избежать путаницы с вершинами графа G. Семейство $M = \{X_i : i \in I\}$ и множество W рёбер дерева $T = (I, W)$ такие, что справедливы следующие условия [3]:

1) объединение всех подмножеств, образующих «мешки», даёт множество V;
2) для всякого ребра графа G существует хотя бы один «мешок», содержащий обе вершины этого ребра;
3) для любой вершины v графа G множество узлов $\{i \in I : v \in X_i\}$, «мешки» которых содержат эту вершину, индуцирует связный подграф, являющийся поддеревом дерева T.

Ширина дерева декомпозиции (M, T) равна $\max\limits_{i \in I}\{|X_i| - 1\}$. Древовидная ширина (*treewidth*) графа G определяется как наименьшая ширина всех допустимых его деревьев декомпозиции и обозначается через $\mathrm{tw}(G)$. Дерево декомпозиции ширины $\mathrm{tw}(G)$ называется оптимальным, а без кратных и вложенных «мешков» — фундаментальным. В фундаментальном дереве декомпозиции всегда $\mathrm{O}(n)$ узлов и к нему можно перейти за время $\mathrm{O}(|I|)$. Для $\mathrm{tw}(G)$ верны естественные границы: $1 \leqslant \mathrm{tw}(G) \leqslant n-1$. Считается, что граф G обладает ограниченной древовидной шириной, если $\mathrm{tw}(G) \leqslant k$ и k есть целая положительная константа, не зависящая от n.

Заметим, что всякое дерево декомпозиции (M, T) графа $G = (V, E)$ есть не что иное, как дерево соединений ациклического гиперграфа $H = (V, M)$, рёбрами которого выступают «мешки» этого дерева [6]. При этом граф смежности вершин гиперграфа H является некоторой (не обязательно минимальной) триангуляцией графа G.

2. Описание метода

Пусть задана некоторая параметризированная задача П, которую надо решить для графа $G = (V, E)$. Полагаем, что в качестве параметра задачи взята древовидная ширина $\mathrm{tw}(G)$ и $\mathrm{tw}(G) \leqslant k$, k — целая положительная константа. Кроме того, считаем, что задача П сформулирована в оптимизационной конструктивной постановке, то есть задана некоторая числовая характеристика (целевая функция) графа $G = (V, E)$, которую надо оптимизировать и указать решение, отвечающее оптимальному значению этой характеристики.

Пусть для исходного графа $G = (V, E)$ известно корневое дерево декомпозиции (M, T), $M = \{X_i : i \in I\}$, $T = (I, W)$, ширины k и с узлом r в роли корня. Семейство подзадач в данном случае можно задать следующим образом. Определим для всякого узла $i \in I$ множество вершин:

$$Y_i = \{v \in X_j : j = i \text{ или } j \text{ — потомок для } i \text{ в } T\}. \quad (1)$$

Множество Y_i индуцирует в G подграф $G_i = G(Y_i)$, а в T — поддерево с корневым узлом i. Примечательно, что $Y_r = V$ и $G_r = G$. Тогда в качестве подзадачи Π_i можно рассматривать решение задачи Π применительно к G_i, $i \in I$. Понятно, что для каждой конкретной задачи Π специфичны характеристика решения и рекуррентные процедуры, связывающие решения подзадач. Между тем сценарий действия алгоритмов, основанных на динамическом программировании по дереву декомпозиции, общий.

Работа подобных алгоритмов включает в себя два этапа. На первом этапе для исходного графа $G = (V, E)$ строится корневое дерево декомпозиции (M, T) ширины k. На втором этапе решается задача Π с помощью (M, T): сначала обход всех узлов дерева T снизу вверх (от листьев к корню r) с целью вычисления необходимой информации и нахождения значений характеристик подзадач; затем обход всех узлов дерева T сверху вниз (от корня r к листьям) с целью конструирования оптимального решения задачи Π для исходного графа G. При этом вся нужная информация вычисляется и хранится в виде таблиц. Каждому узлу $i \in I$ дерева T соответствует таблица A_i, которая содержит информацию по задаче Π_i.

Процесс формирования таблиц и решений отвечает следующим необходимым требованиям. При любом $i \in I$ решение задачи Π_i находится исключительно из таблицы A_i. Для всякого листа $i \in I$ дерева T соответствующая таблица вычисляется непосредственно из $G(X_i)$. Для каждого внутреннего узла $i \in I$ таблица создается из информации о $G(X_i)$ и таблиц, которые отвечают прямым потомкам узла i в T. Дерево декомпозиции гарантирует выполнение указанных требований, поскольку справедливо

Утверждение 1 [3]. Пусть $i \in I$ — произвольный внутренний узел дерева T. Вершины графа $G_i = G(Y_i)$, которые смежны с вершинами, находящимися вне G_i, обязательно содержатся в «мешке» X_i (и таких вершин не более $k + 1$). Иными словами, графы $G(Y_i)$ и $G(V \backslash Y_i)$ «связаны» между собой в G только с помощью вершин из X_i.

В общем случае поиск точного решения задачи Π_i для каждого узла i дерева T осуществляется полным перебором в A_i, где представляются различные подмножества множества вершин как претенденты на оптимальное или допустимое решение. Очевидно, что время подъёма и спуска по дереву декомпозиции зависит от числа и размера создаваемых таблиц динамического программирования, а также от времени обработки каждой строки такой таблицы.

3. Достаточные условия FPT-разрешимости относительно древовидной ширины

Оказывается, что описанный выше метод динамического программирования на основе дерева декомпозиции приводит к FPT-алгоритму относительно древовидной ширины, если задача выбора может быть сформулирована в монадической логике второго порядка. Монадическая логика второго порядка (*Monadic Second Order Logic*, или MSOL) для графа $G = (V, E)$ — язык исчисления предикатов выражения его свойств. Предикатные формулы в MSOL, описывающие свойства графа G, состоят из следующих символов [7]:

— предметных переменных v_i и e_j, при этом каждой переменной v_i соответствует отдельная вершина, а переменной e_j — отдельное ребро графа G ($1 \leqslant i \leqslant n = |V|$, $1 \leqslant j \leqslant m = |E|$);

— предметных переменных V_i и E_j, где переменной V_i отвечает некоторое подмножество множества вершин V, а переменной E_j — некоторое подмножество множества рёбер графа G ($0 \leqslant i \leqslant 2^n$, $0 \leqslant j \leqslant 2^m$);
— предикатных переменных и логических связок $\&$, $\vee$, $\neg$, $\Rightarrow$, $\Leftrightarrow$;
— кванторов $\forall$ и $\exists$, запятых и круглых скобок.

В состав атомарных формул входят:

— предикат совпадения ($x = x'$). Принимает значение «истина», если вершины (рёбра) x и x' графа G совпадают;
— предикат принадлежности ($x \in X$), где предметная переменная x отвечает вершине (ребру) графа G, а переменная X — подмножеству вершин (рёбер) этого графа;
— предикат инцидентности $\mathrm{Incident}(v_i, e_j)$. Принимает значение «истина», если вершина v_i и ребро e_j инцидентны в G.

Все другие формулы MSOL формируются из атомарных формул MSOL с помощью логических связок, кванторов $\exists x$ и $\forall x$ (x — предметная переменная, соответствующая вершине или ребру графа G) и кванторов $\exists X$ и $\forall X$ (X — предметная переменная, отвечающая подмножеству вершин или ребер графа G). Заметим, что монадическая логика первого порядка (*Monadic First Order Logic*, или MFOL) отличается от MSOL лишь тем, что в ней не допускаются кванторы $\exists X$ и $\forall X$.

Всякое свойство графа G может быть задано некоторым предикатом P. Предикат P для графа G (обозначается через $P(G)$) принимает значение «истина», если рассматриваемое свойство выполняется для этого графа, и «ложь» в противном случае. Некоторые популярные свойства графа, выраженные в виде формул MSOL, представлены в таблице. Из всех указанных в этой таблице свойств только свойства связности и гамильтоновости графа не могут быть выражены в MFOL. Заметим, что длина всех формул, приведённых в таблице, не зависит от размера (числа вершин и рёбер) графа G. Такие формулы MSOL называются конечными.

Пусть граф G имеет ограниченную древовидную ширину, то есть $\mathrm{tw}(G) \leqslant k$, где k — некоторая целая положительная константа. Пусть проверяемое свойство графа G представлено в виде MSOL-формулы P и $|P|$ — длина этой формулы.

Теорема 1 (теорема Курселя [7]). Для графа $G = (V, E)$ с $\mathrm{tw}(G) \leqslant k$ существует функция $f : \mathbb{N} \times \mathbb{N} \to \mathbb{N}$ и алгоритм, который проверяет истинность $P(G)$ за время $\mathrm{O}(n \cdot f(|P|, k))$.

Очевидно, что если величина $|P|$ не зависит от $n = |V|$ и $m = |E|$, например, является постоянной или зависит только от k, то по теореме Курселя для графа, удовлетворяющего условиям теоремы 1, существует алгоритм со временем работы $\mathrm{O}(n \cdot f(k))$, то есть FPT-алгоритм относительно древовидной ширины графа G. Таким образом, всякое свойство графа $G = (V, E)$, выраженное в виде конечной формулы MSOL, может быть проверено за полиномиальное время при условии, что граф G имеет ограниченную древовидную ширину. Теорема Курселя демонстрирует широту класса FPT-разрешимых графовых задач, но не даёт точного ответа, как устроен FPT-алгоритм для задачи, отвечающей условиям этой теоремы. Единственное, что известно, — это общая схема такого алгоритма, описанная выше в п. 2.

Для применения теоремы Курселя на практике необходимо: найти MSOL-формулу проверяемого свойства графа; убедиться, что заданный граф G имеет древовидную ширину $\mathrm{tw}(G) \leqslant k$; построить для G дерево декомпозиции ширины k. Относительно последних действий известен следующий результат.

Формулы MSOL, определяющие некоторые свойства графа

Свойство графа	Предикат	Формула MSOL, определяющая предикат
Совпадение рёбер e_i и e_j	$e_i = e_j$	$(\forall v_1)(\text{Incident}(v_1, e_i)) \Leftrightarrow (\text{Incident}(v_1, e_j))$
Смежность вершин v_i и v_j	$\text{Adjacent}(v_i, v_j)$	$\neg(v_i = v_j) \,\&\, (\exists e_1)\ (\text{Incident}(v_i, e_1)\ \&$ $\&\ \text{Incident}(v_j, e_1))$
V_1 — независимое множество вершин	$\text{IndependentSet}(V_1)$	$(\forall v_2)(\forall v_3)((v_2 \in V_1 \,\&\, v_3 \in V_1) \Rightarrow$ $\neg\text{Adjacent}(v_2, v_3))$
V_1 — вершинное покрытие	$\text{VertexCover}(V_1)$	$(\forall e_2)(\exists v_3)(v_3 \in V_1 \,\&\ \text{Incident}(v_3, e_2))$
V_1 — клика	$\text{Clique}(V_1)$	$(\forall v_2)(\forall v_3)((v_2 \in V_1 \,\&\, v_3 \in V_1) \Rightarrow$ $\text{Adjacent}(v_2, v_3))$
V_1 — доминирующее множество	$\text{DominatingSet}(V_1)$	$(\forall v_2)(v_2 \in V_1 \vee (\exists v_3)(v_3 \in V_1 \,\&$ $\&\ \text{Adjacent}(v_2, v_3)))$
Вершинная k-раскраска	$\text{VertexColorable}(V_1, \ldots, V_k)$	$(\forall v_0)(v_0 \in V_1 \vee \ldots \vee v_0 \in V_k) \,\&$ $\text{IndependentSet}(V_1) \,\&\, \ldots$ $\&\, \text{IndependentSet}(V_k)$
E_1 — связность	$\text{Connected}(E_1)$	$(\forall V_2)(\forall V_3)(\neg(\exists v_4)(v_4 \in V_2)) \vee$ $(\neg(\exists v_5)(v_5 \in V_3)) \vee$ $(\exists v_6)(\neg(v_6 \in V_2) \,\&\, \neg(v_6 \in V_3)) \vee$ $(\exists e_7)(\exists v_8)(\exists v_9)(e_7 \in E_1 \,\&\, v_8 \in V_2 \,\&$ $v_9 \in V_3 \,\&\ \text{Incident}(v_8, e_7) \,\&$ $\text{Incident}(v_9, e_7))$
E_1 — гамильтонов цикл	$\text{HamCycle}(E_1)$	$\text{Connected}(E_1) \,\&\, (\forall v_2)(\exists e_3)(\exists e_4)$ $(e_3 \in E_1 \,\&\, e_4 \in E_1 \,\&\, \neg(e_3 = e_4) \,\&$ $\text{Incident}(v_2, e_3) \,\&\ \text{Incident}(v_2, e_4) \,\&$ $(\forall e_5)((e_5 \in E_1 \,\&\, \text{Incident}(v_2, e_5)) \Rightarrow$ $(e_5 = e_3 \vee e_5 = e_4)))$

Теорема 2 (Г. Бодлаендер, Т. Клокс [8]). Существует функция $f : \mathbb{N} \times \mathbb{N} \to \mathbb{N}$ и алгоритм, который за время $O(n \cdot f(k))$ проверяет, имеет ли граф G древовидную ширину $\text{tw}(G) \leqslant k$, и если да, то строит для G дерево декомпозиции ширины k.

Данная теорема свидетельствует о том, что условия применимости теоремы Курселя, касающиеся древовидной ширины графа, могут быть проверены соответствующим FPT-алгоритмом за полиномиальное время.

4. Вычислительная сложность метода и её преодоление с помощью B&S-дерева

Динамическое программирование — метод, рекурсивный по своей природе. Однако рекурсивная реализация данного метода при числе подзадач больше единицы неминуемо влечёт неполиномиальное время вычислений относительно длины входных данных [9]. Согласно соотношению (1), число подзадач на каждом шаге динамического программирования по дереву декомпозиции определяется арностью данного дерева. Очевидно, что в общем случае эта арность больше единицы. Поэтому для получения FPT-алгоритма необходимо динамическое программирование осуществлять итерационно с помощью табличной техники. Между тем эта техника порождает проблему памяти для размещения создаваемых таблиц, поскольку размер всякой таблицы экспоненциально зависит от арности и древовидной ширины применяемого дерева декомпозиции.

Пусть для задачи Π, решаемой на графе $G = (V, E)$ с $\text{tw}(G) \leqslant k$, построено корневое дерево декомпозиции (M, T), $M = \{X_i : i \in I\}$, $T = (I, W)$, ширины k и с узлом r в роли корня. Если дерево T содержит $|I| = O(n)$ узлов, то при динамическом программировании придется хранить $O(n)$ таблиц A_i, $i \in I$. Предположим, что всякая вершина

из X_i может находиться по отношению к возможному решению в q состояниях. Например, в задаче о вершинном покрытии $q = 2$ («принадлежит вершинному покрытию», «не принадлежит вершинному покрытию»), а в задаче о доминирующем множестве $q = 3$ («входит в доминирующее множество», «не входит в доминирующее множество, но доминируется», «не входит в доминирующее множество и не доминируется»). Тогда таблица A_i узла i, являющегося листом в T, имеет размер $\mathrm{O}(kq^k)$, так как $|X_i| \leqslant k+1$. Размер таблицы A_i для внутреннего узла i с двумя прямыми потомками может достигать $\mathrm{O}(kq^{3k})$. Ясно, что чем больше арность дерева T (число прямых потомков для узлов этого дерева) и чем больше ширина (M, T), тем большего размера таблицы могут возникать в процессе вычислений и тем больше времени потребуется для их обработки. Таким образом, проблема памяти — серьезное препятствие практического применения FPT-алгоритмов, основанных на динамическом программировании по дереву декомпозиции. Эта проблема является предметом теоретических исследований многих авторов [3, 8, 10, 11]. В работах практической направленности преимущественно предлагаются различные улучшения дерева декомпозиции. Наиболее известное в этом отношении — «хорошее» дерево декомпозиции (*nice tree decomposition*), описанное Т. Клоксом [12]. Его принято называть деревом декомпозиции Kloks-вида (или просто Kloks-деревом).

Корневое дерево декомпозиции (M, T), $M = \{X_i : i \in I\}$, $T = (I, W)$, отвечает Kloks-виду, если оно удовлетворяет следующим условиям:

1) каждый узел дерева имеет не более двух узлов прямых потомков;
2) если узлу i непосредственно подчинены два узла l и j, то $X_i = X_l = X_j$ (i — узел-соединение);
3) если узел i располагает одним прямым потомком j, то
   - либо $|X_i| = |X_j| + 1$ и $X_j \subset X_i$ (i — узел-вставка),
   - либо $|X_i| = |X_j| - 1$ и $X_i \subset X_j$ (i — узел-удаление).

В работе [12] доказано, что всякое фундаментальное дерево декомпозиции (M, T), имеющее ширину k и $\mathrm{O}(n)$ узлов, может быть преобразовано за время $\mathrm{O}(k^2 n)$ в Kloks-дерево, которое обладает той же шириной и $\mathrm{O}(kn)$ узлами. Заметим, что мощности «мешков» Kloks-дерева, соответствующие смежным узлам этого дерева, различаются не более чем на единицу. Такая сглаженность и бинарность Kloks-дерева даёт возможность размер создаваемых таблиц удерживать в пределах оценки $\mathrm{O}(kq^k)$. Понятно, что это достигается за счет увеличения (возможно в k раз) числа узлов дерева и таблиц динамического программирования.

Желательно иметь такой вид дерева декомпозиции, который бы обеспечивал компромисс между размером и числом таблиц динамического программирования. Для достижения этой цели предлагается использовать бинарное сепараторное дерево декомпозиции (или просто B&S-дерево). Для него характерны следующие особенности: переход от бинарного корневого дерева декомпозиции к B&S-дереву увеличивает число таблиц не более чем в два раза; оно, как и Kloks-дерево, позволяет удерживать размер всякой таблицы динамического программирования в пределах оценки $\mathrm{O}(kq^k)$; для вычисления таблиц динамического программирования по B&S-дереву возможно привлечение аппарата реляционной алгебры и свойств ациклических баз данных. Дадим пояснение и обоснование сказанному.

Корневое дерево декомпозиции (M, T), $M = \{X_i : i \in I\}$, $T = (I, W)$, графа G назовём B&S-деревом, если в нём каждый узел имеет не более двух прямых потомков и относится к одному из четырёх типов узлов:

1) узел-лист — узел, у которого нет потомков;
2) узел-сепаратор — узел s с одним прямым потомком j и узлом i в роли родителя. Всегда X_s — сепаратор графа G и $X_s = X_i \cap X_j \neq \varnothing$, $X_s \subseteq X_i$, $X_s \subseteq X_j$;
3) узел-расширение — узел i с одним прямым потомком s. В данном случае $X_s \subseteq X_i$;
4) узел-соединение — узел i с двумя прямыми потомками l и j. Здесь $X_l \cup X_j \subseteq X_i$.

Утверждение 2. Всякое фундаментальное дерево декомпозиции (M, T) графа G, где $M = \{X_i : i \in I\}$, $T = (I, W)$, имеющее ширину k и $|I| = \mathrm{O}(n)$ узлов, может быть преобразовано за время $\mathrm{O}(n)$ в B&S-дерево, которое обладает той же шириной и $\mathrm{O}(n)$ узлами.

Доказательство. Преобразование заданного дерева декомпозиции (M, T) в B&S-дерево можно выполнить за два шага.

Ш а г 1. Сначала выбрать любой узел $r \in I$ в роли корня. Затем снизить арность дерева до двух следующим образом. Если внутренний узел $i \in I$ обладает одним или двумя прямыми потомками, то ничего не делать. Если внутренний узел $i \in I$ имеет в качестве прямых потомков узлы $j_1, \ldots, j_d$, $d \geqslant 3$, то выполнить «клонирование» узла i в узлы $i_1, \ldots, i_{d-1}$ и приписать каждому из них «мешок» X_i. Отношение подчинённости между узлами установить так, как показано на рис. 1. В результате будет добавлено $\mathrm{O}(n)$ узлов.

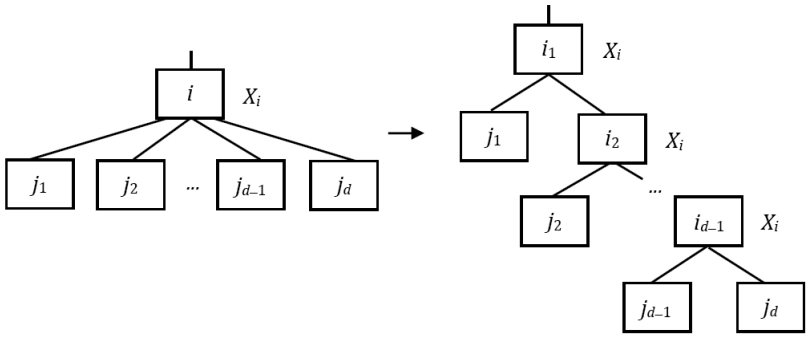


Рис. 1. Схема преобразования дерева декомпозиции к бинарному виду

Ш а г 2. Для любых двух узлов $i, j \in I$, смежных в T и имеющих «мешки» X_i и X_j соответственно, добавить промежуточный узел s и сопоставить ему в качестве «мешка» множество вершин $X_s = X_i \cap X_j \neq \varnothing$ (рис. 2). Исходя из известных свойств дерева декомпозиции [5, с. 67, утверждение 1], множество X_s — сепаратор графа G и, следовательно, s есть узел-сепаратор. Таких узлов будет добавлено $\mathrm{O}(n)$.

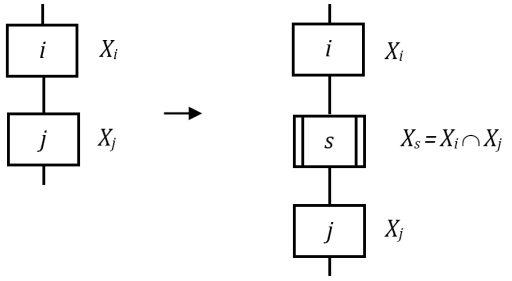


Рис. 2. Схема формирования узла-сепаратора

Общее число узлов в B&S-дереве составит $O(n)$. Время реализации каждого шага — также $O(n)$. Ни одно из действий, предусмотренных шагами 1 и 2, не увеличивает вместимость «мешков» для (M, T), поэтому значение k не изменится. Построенное B&S-дерево — дерево декомпозиции графа G, так как оно очевидным образом отвечает условиям 1–3 определения дерева декомпозиции. ■

Чтобы убедиться, что B&S-дерево сохраняет размер всех создаваемых в процессе динамического программирования таблиц в пределах оценки $O(kq^k)$, сначала рассмотрим, как это достигается при применении Kloks-дерева. Здесь целесообразно прибегнуть к языку реляционной алгебры (алгебры таблиц) [6].

Пусть (M, T), $M = \{X_i : i \in I\}$, $T = (I, W)$, — некоторое корневое бинарное дерево декомпозиции графа $G = (V, E)$ ширины k. Решение подзадачи Π_i вычисляется из таблицы A_i, в которой сформированы все допустимые решения для подграфа $G(Y_i)$, где Y_i задается формулой (1). В общем случае данная таблица может содержать $|Y_i|$ столбцов (по одному столбцу на каждую вершину из Y_i) и $q^{|Y_i|}$ строк (по одной на каждое допустимое решение) при условии, что q — число состояний вершины графа по отношению к допустимому решению. Тот факт, что в таблице A_i столбцами выступают все вершины из Y_i, обозначим через $A_i(Y_i)$. Аналогичным образом обозначим через $A'_i(X_i)$ таблицу, содержащую допустимые решения для подграфа $G(X_i)$, то есть подграфа, индуцированного вершинами только одного «мешка» X_i, и назовем её локальной таблицей. Если узел i является листом, то $A'_i(X_i)$ совпадает с $A_i(Y_i)$. Если i — внутренний узел или корень, то таблица $A_i(Y_i)$ формируется из таблицы $A'_i(X_i)$ и таблиц узлов прямых потомков узла i путем их согласования по строкам. Такое согласование таблиц в реляционной алгебре выполняет естественное соединение $\bowtie$ (далее просто соединение). Данная операция бинарная и отвечает законам коммутативности и ассоциативности. Поэтому общая схема формирования таблицы $A_i(Y_i)$ для внутреннего или корневого узла i, имеющего узлы j и l в качестве прямых потомков, может быть представлена рис. 3. Так как $|X_i| \leqslant k+1$ при всех $i \in I$, то число столбцов всякой таблицы $A_i(Y_i)$, соответствующей узлу i с двумя прямыми потомками, может достигать $3(k+1)$, а число строк в худшем случае (когда операция соединения вырождается в декартово произведение) — значения q^{3k}.

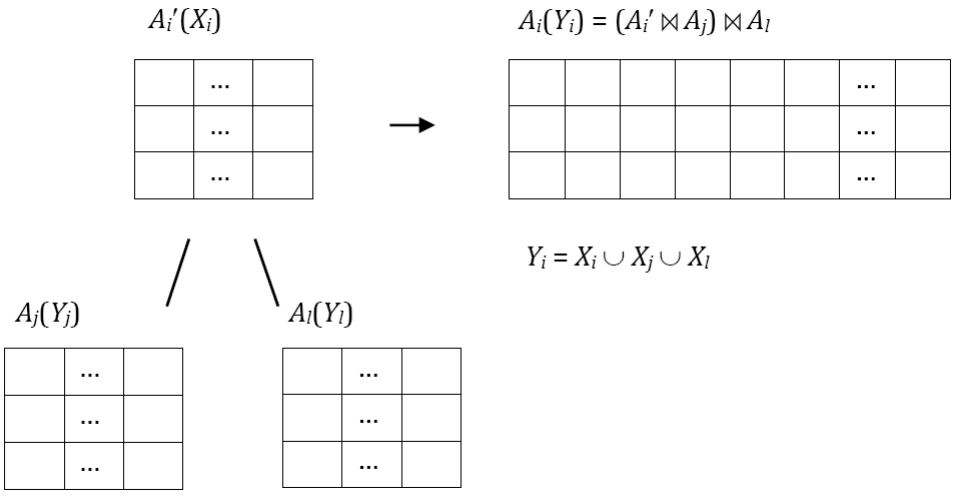


Рис. 3. Создание таблицы для внутреннего узла путем соединения

Пусть (M,T) — Kloks-дерево ширины k. Тогда для всех типов узлов $|Y_i| \leqslant k+1$. Действительно, если узлу i непосредственно подчинены два узла l и j, то

$$X_i = X_l = X_j, \quad Y_i = X_i \cup X_j \cup X_l = X_i, \quad |Y_i| \leqslant k+1.$$

Если i — узел-вставка с прямым потомком j, то

$$X_j \subset X_i, \quad |X_i| = |X_j| + 1, \quad Y_i = X_i \cup X_j = X_i, \quad |Y_i| \leqslant k+1.$$

Если i — узел-удаление с прямым потомком j, то

$$X_i \subset X_j, \ |X_i| = |X_j| - 1, \ Y_i = X_i \cup X_j = X_j, \ |Y_i| \leqslant k+1.$$

Пусть теперь (M,T) — B&S-дерево ширины k. В данном случае также для всех типов узлов $|Y_i| \leqslant k+1$. Для узлов-листьев это очевидно. Рассмотрим внутренние узлы. Заметим, что по построению B&S-дерева узел-сепаратор s всегда имеет только одного родителя (согласно рис. 2 это узел i), одного прямого потомка (это узел j) и $X_s = X_i \cap X_j \neq \varnothing$, $X_s \subseteq X_i$, $X_s \subseteq X_j$. Поэтому для таблицы узла-сепаратора s выполняются отношения $Y_s = X_s \cup X_j = X_j$, $|Y_s| \leqslant k+1$.

Для узла i, играющего роль родителя узла s, справедливы подобные отношения: $Y_i = X_i \cup X_s = X_i$, $|Y_i| \leqslant k+1$. В B&S-дереве любой внутренний узел i является либо узлом-сепаратором, либо родителем одного или двух узлов-сепараторов. Поэтому для него всегда $|Y_i| \leqslant k+1$. Значит, с точки зрения числа столбцов B&S-дерево сопоставимо с Kloks-деревом, хотя имеет в большинстве случаев меньшее число таблиц.

Для Kloks-дерева и B&S-дерева число строк создаваемых таблиц динамического программирования в худшем случае составляет $O(q^k)$. Это теоретическая оценка, и она слишком пессимистичная. На самом деле, как было ранее замечено, всякое дерево декомпозиции (M,T), $M = \{X_i : i \in I\}$, $T = (I,W)$, графа $G = (V,E)$ есть не что иное, как дерево соединений ациклического гиперграфа $H = (V,M)$. Исходя из свойств ациклических гиперграфов, совокупность таблиц A'_i, $i \in I$, обладает монотонным планом соединения. Монотонный план — это такой порядок соединения таблиц, когда число строк всякой промежуточной таблицы не превышает числа строк результирующей таблицы. Монотонный план соединения отвечает обходу дерева соединения ациклического гиперграфа $H = (V,M)$ снизу вверх (от листьев к корню), при этом в роли корня может выступать произвольный узел дерева соединений [6]. Таким образом, при реализации метода динамического программирования по Kloks-дереву или B&S-дереву ширины k в большинстве случаев лишь для таблицы корневого узла число строк достигает значения q^k.

Исключительная особенность B&S-дерева состоит в том, что к нему применимо еще одно свойство ациклических гиперграфов — существование программы полусоединений для совокупности таблиц A'_i, $i \in I$. Представим себе, что мы имеем реляционную базу данных, состоящую из множества таблиц A'_i, $i \in I$. Пусть эта база данных распределена по узлам сети так, что в каждом узле находится только одна таблица, а конфигурация сети задана деревом T. Требуется выполнить запрос к базе данных, в котором надо соединить все таблицы A'_i, $i \in I$. Пусть запрос сформулирован в корневом узле. Тогда процесс вычисления запроса сводится к последовательному соединению таблиц снизу вверх по дереву T с передачей по каналам связи промежуточных таблиц на вышестоящий уровень узлов этой сети. Предположим, что большая часть времени выполнения запроса приходится на время передачи данных. В этом случае

эффективность вычисления запроса зависит от минимизации объёма передаваемых данных. В ациклических реляционных базах данных это достигается за счёт применения программы полусоединений, когда вверх по дереву соединений передаются только те данные, которые участвуют в соединении далее. Очевидно, что процесс выполнения запроса по дереву соединений (M, T) идентичен процессу формирования таблиц динамического программирования по дереву декомпозиции (M, T), а минимизации объёма передаваемых данных — сокращение числа строк промежуточных таблиц динамического программирования.

Поясним действия, выполняемые операцией полусоединения. Пусть заданы две таблицы $A'_i(X_i)$ и $A'_j(X_j)$, отвечающие узлам i и j, где узел j — прямой потомок узла i. Тогда полусоединением $A'_i(X_i)$ и $A'_j(X_j)$ называется таблица, определяемая равенством

$$A'_j \ltimes A'_i = \pi_{X_i}(A'_j \bowtie A'_i), \tag{2}$$

то есть $A'_j \ltimes A'_i$ — эта часть строк таблицы A'_j, которая участвует в соединении со строками из A'_i. По законам реляционной алгебры [6, с. 348]

$$\pi_{X_i}(A'_j \bowtie A'_i) = \pi_{X_i \cap X_j}(A'_j) \bowtie A'_i; \tag{3}$$

$$A'_j \bowtie A'_i = (A'_j \bowtie A'_i) \bowtie A'_i. \tag{4}$$

Если $X_s = X_i \cap X_j$, то из (2)–(4) следует, что

$$A'_j \bowtie A'_i = \pi_{X_s}(A'_j) \bowtie A'_i, \tag{5}$$

где $\pi_{X_s}(A'_j)$ — таблица, которая получается из A_j извлечением столбцов, входящих в X_s, и удалением строк-дубликатов. Так действует операция π, называемая в реляционной алгебре проекцией. Эта операция неизменно уменьшает число строк и столбцов таблицы-операнда. Заметим, что множество $X_s = X_i \cap X_j$ соответствует узлу-сепаратору s в B&S-дереве (рис. 2). Если таблицу для узла-сепаратора получать из таблицы прямого его потомка, применяя операцию проекции, то, согласно (5), операцию соединения таблиц можно осуществить более эффективно с точки зрения времени и памяти. Узел-сепаратор играет в этом случае роль транзита, передающего вверх по дереву только те данные, которые используются для формирования таблицы A_r. Такой эффект от узла-сепаратора гарантируется ацикличностью гиперграфа, ассоциированного с B&S-деревом.

5. Решение задачи о вершинном покрытии методом динамического программирования по B&S-дереву

Задача о вершинном покрытии является типичной NP-трудной задачей на графах, к которой сводятся многие подобные задачи. Напомним, что подмножество $V_1 \subseteq V$ образует вершинное покрытие графа $G = (V, E)$, если каждое ребро из E инцидентно хотя бы одной вершине из V_1. Покрытие называется минимальным, если оно не содержит покрытия с меньшим числом вершин, и наименьшим, если число вершин в нём наименьшее среди всех покрытий графа G. Число вершин в наименьшем покрытии графа G называется числом вершинного покрытия этого графа и обозначается $\beta_0(G)$. В оптимизационной постановке данная задача формулируется следующим образом. Задан граф $G = (V, E)$. Требуется найти число вершинного покрытия $\beta_0(G)$ и наименьшее вершинное покрытие графа G, отвечающее $\beta_0(G)$.

Свойство подмножества $V_1 \subseteq V$ быть вершинным покрытием графа $G = (V, E)$ выражается конечной MSOL-формулой (см. таблицу):

$$\mathrm{VertexCover}(V_1) \Leftrightarrow (\forall e_2)(\exists v_3)(v_3 \in V_1 \,\&\, \mathrm{Incident}(v_3, e_2)). \tag{6}$$

Согласно теореме Курселя, применение к данной задаче метода динамического программирования по дереву декомпозиции приведёт к FPT-алгоритму при условии, что входному графу свойственна ограниченная древовидная ширина. Убедимся в этом. Пусть заданы граф G с $\mathrm{tw}(G) \leqslant k$, его B&S-дерево (M, T), $M = \{X_i : i \in I\}$, $T = (I, W)$, ширины k и с узлом r в роли корня. Реализация метода динамического программирования по B&S-дереву (M, T) предполагает конкретизацию рекуррентной процедуры формирования таблиц $A_i(Y_i)$, то есть определения для каждого из четырёх типов узлов правил, по которым вычисляются все допустимые решения подзадачи Π_i $(i \in I)$. Дадим описание этих правил для задачи о вершинном покрытии, используя язык реляционной алгебры.

Пусть имеем узел-лист i с «мешком» X_i. Для графа $G(X_i)$ задача решается полным перебором. Для этого создается таблица $A_i(Y_i)$ как $A'_i(X_i)$ по следующим правилам. В этой таблице $|X_i| + 1$ столбцов и $2^{|X_i|}$ строк, отдельная строка для каждого из $2^{|X_i|}$ различных подмножеств $Z \subseteq X_i$. Состав вершин, входящих в Z, представляется битовой шкалой b: $b(v) = 1$, если вершина $v \in Z$, и $b(v) = 0$ в противном случае. Столбец $c(Z)$ таблицы $A'_i(X_i)$ содержит характеристику решения: $c(Z) = |Z|$, если Z — вершинное покрытие для $G(X_i)$, иначе $c(Z) = +\infty$. Является ли подмножество Z допустимым решением для Π_i, устанавливается путём проверки на истинность значения предиката (6). Если да, то $c(Z)$ вычисляется по формуле $c(Z) = |Z| = \sum\limits_{v \in X_i} b(v)$.

Рассмотрим узел-сепаратор s с «мешком» X_s и одним прямым потомком j, который имеет «мешок» X_j. В данном случае $X_s \subseteq X_j \subseteq Y_j$, где Y_j — множество вершин исходного графа G, которые принадлежат X_j и всем «мешкам» узлов, являющихся в T потомками узла j. Пусть для узла j ранее уже была построена таблица $A_j(Y_j)$. Тогда таблица $A_s(Y_s)$ вычисляется по формуле

$$A_s = \pi_Q(A_j), \tag{7}$$

где π — операция проекции, которая выбирает из таблицы $A_j(Y_j)$ подмножество столбцов $Q = X_s \cup \{c(Z)\}$. В полученной таблице возможны строки, совпадающие по битовым шкалам и имеющие разные значения $c_1(Z), c_2(Z), \ldots, c_h(Z), h \geqslant 1$. Для всех таких строк полагается

$$c(Z) = \min\{c_1(Z), c_2(Z), \ldots, c_h(Z)\} \tag{8}$$

и в $A_s(Y_s)$ оставляется только одна из них. После этого результирующая таблица $A_s(Y_s)$ содержит только ту информацию из $A_j(Y_j)$, которая необходима для согласования её с таблицей вышестоящего узла. Справедливость формул (7) и (8) вытекает из наследственности свойства, представленного формулой (6): если Z — вершинное покрытие графа $G_i = G(Y_i)$, то множество $Z \cap X_s \subseteq Y_j$ является вершинным покрытием графа $G(X_s)$.

Пусть i — узел-расширение с одним прямым потомком s, для которого $X_s \subseteq X_i$. Предположим, что для узла s ранее уже была создана таблица $A_s(Y_s)$. В данной ситуации вначале формируется локальная таблица $A'_i(X_i)$, которая затем связывается с таблицей $A_s(Y_s)$, и результат записывается в $A_i(Y_i)$:

$$A_i = A'_i \bowtie A_s. \tag{9}$$

Поскольку $X_s \subseteq X_i$, то, согласно (9), в $A_i(Y_i)$ войдут все столбцы из $A'_i(X_i)$ и добавится ещё дополнительный столбец со значениями $c(Z)$ из $A_s(Y_s)$. Пусть $A'_i.c(Z)$ и $A_s.c(Z)$ — характеристики $Z \subseteq X_i$, включённые в результирующую таблицу $A_i(Y_i)$ из A'_i и A_s соответственно. В этих обозначениях формула пересчёта значений $c(Z)$ для A_i имеет вид

$$c(Z) = A'_i.c(Z) + A_s.c(Z) - \sum_{v \in X_s} b(v). \tag{10}$$

Данная формула реализует принцип включения и исключения: мощности допустимых и согласованных между собой решений складываются и вычитается число вершин, учтённых дважды (это вершины из «мешка» узла-сепаратора). Справедливость этой формулы вытекает из наследственности свойства, представленного формулой (6).

Пусть узел i имеет в качестве прямых потомков узлы-сепараторы l и j с «мешками» X_l и X_j соответственно. Допустим, что этим узлам отвечают таблицы $A_l(Y_l)$ и $A_j(Y_j)$, а узлу i — локальная таблица $A'_i(X_i)$. Сначала таблица $A'_i(X_i)$ связывается с таблицей $A_l(Y_l)$ по формулам (9), (10), а затем полученная таблица — с таблицей $A_j(Y_j)$. Таким образом, обработка узла-соединения сводится к двукратному повторению действий, определённых для узла-расширения.

Как только достигается корневой узел r и вычисляется таблица $A_r(V)$, значение числа вершинного покрытия графа G находится по формуле

$$\beta_0(G) = \min\{c(Z) : Z \subseteq Y_r\}. \tag{11}$$

Построение самого вершинного покрытия выполняется при обходе всех узлов дерева T сверху вниз (от корня к листьям) с использованием созданных ранее таблиц динамического программирования. Для хранения $O(n)$ таблиц, каждая из которых имеет размер $O(k2^k)$, необходима память $O(k2^k n)$. Обход дерева T с $O(n)$ узлами можно реализовать за время $O(n)$. Время обработки каждого узла сопоставимо с размером создаваемых таблиц. Поэтому для задачи о вершинном покрытии время работы FPT-алгоритма, реализующего метод динамического программирования по B&S-дереву ширины k, составляет $O(k2^k n)$. Это полностью согласуется с оценкой, приведенной в теореме Курселя. На практике эта оценка может быть более оптимистичной за счет применения B&S-дерева.

Рассмотрим конкретный граф G вместе с его деревом декомпозиции (рис. 4).

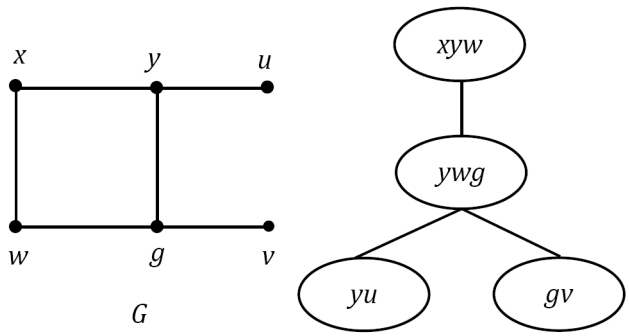


Рис. 4. Граф G и его оптимальное дерево декомпозиции ширины 2

Соответствующее B&S-дерево графа G изображено на рис. 5. Найдём наименьшее вершинное покрытие заданного графа с помощью описанного выше FPT-алгоритма. Процесс подъёма по B&S-дереву начнём с узла 1. Это узел-лист. Узел 2 является узлом-

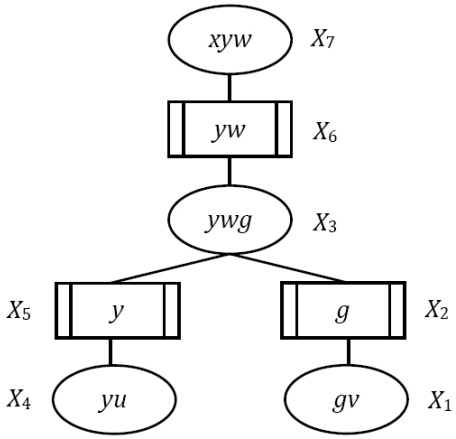


Рис. 5. B&S-дерево ширины 2. Узлу с номером i отвечает «мешок» X_i, $i = 1, 2, \ldots, 7$

сепаратором и родителем для узла 1. Таблицы A_1 и A_2 отвечают узлам 1 и 2 (рис. 6,*а* и *б*). Исходя из (7), (8), $A_2 = \pi_Q(A_1)$, $Q = \{g\} \cup \{c(Z)\}$. Аналогично узлам 4 и 5 соответствуют таблицы A_4, A_5 (рис. 6,*в* и *г*) и $A_5 = \pi_Q(A_4)$, $Q = \{y\} \cup \{c(Z)\}$. Узел 3 — это узел-объединение по отношению к узлам 2 и 5. Для него таблица A'_3 (рис. 6,*д*) содержит локальную информацию для подграфа $G(X_3)$. Соединение A'_3 с таблицей A_2, а затем с таблицей A_5 по формулам (9) и (10) даёт A_3 (рис. 6,*е*). Узел 6 — узел-сепаратор. Для него $A_6 = \pi_Q(A_3)$, $Q = \{y, w\} \cup \{c(Z)\}$ (рис. 6,*ж*). Корневой узел 7 — это узел-расширение по отношению к узлу 6. В соответствии с формулами (9) и (10) таблица A_7 имеет вид, представленный на рис. 6,*з*. Таблица A_7 отвечает корневому узлу. Применение формулы (11) к A_7 даёт $\beta_0(G) = 3$.

A_1

g	v	$c(Z)$
0	0	∞
0	1	1
1	0	1
1	1	2

а

A_2

g	$c(Z)$
0	1
1	1

б

A_4

y	u	$c(Z)$
0	0	∞
0	1	1
1	0	1
1	1	2

в

A_5

y	$c(Z)$
0	1
1	1

г

A_3'

y	w	g	$c(Z)$
0	0	0	∞
0	0	1	1
0	1	0	∞
0	1	1	2
1	0	0	∞
1	0	1	2
1	1	0	2
1	1	1	3

д

A_3

y	w	g	$c(Z)$
0	0	0	∞
0	0	1	2
0	1	0	∞
0	1	1	3
1	0	0	∞
1	0	1	2
1	1	0	3
1	1	1	3

е

A_6

y	w	$c(Z)$
0	0	2
0	1	3
1	0	2
1	1	3

ж

A_7

x	y	w	$c(Z)$
0	0	0	∞
0	0	1	∞
0	1	0	∞
0	1	1	3
1	0	0	3
1	0	1	4
1	1	0	3
1	1	1	4

з

Рис. 6. Таблицы динамического программирования для вычисления вершинного покрытия графа с рис. 4

Спуск по B&S-дереву от корня к листьям определяет следующие наименьшие вершинные покрытия графа G : $\{x, y, g\}, \{x, g, u\}, \{y, w, g\}, \{y, w, v\}$. Заметим, что для исходного графа допустимы деревья декомпозиции, отличные от дерева, указанного на рис. 4, а значит, и B&S-деревья, несхожие с рассмотренным выше B&S-деревом. Решение задачи о вершинном покрытии на их основе приведёт, возможно, к другим оптимальным решениям, хотя неизменным будет значение $\beta_0(G) = 3$.

Очевидно, что приведённый выше FPT-алгоритм можно применять для взвешенной версии задачи о вершинном покрытии, а его идеи относительно правил формирования таблиц динамического программирования для различных типов узлов B&S-дерева — для решения других подобных графовых задач: нахождения наибольшего независимого множества вершин, поиска наибольшей клики, отыскания наименьшего доминирующего множества графа. Конечно, для этих задач необходимо уточнение формул вычисления числовых характеристик решения.

Заключение

Метод динамического программирования на основе дерева декомпозиции — один из современных способов преодоления вычислительной сложности NP-трудных задач выбора с помощью FPT-алгоритмов. В работе предложено бинарное сепараторное дерево декомпозиции, которое обеспечивает компромисс между размером и числом таблиц динамического программирования и расширяет область практического применения данного подхода. Реляционный взгляд на процесс динамического программирования (как процесс соединения таблиц базы данных, распределённой по узлам сети, конфигурация которой задаётся деревом декомпозиции) позволяет применять при решении задач выбора свойства ациклических гиперграфов и способствует алгоритмической конкретизации метода.

ЛИТЕРАТУРА

1. *Downey R. and Fellows M.* Parameterized complexity. New York: Springer Verlag, 1999.
2. *Быкова В. В.* FPT-алгоритмы и их классификация на основе эластичности // Прикладная дискретная математика. 2011. №2(12). С. 40–48.
3. *Bodlaender H. L.* Treewidth: Algorithmic techniques and results // LNCS. 1997. V. 1295. P. 19–36.
4. Компьютер и задачи выбора. М.: Наука, 1989.
5. *Быкова В. В.* Вычислительные аспекты древовидной ширины графа // Прикладная дискретная математика. 2011. №3(13). С. 65–79.
6. *Мейер Д.* Теория реляционных баз данных. М.: Мир, 1987.
7. *Courcelle B.* The monadic second-order logic of graphs. III. Tree decompositions, minors and complexity issues // RAIRO Inform. Theor. Appl. 1992. V. 26(3). P. 257–286.
8. *Bodlaender H. L. and Kloks T.* Efficient and constructive algorithms for the pathwidth and treewidth of graphs // J. Algorithms. 1996. V. 21. P. 358–402.
9. *Быкова В. В.* Математические методы анализа рекурсивных алгоритмов // Журнал СФУ. Математика и физика. 2008. № 1(3). С. 236–246.
10. *Aspvall B., Proskurowski A., and Telle J. A.* Memory requirements for table computations in partial k-tree algorithms // Algorithmica. 2000. V. 27(3). P. 382–394.
11. *Bodlaender H. L. and Fomin F. V.* Tree decompositions with small cost // Discr. Appl. Math. 2005. V. 145(2). P. 143–154.
12. *Kloks T.* Treewidth. Computations and Approximations. Springer Verlag. LNCS. 1994. V. 842.

ПРИКЛАДНАЯ ТЕОРИЯ ГРАФОВ

DOI 10.17223/20710410/16/6
УДК 519.1

ИНДЕКСЫ В ДИНАМИЧЕСКОЙ СИСТЕМЕ ДВОИЧНЫХ ВЕКТОРОВ, АССОЦИИРОВАННЫХ С ОРИЕНТАЦИЯМИ ЦИКЛОВ

А. В. Жаркова

Саратовский государственный университет им. Н. Г. Чернышевского, г. Саратов, Россия

E-mail: VAnastasiyaV@gmail.com

Предлагается алгоритм для подсчёта индексов состояний конечной динамической системы двоичных векторов, ассоциированных с ориентациями циклов, в которой эволюционная функция преобразует вектор с помощью одновременного выполнения следующих действий: начальный 0 и конечная 1 заменяются на 1 и 0 соответственно, каждая диграмма 10 — на 01. Определяется максимальный из индексов системы заданной размерности.

Ключевые слова: *конечная динамическая система, эволюционная функция, двоичные векторы, циклы, индекс.*

Введение

Графовые модели, в которых отказы процессоров интерпретируются как удаление соответствующих вершин, а отказы сетевых каналов — как удаление дуг, занимают важное место в задачах, связанных с отказоустойчивостью компьютерных сетей. Здесь можно выделить следующую конструкцию, получившую самостоятельное значение в теории графов — бесконтурный граф с заданной структурой источников и стоков [1]. В модели [1] в качестве механизма восстановления работоспособности сети предлагается так называемая SER-динамика бесконтурных графов. Это позволяет использовать при изучении модельных графов идеи и методы теории конечных динамических систем и, в частности, динамических систем двоичных векторов (см., например, [2, 3]) — когда имеется естественная двоичная кодировка графов рассматриваемого класса.

Под *конечной динамической системой* понимается пара $(S,\ \delta)$, где S — конечное непустое множество, элементы которого называются *состояниями системы*; $\delta\colon S \to S$ — отображение множества состояний в себя, называемое *эволюционной функцией системы*. Каждой конечной динамической системе сопоставляется *карта* — ориентированный граф с множеством вершин S и дугами, проведенными из каждой вершины $s \in S$ в вершину $\delta(s)$. Этот граф является функциональным, то есть из каждой его вершины выходит точно одна дуга. Компоненты связности графа, задающего динамическую систему, называются её *бассейнами*. Каждый бассейн представляет собой контур с входящими в него деревьями. Контуры называются предельными циклами или *аттракторами*.

Основными проблемами теории конечных динамических систем являются задачи отыскания эволюционных параметров без проведения динамики. К их числу относится индекс состояния — расстояние до аттрактора того бассейна, которому принадлежит

состояние. Автором составлены программы для ЭВМ, позволяющие вычислять различные параметры динамических систем двоичных векторов, ассоциированных с некоторыми типами графов, в частности [4].

В данной работе предлагается алгоритм для подсчёта индексов состояний в динамической системе двоичных векторов, порожденных такими графами, как циклы. Определяется также максимальный из индексов системы заданной размерности.

1. Описание динамической системы

На множестве $B = \bigcup\limits_{n=3}^{\infty} B^n$, где через B^n, $n > 2$, обозначается множество всех двоичных векторов размерности n, рассмотрим динамическую систему (B, θ). Пусть состоянием динамической системы в данный момент времени является вектор $v \in B$. Тогда в следующий момент времени она окажется в состоянии $\theta(v)$, полученном путем одновременного применения следующих правил:

I) если первой компонентой в v является 0 и последней компонентой является 1, то первой компонентой в $\theta(v)$ будет 1, а последней компонентой — 0;

II) если в составе v имеются диграммы вида 10, то в $\theta(v)$ каждая из них заменяется на 01;

III) других отличий между v и $\theta(v)$ нет.

Каждое состояние размерности n при динамике переходит в состояние также размерности n. Таким образом, система (B, θ) в зависимости от n разбивается на конечные подсистемы (B^n, θ), $n > 2$.

Теперь пусть имеется n-звенный цикл c. Выберем в нём какую-либо вершину в качестве начальной и обозначим её через c_0, тогда цикл можно записать как $c = c_0 c_1 \dots c_{n-1} c_n$, где $c_n = c_0$. Придадим каждому ребру цикла произвольную ориентацию. Дуги, имеющие направление по часовой стрелке (от вершины c_0 к вершине c_n), пометим символом 1, а дуги с противоположной ориентацией — символом 0, и последовательно выпишем получившуюся последовательность из нулей и единиц.

Таким образом, каждой ориентации цикла сопоставляется n-мерный двоичный вектор. С другой стороны, каждый такой вектор однозначно определяет некоторую ориентацию цикла, так что между множеством C_n, $n > 2$, всевозможных ориентированных n-звенных циклов указанного вида и множеством B^n, $n > 2$, устанавливается взаимно однозначное соответствие. На языке циклов структура динамической системы вводится следующим образом: если дан некоторый цикл $c \in C_n$, то его динамическим образом $\theta(c)$ является цикл, получаемый из c одновременным превращением всех стоков в источники. Это частный случай динамики бесконтурных графов, введённой в [1]. Заметим, что контуры в динамической системе (C_n, θ), $n > 2$, образуют аттракторы единичной длины. Преобразования ориентаций циклов соответствуют эволюционным преобразованиям соотносимых им двоичных векторов в динамической системе (B^n, θ), $n > 2$, а именно $v_{\theta(c)} \cong \theta(v_c)$.Таким образом, динамические системы (B^n, θ) и (C_n, θ), $n > 2$, изоморфны. На рис. 1 и 2 изображены карты изоморфных динамических систем (B^3, θ) и (C_3, θ).

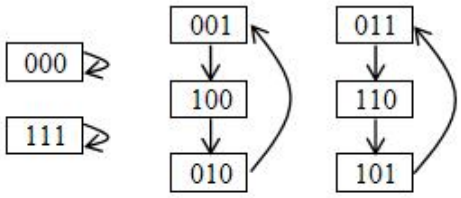


Рис. 1. Карта динамической системы (B^3, θ)

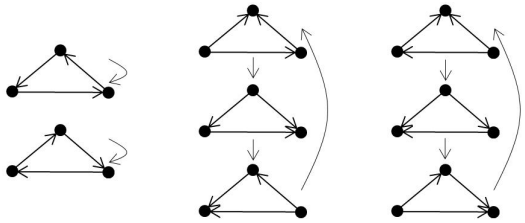


Рис. 2. Карта динамической системы (C_3, θ)

2. Индексы состояний

Будем считать два вектора *циклически идентичными*, если один получается из другого путем циклического сдвига.

Теорема 1. Состояния динамической системы (B^n, θ), $n > 2$, являющиеся циклически идентичными, имеют одинаковые индексы.

Доказательство. Возьмём два циклически идентичных вектора u и v. На языке ориентаций циклов получаем, что соответствующие им графы c_u и c_v являются изоморфными с точностью до переобозначения вершин. Так как данные изоморфные циклы на следующем шаге динамики переходят также в изоморфные циклы, то есть $\theta(c_u) \cong \theta(c_v)$, то они, а значит, и соответствующие им векторы имеют одинаковый индекс. ■

Через $p_c(v)$ обозначим *циклическую плотность* вектора v, то есть количество пар совпадающих соседних компонент в нём с учётом циклического сдвига. Например, $p_c(11001011) = 1 + 3 = 4$. Очевидно, что для состояния v системы (B^n, θ) выполняется $0 \leqslant p_c(v) \leqslant n$. Под *циклическим блоком* будем понимать максимальное по включению множество подряд стоящих нулей (0-блок) или единиц (1-блок) в количестве больше 1 с учётом циклического сдвига. Длина блока — число нулей (единиц) в блоке, уменьшенное на 1. Обозначим через p_c^0, p_c^1 суммы длин с учетом циклического сдвига рассматриваемых 0-блоков и 1-блоков соответственно.

Под *блок-группой* будем понимать последовательность компонент вектора, возможно при циклическом сдвиге, начинающуюся с 0-блока и заканчивающуюся 1-блоком. Под *первичной блок-группой* будем понимать блок-группу, в которой сначала идут только 0-блоки, затем только 1-блоки.

Введем необходимые обозначения: i — индекс состояния; l — длина рассматриваемой блок-группы.

Алгоритм вычисления индекса состояния системы (B, θ)

Индекс $i(v)$ состояния v системы (B, θ) вычисляется исходя из его векторного представления.

I. Если $p_c^0 = 0$ или $p_c^1 = 0$, то $i(v) = 0$.

II. Если $p_c^0 \neq 0$ и $p_c^1 \neq 0$, то выполняем следующие действия.

1. Помечаем в векторе все первичные блок-группы; их количество обозначим h.

2. В каждой блок-группе подсчитываем суммы длин 0- и 1-блоков. Пусть $0 < j \leqslant h$, тогда считаем $p_{c(j)}^0$, $p_{c(j)}^1$ и помечаем блок-группы знаками «$-(x)$», «$=$» и «$+(x)$», если в них $p_{c(j)}^0 > p_{c(j)}^1$, $p_{c(j)}^0 = p_{c(j)}^1$, $p_{c(j)}^0 < p_{c(j)}^1$ соответственно, где $x = |p_{c(j)}^0 - p_{c(j)}^1|$.

3. Если в векторе существуют одновременно «$-$» и «$+$» блок-группы, то идём в п. 4, иначе идём в п. 5.

4. Если в векторе подряд стоят «$-(x)$» блок-группа и «$+(y)$» блок-группа (без учёта остальных компонент и «$=$»-групп между ними, если они имеются), то объединяем их в одну блок-группу, включая возможно стоящие между ними компоненты и «$=$»-группы (их количество в данном случае обозначим за $h_=$), и помечаем знаком «$-(x-y)$», «$=$» или «$+(y-x)$», если $x > y$, $x = y$, $x < y$ соответственно. Полагаем $h := h - 1 - h_=$ и идём в п. 3.

5. Считаем i_j, $0 < j \leqslant h$, согласно следующим правилам:

- в «$-$» блок-группе $i_j = t_j/2 - 1$, где t_j — длина той части блок-группы (если её рассматривать с конца циклически влево), в которой выполняется равенство $p_c^0 = p_c^1$;
- в «$=$» блок-группе $i_j = l_j/2 - 1$;
- в «$+$» блок-группе $i_j = t_j/2 - 1$, где t_j — длина той части блок-группы (если её рассматривать с начала циклически вправо), в которой выполняется равенство $p_c^0 = p_c^1$.

6. Ответ: $i(v) = \max\limits_{0<j\leqslant h} i_j$.

Теорема 2. Предложенный алгоритм вычисления индекса состояния динамической системы (B, θ) корректен.

Доказательство.

I. У вектора $p_c^0 = 0$ или $p_c^1 = 0$. В работе [5] показано, что состояния, для которых $p_c^0 = 0$ или $p_c^1 = 0$, принадлежат аттрактору, поэтому такие состояния имеют нулевой индекс.

II. У вектора v существуют одновременно 0- и 1-блоки. В данном случае индекс вектора равен количеству шагов эволюции, в результате которых получится состояние, содержащее только 0-блоки, только 1-блоки или не содержащее ни 0-, ни 1-блоков.

1. Рассмотрим ситуацию, когда в состоянии идут сначала все 0-блоки, затем все 1-блоки. Заметим, что в таких состояниях существует единственная блок-группа, а именно первичная блок-группа, которая начинается с первого 0-блока и заканчивается последним 1-блоком.

а) $p_c^0 < p_c^1$.

Из доказательства критерия принадлежности состояния аттрактору в работе [5] следует: в таком состоянии на каждом шаге эволюции одновременно 0-блоки начинают движение вправо, 1-блоки начинают двигаться влево за счёт поглощения компонент, стоящих между блоками, с учетом циклического сдвига; когда они оказываются стоящими рядом, то блоки начинают уменьшаться на единицу каждый за счёт поглощения компонент друг друга, пока один из блоков полностью не поглотится, после чего опять продолжаются сдвиги блоков навстречу друг другу, пока не поглотится самый последний 0-блок, а это означает, что достигнут аттрактор.

Таким образом, для вычисления индекса нужно, рассматривая вектор циклически вправо от начала блок-группы, дойти до такой компоненты состояния, чтобы выполнялось равенство $p_c^0 = p_c^1$ — именно этой компонентой и закончится 1-блок, который

поглотит последнюю оставшуюся компоненту 0-блока единичной длины. Тогда действительно $i = t/2 - 1$, где t — длина такой части данной блок-группы. Чётность t показана в работе автора [6].

б) $p_c^0 = p_c^1$.

В работе [5] показано, что такая ситуация возможна только для четного n. Из доказательства критерия принадлежности состояния аттрактору в [5] следует, что в таком состоянии при эволюции 0-блоки и 1-блоки начнут движение навстречу друг другу, пока не окажутся рядом, а затем начнут поглощать друг друга с каждым следующим шагом эволюции, пока не поглотится один из блоков. После этого продолжится аналогичное движение, пока рядом не окажутся последние оставшиеся 0-блок и 1-блок, которые начнут поглощать друг друга с каждым следующим шагом эволюции, пока от них не останется по одной компоненте, что будет означать, что достигнут аттрактор. Таким образом, индекс данного состояния действительно равен $i = l/2 - 1$.

в) $p_c^0 > p_c^1$.

Здесь ситуация аналогична случаю II.1a, только тут поглотятся все 1-блоки. Поэтому для вычисления индекса нужно, рассматривая вектор циклически влево от начала блок-группы, дойти до такой компоненты состояния, чтобы выполнялось равенство $p_c^0 = p_c^1$ — именно этой компонентой и закончится 0-блок, который поглотит последнюю оставшуюся компоненту 1-блока единичной длины. Тогда действительно $i = t/2 - 1$.

2. Теперь рассмотрим ситуацию, когда в состоянии 0-блоки и 1-блоки расположены случайным образом.

Из доказательств предыдущих пунктов можно заключить, что при эволюции такого состояния 0-блоки будут сдвигаться циклически вправо, при этом если они будут встречаться с 1-блоками, то длина 0-блока будет уменьшаться с очередным шагом эволюции. То есть если есть подряд стоящие 0-блоки, то они или все поглотятся, если следующие за ними подряд стоящие 1-блоки в сумме имеют равную или большую сумму длин; или подряд стоящие 0-блоки поглотят сами следующие за ними подряд стоящие 1-блоки и продолжат циклический сдвиг вправо, встречая очередные 1-блоки, если сумма длин этих 0-блоков больше суммы длин 1-блоков. Для 1-блоков ситуация аналогична.

В таком состоянии существует не менее двух первичных блок-групп, обозначим их количество за h. Причём если в первичной блок-группе сумма длин 0-блоков больше, чем сумма длин 1-блоков, то при эволюции на очередном шаге в блок-группе останутся только 0-блоки, которые при последующей эволюции продолжат движение вправо. Если в первичной блок-группе суммы длин 0- и 1-блоков равны, то при эволюции на очередном шаге в блок-группе останутся только чередующиеся нули и единицы. Если же в первичной блок-группе сумма длин 0-блоков меньше, чем сумма длин 1-блоков, то при эволюции на очередном шаге в блок-группе останутся только 1-блоки, которые при последующей эволюции продолжат движение влево.

Пометим в состоянии все первичные блок-группы в зависимости от суммы длин 0- и 1-блоков. А именно, в каждой первичной блок-группе подсчитываем $p^0_{c(j)}$, $p^1_{c(j)}$, где $0 < j \leqslant h$, и помечаем блок-группы знаками «$-(x)$», «$=$» и «$+(x)$», если в них $p^0_{c(j)} > p^1_{c(j)}$, $p^0_{c(j)} = p^1_{c(j)}$, $p^0_{c(j)} < p^1_{c(j)}$ соответственно, где $x = |p^0_{c(j)} - p^1_{c(j)}|$.

Так как на очередном шаге эволюции движение 0- и 1-блоков происходит одновременно, то если в состоянии существуют одновременно «$-(x)$» и «$+(y)$» блок-группы, то в них при эволюции останутся 0-блоки и 1-блоки соответственно, которые по-прежнему будут друг друга поглощать. Значит, игнорируя чередующиеся нули и единицы, а также «$=$» блок-группы, которые могут стоять между «$-(x)$» и «$+(y)$» блок-группами,

объединяем их в одну блок-группу, включая возможно стоящие между ними компоненты и «=»-группы (их количество в данном случае обозначим за $h_=$), и помечаем знаком «$-(x-y)$», «=» или «$+(y-x)$», если $x > y$, $x = y$, $x < y$ соответственно. При этом количество блок-групп уменьшается, то есть $h := h - 1 - h_=$. Если в состоянии после этого существуют одновременно «$-(x)$» и «$+(y)$» блок-группы, то повторяем данную процедуру объединения, пока не избавимся от этой ситуации.

В результате получим состояние, в котором возможна одна из следующих ситуаций по наличию блок-групп: только «−» блок-группы; только «+» блок-группы; только «=» блок-группы; одновременно «−» и «=» блок-группы; одновременно «=» и «+» блок-группы. При эволюции от этих блок-групп в результате останутся: только 0-блоки; только 1-блоки; ни 0-, ни 1-блоков; только 0-блоки или ни 0-, ни 1-блоков; ни 0-, ни 1-блоков или только 1-блоки — это уже состояния, принадлежащие аттрактору. Таким образом, для вычисления индекса состояния нужно подсчитать количество шагов эволюции, за которые от исходных блок-групп останутся вышеперечисленные компоненты, и выбрать из этих чисел максимальное. Используя формулы, полученные в доказательстве (случаи II.1*а*–*в*), считаем i_j, $0 < j \leqslant h$, согласно следующим правилам:

- в «−» блок-группе $i_j = t_j/2 - 1$, где t_j — длина той части блок-группы (если её рассматривать с конца циклически влево), в которой выполняется равенство $p_c^0 = p_c^1$;
- в «=» блок-группе $i_j = l_j/2 - 1$;
- в «+» блок-группе $i_j = t_j/2 - 1$, где t_j — длина той части блок-группы (если её рассматривать с начала циклически вправо), в которой выполняется равенство $p_c^0 = p_c^1$.

Тогда $i(v) = \max\limits_{0<j\leqslant h} i_j$. ■

Для поиска индекса состояния по данному алгоритму вектор сразу разбивается на группы, которые могут быть потом объединены, а затем подсчитываются индексы отдельно для каждой группы и выбирается максимальный из них, поэтому алгоритм имеет линейную сложность $\mathrm{O}(n)$, где n — количество компонент в состоянии.

Пример 1. Подсчитаем индекс состояния $1^{100}010^{100}$.

Считаем индекс состояния согласно п. II алгоритма.

1. Помечаем в векторе все первичные блок-группы: $1^{100}\lrcorner 01 \llcorner 0^{100}$, $h = 1$.
2. Получаем, что в состоянии присутствует единственная «=» блок-группа.
3. В векторе не существует одновременно «−» и «+»-группы, значит, идём в п. 5.
5. Подсчитываем групповой индекс: $i_1 = l_1/2 - 1 = 200/2 - 1 = 99$.
6. Таким образом, $i(v) = 99$, т. е. данное состояние придет к аттрактору за 99 шагов.

Следствие 1. Система (B^n, θ), $n > 2$, имеет максимальный индекс $(n-1)/2 - 1$ при нечетном n и $n/2 - 1$ — при четном n.

Доказательство. Все формулы вычисления индекса состояния рассматривают, по сути, такую часть блок-групп, что при $p_c^0 < p_c^1$ сумма длин 1-блоков равна сумме длин всех 0-блоков этой блок-группы; при $p_c^0 > p_c^1$ сумма длин 0-блоков равна сумме длин всех 1-блоков этой блок-группы, а при $p_c^0 = p_c^1$ берётся длина этой блок-группы. Соответственно при чётном n максимальным является индекс у состояния $00(10)^{n-4}11$, для которого $i(00(10)^{n-4}11) = n/2 - 1$; при нечётном n максимальный индекс у состояния $00(10)^{n-4}11x$, где x — это 0 или 1; для него $i(00(10)^{n-4}11x) = (n-1)/2 - 1$. ■

Заключение

Предложен алгоритм для подсчёта индексов состояний в динамической системе двоичных векторов, ассоциированных с ориентациями циклов, а также определено, чему равен максимальный индекс подсистемы заданной размерности.

ЛИТЕРАТУРА

1. *Barbosa V. C.* An atlas of edge-reversal dynamics. London: Chapman&Hall/CRC, 2001. 372 p.
2. *Салий В. Н.* Об одном классе конечных динамических систем // Вестник Томского госуниверситета. Приложение. 2005. № 14. С. 23–26.
3. *Colon-Reyes O., Laubenbacher R., and Pareigis B.* Boolean monomial dynamical systems // Ann. Combinator. 2004. V. 8. P. 425–439.
4. *Власова А. В.* Исследование эволюционных параметров в динамических системах двоичных векторов // Свидет. РОСПАТЕНТа № 2009614409, зарегистр. 20 августа 2009.
5. *Власова А. В.* Аттракторы динамических систем, ассоциированных с циклами // Прикладная дискретная математика. 2011. № 2(12). С. 90–95.
6. *Власова А. В.* Индексы в динамической системе (B, δ) двоичных векторов // Изв. Сарат. ун-та. Нов. сер. 2011. Т. 11. Сер. Математика. Механика. Информатика. Вып. 3. Ч. 1. С. 116–122.

DOI 10.17223/20710410/16/7
УДК 519.1

КОНГРУЭНЦИИ ЦЕПЕЙ: НЕКОТОРЫЕ КОМБИНАТОРНЫЕ СВОЙСТВА

Е. О. Карманова

Саратовский государственный университет им. Н. Г. Чернышевского, г. Саратов, Россия

E-mail: janekao@mail.ru

Под конгруэнцией цепи понимается отношение эквивалентности на множестве её вершин, все классы которого являются независимыми подмножествами. Доказана теорема 1 о количестве всех конгруэнций для m-рёберной цепи. Для заданного связного графа G теорема 2 находит длину наименьшей цепи, факторизующейся на данный граф.

Ключевые слова: *цепь, конгруэнция, отношение эквивалентности, фактор-граф, число Белла.*

Ориентированным графом (орграфом) называется пара $G = (V, \alpha)$, где V — конечное непустое множество вершин; α — отношение (смежности) на V. Пары в α называются дугами орграфа G. Основные понятия приводятся в соответствии с [1].

Пусть ε — некоторое отношение эквивалентности на множестве вершин V орграфа G. *Фактор-графом* орграфа G по эквивалентности ε называется орграф $G/\varepsilon = (V/\varepsilon, \alpha_\varepsilon)$, где V/ε — множество классов эквивалентности ε; $\alpha_\varepsilon = \{(\varepsilon(v_1), \varepsilon(v_2)) : \exists u_1 \in \varepsilon(v_1)\, \exists u_2 \in \varepsilon(v_2)\ ((u_1, u_2) \in \alpha)\}$.

Пусть K — некоторый класс орграфов. *Конгруэнцией* K-графа G называется такое отношение эквивалентности θ на V, что фактор-граф G/θ является K-графом. В [2] рассмотрен случай, когда K — класс сильно связных орграфов, и предложен способ построения сильно связной конгруэнции произвольного орграфа, наибольшей по числу вершин в фактор-графе. В [3] установлено, что n-элементная ориентированная цепь имеет 2^{n-3} минимальных сильно связных конгруэнций.

В качестве класса K возьмём класс неориентированных графов.

Под неориентированным графом (или, для краткости, графом) понимается пара $G = (V, \alpha)$, где α — симметричное и антирефлексивное отношение на множестве вершин V. В графе пара встречных дуг (u, v), (v, u) рассматривается как один элемент, называемый ребром $\{u, v\}$. Вершины u и v по определению инцидентны ребру $\{u, v\}$.

Множество вершин графа называется независимым, если любые две вершины из этого множества несмежны. Очевидно, что отношение эквивалентности θ на множестве вершин графа $G = (V, \alpha)$ тогда и только тогда будет конгруэнцией этого графа, когда каждый θ-класс образует в G независимое подмножество.

Пусть P_m — неориентированная m-рёберная цепь, и пусть её вершины пронумерованы натуральными числами $0, 1, 2, \ldots, m$. Через $\mathrm{Con}P_m$ обозначим множество всех конгруэнций цепи P_m, а через $E(m)$ — множество всех эквивалентностей на m-элементном множестве. Количество элементов в $E(m)$ называется числом Белла $B(m)$. В работе [4] высказана гипотеза о том, что $|\mathrm{Con}(P_m)| = B(m)$, то есть что $|\mathrm{Con}(P_m)| = |E(m)|$. Ниже приводится доказательство этого факта.

Напомним, что числом Стирлинга второго рода $s_2(m, k)$ называется количество всех эквивалентностей с k классами на m-элементном множестве. Очевидно, что

$\sum\limits_{n=1}^{m} s_2(m,n) = B(m)$. Известно, что $s_2(m,k) = s_2(m-1,k-1) + ks_2(m-1,k)$ для $0 < k < m$ (рекуррентная формула для чисел Стирлинга второго рода). Числа Белла и Стирлинга для графов рассматривались, например, в работе [5].

Пусть $c(P_m,k)$ — количество всех конгруэнций цепи P_m с k классами. Очевидно, что если $\mathrm{Con}(P_m)$ — множество всех конгруэнций цепи P_m, то $|\mathrm{Con}(P_m)| = \sum\limits_{k=2}^{m+1} c(P_m,k)$.

Теорема 1. $|\mathrm{Con}(P_m)| = |E(m)|$.

Доказательство. Методом математической индукции докажем равенство $c(P_m,k+1) = s_2(m,k)$ для $1 \leqslant k \leqslant m$.

Базис индукции: $m = 2$. Цепь P_2 имеет две конгруэнции: одна с двумя классами $\{0,2\},\{1\}$, вторая с тремя классами $\{0\},\{1\},\{2\}$, таким образом, $c(P_2,2) = 1$ и $c(P_2,3) = 1$. Так как двухэлементное множество $\{0,1\}$ имеет два разбиения — одно с одним блоком $\{0,1\}$, второе с двумя блоками $\{0\},\{1\}$, то $s_2(2,1) = 1$, $s_2(2,2) = 1$.

Шаг индукции. Предположим, что $c(P_m,k+1) = s_2(m,k)$, $1 \leqslant k \leqslant m$, верно для цепи P_m и цепей длины меньше m, и покажем, что $c(P_{m+1},k+1) = s_2(m+1,k)$, $1 \leqslant k \leqslant m+1$.

Заметим, что из каждой конгруэнции θ с $k+1$ классами цепи P_m можно получить конгруэнции с $k+1$ классами цепи P_{m+1} двумя способами. Первый способ: добавив в один из k θ-классов (кроме класса $\theta(m)$) вершину $m+1$, получаем разбиение множества вершин цепи P_{m+1}, все $k+1$ классов которого будут независимыми, и, следовательно, разбиение является конгруэнцией цепи P_{m+1}. Таких возможностей имеется $kc(P_m,k+1)$. Второй способ: создаём новый класс $\{m+1\}$, который вместе с классами конгруэнции θ образует разбиение множества вершин цепи P_{m+1}. У этого разбиения все классы образуют независимые подмножества, т. е. получаем конгруэнцию цепи P_{m+1}. Этим способом может быть получено $c(P_m,k)$ конгруэнций цепи P_{m+1}. Таким образом, $c(P_{m+1},k+1) = c(P_m,k) + kc(P_m,k+1)$.

Ввиду предположения индукции и рекуррентной формулы для чисел Стирлинга второго рода запишем $c(P_{m+1},k+1) = c(P_m,k)+kc(P_m,k+1) = s_2(m,k-1)+ks_2(m,k) = s_2(m+1,k)$. Тем самым доказано равенство $c(P_m,k+1) = s_2(m,k)$, $1 \leqslant k \leqslant m$.

Докажем теперь, что $|\mathrm{Con}(P_m)| = |E(m)|$. В самом деле, $|\mathrm{Con}(P_m)| = \sum\limits_{k=2}^{m+1} c(P_m,k) = \sum\limits_{k=1}^{m} c(P_m,k) = \sum\limits_{k=1}^{m} s_2(m,k) = |E(m)|$. ■

В [4] обсуждалась также задача нахождения для данного связного графа G цепи с минимально возможным числом ребер $p(G)$, фактор-графом которой является G.

Задача китайского почтальона для произвольного связного графа [6] состоит в нахождении кратчайшего обхода, начинающегося и заканчивающегося в одной и той же вершине и содержащего все рёбра графа (рёбра в этом обходе могут повторяться в случае необходимости). Используя один из алгоритмов, связанных с этой задачей [7], докажем следующую теорему.

Теорема 2. Пусть G — связный граф. Тогда $p(G) = m+l-k$, где m — количество рёбер графа G; l — количество рёбер в минимальном цепном паросочетании на множестве вершин нечётной степени (нечётных вершин) графа G; k — максимальная из длин цепей в таких паросочетаниях.

Доказательство. Пусть дан произвольный связный граф G с m рёбрами и n вершинами. Найдем длину наименьшей цепи P_r, факторизующейся на G. Так как связный

граф тогда и только тогда является фактор-графом r-рёберной цепи, когда в нем есть обход длины r [4, теорема 1], то найдем длину r минимального обхода R.

Рассмотрим случай, когда граф G не имеет эйлерова пути, поскольку иначе этот граф имеет обход длины m. Тогда в графе G существуют ребра, проходимые в R более чем один раз.

Построим обход графа G, используя алгоритм решения задачи китайского почтальона.

Пусть V^* — множество нечётных вершин графа G. Так как G не содержит эйлеровых цепей, то число таких вершин больше двух и чётное.

Пусть M — множество цепей между концевыми вершинами v_i и v_j, $v_i, v_j \in V^*$, $i \neq j$, в графе G, таких, что никакие две цепи не имеют одинаковых концевых вершин и каждая вершина из V^* является концом какой-нибудь цепи. Такое множество цепей называется цепным паросочетанием; количество цепей в M равно $|V^*|/2$.

Следуя [7], строим все минимальные (по числу рёбер) цепные паросочетания на множестве нечётных вершин V^* графа G. В силу минимальности никакие две цепи в таком паросочетании не имеют общих рёбер. Выберем цепь наибольшей длины из всех минимальных цепных паросочетаний. Пусть это будет цепь длины k из цепного паросочетания M^*, соединяющая вершины u и v. Далее удвоим все ребра, входящие в M^*, кроме рёбер, входящих в цепь максимальной длины.

Таким образом, получим мультиграф G^* с двумя нечётными вершинами u и v. Мультиграф G^* имеет эйлеров путь из u в v (или из v в u). Следовательно, граф G имеет минимальный обход R длины $m+l-k$, где l — количество рёбер в минимальном цепном паросочетании M^* на множестве нечётных вершин V^* графа G. Значит, $p(G) = m + l - k$. ■

Полный граф с n вершинами обозначается K_n. Каждая его вершина имеет степень $n-1$. Число рёбер в графе K_n равно $n(n-1)/2$.

Следствие 1. Для полного графа K_n имеем:

1) $p(K_n) = n(n-1)/2$, если n нечётно;
2) $p(K_n) = n^2/2 - 1$, если n чётно.

Доказательство. Первый случай очевиден, так как полный граф с нечётным количеством вершин эйлеров.

Во втором случае используем результат теоремы 2: $p(G) = m + l - k$. Так как все вершины полного графа K_n при чётном n являются нечётными, то $l = n/2$. Используя тот факт, что в полном графе каждая вершина соединена со всеми другими, получаем $k = 1$. В итоге имеем $p(G) = n(n-1)/2 + n/2 - 1 = n^2/2 - 1$. ■

Звезда — это граф, все ребра которого инцидентны одной и той же вершине. Звезду с m рёбрами будем обозначать S_m.

Следствие 2 (см. также [4]). Для звезды S_m имеем $p(S_m) = 2m - 2$.

Доказательство. Так как у звезды S_m нечётными будут либо все вершины, либо m вершин, то все ребра звезды S_m входят в минимальное цепное паросочетание на множестве её нечётных вершин, то есть $l = m$. Наибольшая длина цепи в паросочетании равна 2. Таким образом, $p(G) = m + l - k = m + m - 2 = 2m - 2$. ■

ЛИТЕРАТУРА

1. *Богомолов А. М., Салий В. Н.* Алгебраические основы теории дискретных систем. М.: Наука, 1997. 367 с.

2. *Мирзаянов М. Р.* Сильно связные конгруэнции ориентированных графов // Теоретические проблемы информатики и ее приложений. Вып. 7. Саратов: Изд-во Сарат. ун-та, 2006. С. 104–114.
3. *Мирзаянов М. Р.* О минимальных сильно связных конгруэнциях ориентированных цепей // Изв. Сарат. ун-та. Сер. Математика. Механика. Информатика. 2006. Т. 6. Вып. 1/2. С. 91–95.
4. *Карманова Е. О.* О конгруэнциях цепей // Прикладная дискретная математика. 2011. № 2(12). С. 96–100.
5. *Duncan B. and Peele R.* Bell and Stirling numbers for Graphs // J. Integ. Sequenc. 2009. V. 12. Article 09.7.1.
6. *Кристофидес Н.* Теория графов. Алгоритмический подход. М.: Мир, 1978. С. 227–239.
7. *Bellman R. and Cooke K. L.* The Konigsberg bridges problem generalized // J. Math. Anal. Appl. 1969. No. 25. P. 1–7.

2012 Прикладная теория графов №2(16)

DOI 10.17223/20710410/16/8

УДК 519.17

СИСТЕМА АБСТРАКТНЫХ СВЯЗНЫХ ПОДГРАФОВ ЛИНЕЙНОГО ГРАФА

В. Н. Салий

Саратовский государственный университет им. Н. Г. Чернышевского, г. Саратов, Россия

E-mail: SaliiVN@info.sgu.ru

Линейным графом называется граф, полученный из некоторой цепи путём какой-либо ориентации её рёбер. Множество всех графов, изоморфных связным подграфам заданного линейного графа L, упорядочивается отношением вложимости. Выясняется, для каких L это упорядоченное множество будет решёткой.

Ключевые слова: *линейный граф, абстрактный подграф, упорядоченное множество, решётка, двоичный вектор, двойственность.*

Под ориентированным графом (далее — граф) понимается пара $G = (V, \alpha)$, где V — конечное непустое множество и $\alpha \subseteq V \times V$ — отношение на нём. Элементы множества V называются вершинами графа, а пары, входящие в отношение смежности α, — дугами. Если $(u, v) \in \alpha$, то говорят, что вершина u является началом дуги (u, v), а вершина v — её концом. Граф G можно изобразить на плоскости, сопоставляя каждой его вершине некоторую точку и проводя отрезок, ориентированный от точки u к точке v, если $(u, v) \in \alpha$.

Если $V' \subseteq V$ и $\alpha' = \alpha \cap (V' \times V')$, то граф $G' = (V', \alpha')$ называется подграфом графа G. При $|V| = n$ граф $G = (V, \alpha)$ имеет 2^n подграфов (включая и пустой подграф).

Изоморфизмом графа $G = (V, \alpha)$ на граф $H = (W, \beta)$ называется биекция $\varphi : V \to W$, такая, что $(\forall u, v \in V)((u, v) \in \alpha \iff (\varphi(u), \varphi(v)) \in \beta)$. Два графа изоморфны тогда и только тогда, когда они допускают одно и то же немое (т. е. с точностью до обозначения вершин) изображение. Если граф G изоморфен некоторому подграфу графа H, то говорят, что G вкладывается в H. Класс $\mathbf{G}'$ графов, изоморфных подграфу G' графа G, можно трактовать как абстрактный подграф графа G, представленный в G подграфом G'.

Вершины $u, v \in V$ графа G называются связанными, если $(\exists u_1, u_2, \dots u_k \in V)$ $((u, u_1) \in \alpha \cup \alpha^{-1})\&((u_1, u_2) \in \alpha \cup \alpha^{-1})\& \dots \&((u_k, v) \in \alpha \cup \alpha^{-1}))$. Граф, в котором любые две вершины связаны, по определению является связным.

Маршрутом с началом u и концом v называется последовательность примыкающих дуг $(u, u_1), (u_1, u_2), \dots, (u_k, v)$. Маршрут можно представить в виде перечисления проходимых вдоль него вершин: $uu_1u_2 \dots u_kv$. Цепь — это маршрут, в котором все вершины разные. Цепь, состоящую из m дуг, обозначим через P_m и будем использовать далее её стандартную запись $P_m = v_0v_1 \dots v_m$.

Линейным графом длины m назовём граф L, полученный из цепи P_m переориентацией некоторых её дуг. Линейные графы рассматривались в [1] в связи с задачей построения минимальных примитивных расширений графов.

Прямое изображение линейного графа L длины m получается, если его вершины v_0, v_1, v_2, …, v_m расположить в этом порядке слева направо на горизонтальной оси. Отобразив этот рисунок относительно вертикальной оси, проведённой правее вершины v_m, и пометив отображённые вершины слева направо как v_0, v_1, v_2, …, v_m, получим обратное изображение графа L.

Если L' — связный подграф линейного графа L, то прямое изображение для L' представляет собой в естественном понимании отрезок в прямом изображении графа L.

Пусть L' и L'' — два линейных графа и L' допускает вложение в L''. Тогда в качестве отрезка в L'' содержится одно из изображений графа L' — прямое или обратное — с согласованным для каждого случая обозначением соответствующих вершин.

Для линейного графа L через $\mathrm{ASubc}\, L$ обозначим совокупность всех линейных графов, вложимых в L. Элементы этого множества можно считать абстрактными связными подграфами графа L. Множество $\mathrm{ASubc}\, L$ упорядочивается отношением вложимости: $\mathbf{L}' \leqslant \mathbf{L}''$ для двух его элементов означает, что граф L' вкладывается в граф L''.

Рассмотрим вопрос о том, для каких линейных графов L упорядоченное множество $\mathrm{ASubc}\, L$ является решёткой, т. е. когда в нём для любых $\mathbf{L}', \mathbf{L}''$ существуют точные грани $\inf(\mathbf{L}', \mathbf{L}'')$ и $\sup(\mathbf{L}', \mathbf{L}'')$.

Для неориентированных (т. е. с симметричным и антирефлексивным отношением смежности) графов G близкие вопросы рассматривались различными авторами. Так, в [2] установлены некоторые общие свойства упорядоченных множеств вида $(\mathrm{ASubc}\, G, \leqslant)$. В [3] показано, что не для всякого G упорядоченное множество связных абстрактных подграфов будет шпернеровым. В [4] дана абстрактная характеризация упорядоченных множеств рассматриваемого вида. В [5] изучаются решёточные упорядочения на множестве $\mathrm{ASubc}\, G$. В других работах (см., например, [6, 7]) авторы отказываются от условия связности и исследуют упорядоченное множество всех вообще абстрактных подграфов данного неориентированного графа. В частности, в [7] доказано, что упорядоченное множество всех абстрактных подграфов неориентированного графа тогда и только тогда будет решёткой, когда либо сам этот граф, либо его дополнение представляет собой полный многодольный граф.

Пусть $\mathbf{b}$ — некоторый двоичный вектор. Двойственным для него называется вектор $\mathbf{b}^\delta$, получаемый из $\mathbf{b}$, если компоненты вектора $\mathbf{b}$ записать в обратном порядке, а затем взаимно заменить в компонентах нули и единицы, т. е. осуществить преобразование $\mathbf{b} \mapsto (\mathbf{b}^{-1})'$. Понятно, что $\mathbf{b}^{\delta\delta} = \mathbf{b}$. Например, для $\mathbf{b} = 011100$ будет $\mathbf{b}^\delta = 110001$.

Под отрезками вектора понимаются блоки, состоящие из подряд идущих компонент этого вектора. Через $\mathrm{ASubc}\, \mathbf{b}$ обозначим совокупность всех попарно не двойственных отрезков двоичного вектора $\mathbf{b}$. На множестве $\mathrm{ASubc}\, \mathbf{b}$ вводится порядок: $\mathbf{b}' \leqslant \mathbf{b}''$, если $\mathbf{b}'$ или $(\mathbf{b}')^\delta$ является отрезком в $\mathbf{b}''$.

Двоичные векторы естественным образом кодируют линейные графы, а именно, линейному графу L длины m соотносится двоичный m-мерный вектор $\mathbf{b} = \mathbf{b}(L)$ путём сопоставления каждой дуге графа в прямом изображении символа 1, если эта дуга направлена от v_0 к v_m, и символа 0 в противном случае. С другой стороны, каждому m-мерному двоичному вектору $\mathbf{b}$ соответствует линейный граф $L = L(\mathbf{b})$ длины m, получающийся из цепи P_m в прямом изображении переориентацией её дуг, согласованной со значениями компонент вектора $\mathbf{b}$. Двоичным кодом каждого связного подграфа линейного графа L, очевидно, является отрезок вектора $\mathbf{b}(L)$.

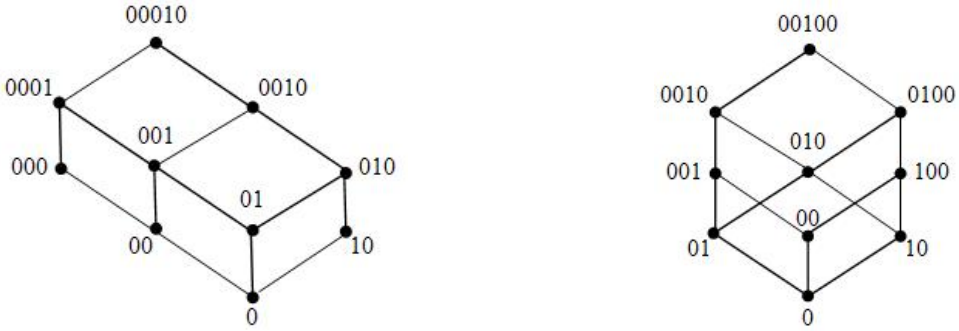
Линейный граф L длины m имеет $m(m+1)/2$ связных подграфов, среди которых могут оказаться и изоморфные, так что в $\mathrm{ASubc}\, \mathbf{b}$ в общем случае будет меньше элементов. Например, у линейного графа L, представленного вектором 0110, имеется 10 связных подграфов: 0, 01, 011, 0110, 1, 11, 110, 1, 10, 0, различны (т. е. неизоморфны) из которых 7, а именно: 0, 01, 011, 0110, 11, 110, 10. У графа L=0101 из 10 связных подграфов различными являются 5: 0, 01, 010, 0101, 10.

Лемма 1. Если L — линейный граф и $\mathbf{b}$ — соответствующий ему двоичный вектор, то упорядоченные множества $(\mathrm{ASubc}\, L, \leqslant)$ и $(\mathrm{ASubc}\, \mathbf{b}, \leqslant)$ изоморфны.

Доказательство. Между множествами $\operatorname{ASubc} L$ и $\operatorname{ASubc}\mathbf{b}$ устанавливается взаимно однозначное соответствие $\mathbf{L}' \mapsto \mathbf{b}(L')$, $\mathbf{b}' \mapsto \mathbf{L}(\mathbf{b}')$. Пусть $\mathbf{L}', \mathbf{L}'' \in \operatorname{ASubc} L$ и $\mathbf{L}' \leqslant \mathbf{L}''$, т. е. L' допускает вложение в L''. Это означает, что в L'' содержится в качестве отрезка граф L' в его прямом или обратном изображении. Тогда в векторе $\mathbf{b}(L'')$ имеется отрезок, совпадающий с $\mathbf{b}(L')$ или с $(\mathbf{b}(L'))^{\delta}$, т. е. $\mathbf{b}(L') \leqslant \mathbf{b}(L'')$. Аналогично, если $\mathbf{b}' \leqslant \mathbf{b}''$ для некоторых $\mathbf{b}', \mathbf{b}'' \in \operatorname{ASubc}\mathbf{b}$, то $L(\mathbf{b}')$ вкладывается в $L(\mathbf{b}'')$, т. е. $\mathbf{L}(\mathbf{b}') \leqslant \mathbf{L}(\mathbf{b}'')$. ■

Из леммы следует, что упорядоченное множество $(\operatorname{ASubc} L, \leqslant)$ абстрактных связных подграфов линейного графа L является решёткой тогда и только тогда, когда решёткой является упорядоченное множество $(\operatorname{ASubc}\mathbf{b}, \leqslant)$ попарно не двойственных отрезков двоичного вектора $\mathbf{b}$, кодирующего граф L.

Пример 1. Рассмотрим диаграммы упорядоченных множеств $(\operatorname{ASubc}\mathbf{b}, \leqslant)$ для двоичных векторов $\mathbf{b} = 00010$ (рис. 1,*а*) и $\mathbf{b} = 00100$ (рис. 1,*б*).



а *б*

Рис. 1. Диаграмма упорядоченного множества $(\operatorname{ASubc}\mathbf{b}, \leqslant)$: *а* — $\mathbf{b} = 00010$; *б* — $\mathbf{b} = 00100$

Как видим, первое из этих упорядоченных множеств является решёткой, а второе — нет, так как в нем не определён, например, $\inf(0010, 100)$: у элементов 0010 и 100 общими нижними гранями являются элементы 0, 00 и 10, но среди них нет наибольшего.

В дальнейшем при записи двоичных векторов будем группировать одинаковые компоненты и использовать экспоненциальную запись: $01100110010 = 0(1^2 0^2)^2 10$ и т. п.

Теорема 1. Пусть L — линейный граф. Упорядоченное множество $(\operatorname{ASubc} L, \leqslant)$ его связных абстрактных подграфов тогда и только тогда является решёткой, когда двоичный вектор $\mathbf{b}$, кодирующий этот граф, имеет вид $\mathbf{b} = 0^k(1^l 0^l)^s 1^t$ или $\mathbf{b} = 1^k(0^l 1^l)^s 0^t$, где k, l, s, t — неотрицательные целые числа.

Доказательство. Необходимость. Пусть для линейного графа L упорядоченное множество $(\operatorname{ASubc} L, \leqslant)$ является решёткой. Вследствие леммы решёткой будет и изоморфное ему упорядоченное множество $(\operatorname{ASubc}\mathbf{b}, \leqslant)$, где $\mathbf{b}$ — двоичный вектор, кодирующий L. Покажем, что в составе вектора $\mathbf{b}$ не могут присутствовать одновременно триграммы 001, 010 и 100. Пусть это не так и в составе $\mathbf{b}$ есть все указанные триграммы. Тогда в составе $\mathbf{b}$ обязательно содержится хотя бы одна из тетраграмм 0010 или 0100. Допустим, что таких тетраграмм в $\mathbf{b}$ нет. По предположению, в составе $\mathbf{b}$ есть триграмма 010. Посмотрим, как может расширяться этот отрезок в $\mathbf{b}$. Добавление 0 ни слева, ни справа невозможно, так как получим запрещённые 0010 или 0100. Значит, слева и справа могут быть только 1, т. е. получатся 0101 или 1010.

Далее, как только появятся два одинаковых соседних символа (00 или 11), получим 0010 (возможно, в виде 1011) или 0100 (возможно, в виде 1101). Следовательно, $\mathbf{b}$ состоит из чередующихся 0 и 1. Но тогда в составе $\mathbf{b}$ нет ни 001, ни 100, что противоречит предположению.

Итак, $\mathbf{b}$ содержит все три триграммы 001, 010 и 100 и хотя бы одну из тетраграмм 0010 или 0100. Если в $\mathbf{b}$ есть 0010, то в $(\mathrm{ASubc}\,\mathbf{b}, \leqslant)$ не существует точной нижней грани для 0010 и 100. Действительно, общими нижними гранями для 0010 и 100 являются 0, 00 и 10, но среди них нет наибольшей. Аналогично, если в $\mathbf{b}$ присутствует отрезок 0100, то не существует точной нижней грани для 0100 и 001: общими нижними гранями для этих элементов будут 0, 00 и 01, но среди них нет наибольшей.

Проведённые рассуждения показывают, что в составе $\mathbf{b}$ нет по крайней мере одной из триграмм 001, 010 или 100. Рассмотрим три случая:

1) если в $\mathbf{b}$ отсутствует 001, то $\mathbf{b} = 1^k(01)^s0^t$, так что этот вектор имеет требуемый вид;
2) если в $\mathbf{b}$ отсутствует 100, то $\mathbf{b} = 0^k(10)^s0^t$ — требуемый вид;
3) если в $\mathbf{b}$ отсутствует 010, то $\mathbf{b} = 0^k1^{l_1}0^{l_2}1^{l_3}\dots0^{l_s}1^t$ или $\mathbf{b} = 1^k0^{l_1}1^{l_2}0^{l_3}\dots1^{l_s}0^t$, где $l_i > 1, 1 \leqslant i \leqslant s$.

   Покажем, что все внутренние блоки имеют одинаковую длину, т. е. что $l_1 = l_2 = l_3 = \dots = l_s$. Предположим, что это не так и в составе $\mathbf{b}$ есть внутренние блоки 0^λ и 0^μ разной длины. Так как эти блоки внутренние, то $\mathbf{b}$ содержит отрезки $10^\lambda1$ и $10^\mu1$. В $\mathrm{ASubc}\,\mathbf{b}$ эти элементы не имеют точной нижней грани, поскольку оба они содержат 01 и 10, но в указанном упорядоченном множестве нет элемента, который содержал бы 01 и 10 и содержался бы в $10^\lambda1$ и в $10^\mu1$. Если же в составе $\mathbf{b}$ есть внутренние блоки 0^λ и 1^μ разной длины, то в $\mathrm{ASubc}\,\mathbf{b}$ не существует $\inf(10^\lambda1, 01^\mu0)$, так как оба эти элемента содержат 01 и 10, но в указанном упорядоченном множестве нет никакого элемента, содержащего 01 и 10 и содержащегося в $10^\lambda1$ и $01^\mu0$. Итак, $\mathbf{b} = 0^k(1^l0^l)^s1^t$ или $\mathbf{b} = 1^k(0^l1^l)^s0^t$, где $l > 1$, что соответствует утверждению теоремы.

Достаточность. Пусть $\mathbf{b} = 0^k(1^l0^l)^s1^t$, где k, l, s, t — неотрицательные целые числа (второй случай рассматривается вполне аналогично). Возможными отрезками в $\mathbf{b}$ являются векторы следующих видов: 1) 0^a; 2) 0^a1^b; 3) $0^a1^l0^b$; 4) $0^a1^l0^l1^b$; 5) $0^a1^l0^l1^l0^b$; 6) $0^a(1^l0^l)^\sigma1^b$; 7) $0^a(1^l0^l)^\sigma1^l0^b$, где a, b, σ — положительные целые числа Покажем, что у любых двух отрезков есть точная нижняя грань в $(\mathrm{ASubc}\,\mathbf{b}, \leqslant)$, т. е. что это упорядоченное множество является нижней полурешёткой. В п. ij указывается точная нижняя грань для отрезков i и j, $1 \leqslant i \leqslant j \leqslant 7$. Заметим ещё, что $\inf((\mathbf{b}_1)^\delta, (\mathbf{b}_2)^\delta) = (\inf(\mathbf{b}_1, \mathbf{b}_2))^\delta$ для любых отрезков $\mathbf{b}_1, \mathbf{b}_2$ вектора $\mathbf{b}$.

11) $\inf(0^a, 0^b) = 0^{\min(a,b)}$;
12) $\inf(0^a, 0^b1^c) = 0^{\min(a,\max(b,c))}$;
13) $\inf(0^a, 0^b1^l0^c) = 0^{\min(a,\max(b,l,c))}$;
14) $\inf(0^a, 0^b1^l0^l1^c) = 0^{\min(a,\max(b,l,c))}$;
15) $\inf(0^a, 0^b1^l0^l1^l0^c) = 0^{\min(a,\max(b,l,c))}$;
16) $\inf(0^a, 0^b(1^l0^l)^\sigma1^c) = 0^{\min(a,\max(b,l,c))}$;
17) $\inf(0^a, 0^b(1^l0^l)^\sigma1^l0^c) = 0^{\min(a,\max(b,l,c))}$;
22) $\inf(0^a1^b, 0^c1^d) = 0^{\min(\max(a,b),\max(c,d))}1^{\min(a,b,c,d)}$;
23) $\inf(0^a1^b, 0^c1^l0^d) = 0^{\min(\max(a,b),\max(c,l))}1^{\min(a,b,\max(a,l))}$;
24) $\inf(0^a1^b, 0^c1^l0^l1^d) = 0^{\min(\max(a,b),\max(c,d,l))}1^{\min(a,b,\max(c,d,l))}$;
25) $\inf(0^a1^b, 0^c1^l0^l1^l0^d) = 0^{\min(\max(a,b),\max(c,l,))}1^{\min(a,b,\max(c,l))}$;

26) $\inf(0^a1^b, 0^c(1^l0^l)^\sigma1^d)=0^{\min(\max(a,b),\max(c,d,l))}1^{\min(a,b,\max(c,d,l))}$;
27) $\inf(0^a1^b, 0^c(1^l0^l)^\sigma1^l0^d)=0^{\min(\max(a,b),\max(c,l,))}1^{\min(a,b,\max(c,l))}$;
33) $\inf(0^a1^l0^b, 0^c1^l0^d)=0^{\min(\max(a,b),\max(c,l))}1^l0^{\min(\max(a,b),\max(d,l))}$;
34) $\inf(0^a1^l0^b, 0^c1^l0^l1^d)=0^{\min(a,\ max(c,d))}1^l0^{\min(b,l)}$;
35) $\inf(0^a1^l0^b, 0^c1^l0^l1^l0^d)=0^{\min(a,\max(c,l))}1^l0^{\min(b,\max(d,l))}$;
36) $\inf(0^a1^l0^b, 0^c(1^l0^l)^\sigma1^d)=0^{\min(a,\max(c,d,l))}1^l0^{\min(b,l)}$;
37) $\inf(0^a1^l0^b, 0^c(1^l0^l)^\sigma1^l0^d)=0^{\min(a,\max(c,d,l))}1^l0^{\min(b,l)}$;
44) $\inf(0^a1^l0^l1^b, 0^c1^l0^l1^d)=0^{\min(\max(a,b),\max(c,d))}1^l0^l1^{\min(a,b,c,d)}$;
45) $\inf(0^a1^l0^l1^b, 0^c1^l0^l1^l0^d)=0^{\min(\max(a,b),\max(c,l))}1^l0^l1^{\min(a,b,c,d)}$;
46) $\inf(0^a1^l0^l1^b, 0^c(1^l0^l)^\sigma1^d)=0^{\min(\max(a,b),\max(c,d,l))}1^l0^l1^{\min(a,b,c,d,l)}$;
47) $\inf(0^a1^l0^l1^b, 0^c(1^l0^l)^\sigma1^l0^d)=0^{\min(\max(a,b),\max(c,l))}1^l0^l1^{\min(a,b,c,l)}$;
55) $\inf(0^a1^l0^l1^l0^b, 0^c1^l0^l1^l0^d)=0^{\min(\max(a,c),\max(c,l))}1^l0^l1^l0^{\min(b,d)}$;
56) $\inf(0^a1^l0^l1^l0^b, 0^c(1^l0^l)^\sigma1^d)=0^{\min(a,\max(c,l))}1^l0^l1^l0^{\min(b,l)}$;
57) $\inf(0^a1^l0^l1^l0^b, 0^c(1^l0^l)^\sigma1^l0^d)=0^{\min(a,\max(c,l))}1^l0^l1^l0^{\min(b,d,l)}$;
66) $\inf(0^a(1^l0^l)^\sigma1^b, 0^c(1^l0^l)^\tau1^d)=0^{\min(a,\max(c,d,l))}(1^l0^l)^{\min(\sigma,\tau)}1^{\min(a,b,c,d,l)}$;
67) $\inf(0^a(1^l0^l)^\sigma1^b, 0^c(1^l0^l)^\tau1^l0^d)=0^{\min(\max(a,b),\max(c,l))}(1^l0^l)^{\min(\sigma,\tau)}1^{\min(b,c,l)}$;
77) $\inf(0^a(1^l0^l)^\sigma1^l0^b, 0^c(1^l0^l)^\tau1^l0^d)=0^{\min(a,\max(c,l))}(1^l0^l)^{\min(\sigma,\tau)}1^l0^{\min(b,d,l)}$.

Таким образом, доказано, что упорядоченное множество $(\mathrm{ASubc}\,\mathbf{b}, \leqslant)$ является нижней полурешёткой, а поскольку в нём есть наибольший элемент $\mathbf{b}$, то оно является и решёткой. ■

ЛИТЕРАТУРА

1. *Салий В. Н.* Минимальные примитивные расширения ориентированных графов // Прикладная дискретная математика. 2008. № 1(1). С. 116–119.
2. *Trotter W. T. and Moore J. I.* Some theorems on graphs and posets // Discr. Math. 1976. V. 15. No. 1. P. 79–84.
3. *Jacobson M. S., Kézdy F. E., and Seif S.* The poset of connected induced subgraphs of a graph need not be Sperner // Order. 1995. V. 12. No. 3. P. 315–318.
4. *Kézdy A. E. and Seif S.* When is a poset isomorphic to the poset of connected induced subgraphs of a graph? // Southwest J. Pure Appl. Math. 1996. V. 1. P. 42–50. (Electronic.)
5. *Nieminen J.* The lattice of connected subgraphs of a connected graph // Comment. Math. Prace Mat. 1980. V. 21. No. 1. P. 187–193.
6. *Adams P., Eggleton R. B., and MacDougall J. A.* Degree sequences and poset structure of order 9 graphs // Proc. XXXV Southeast. Conf. Comb., Graph Theory and Computing. Boca Raton, FL, USA. 2004. V. 166. P. 83–95.
7. *Leach D. and Walsh M.* A characterization of lattice-ordered graphs // Proc. Integers Conf. 2005. N. Y.: Gruyter, 2007. P. 327–332.

2012 Прикладная теория графов №2(16)

DOI 10.17223/20710410/16/9

УДК 519.173.5

ИСПОЛЬЗОВАНИЕ ОСОБЕННОСТЕЙ ВЗВЕШЕННЫХ ГРАФОВ ДЛЯ БОЛЕЕ БЫСТРОГО ОПРЕДЕЛЕНИЯ ИХ ХАРАКТЕРИСТИК

А. Р. Ураков, Т. В. Тимеряев

Уфимский государственный авиационный технический университет, г. Уфа, Россия

E-mail: timeryaev@yandex.ru

Предлагаются алгоритмы быстрого поиска центра, радиуса и диаметра взвешенного графа по матрице кратчайших расстояний, использующие особенности графов реальных дорожных сетей, и приводятся результаты сравнительной оценки алгоритмов с поиском характеристик простым проходом по матрице.

Ключевые слова: *центр графа, радиус графа, диаметр графа, матрица кратчайших расстояний, взвешенный граф.*

Введение

В практических задачах взвешенные графы чаще всего соответствуют некоторой реальной физической сети (например, автомобильных дорог или передачи данных). При проектировании и строительстве таких сетей обязательно учитывается тот факт, что связывание пары узлов ребром имеет некоторую себестоимость, которая зависит от длины участка. В свою очередь, длина участка некоторым образом задает вес ребра. Таким образом, необходимость снижения себестоимости определяет как топологию самой сети, так и вид графа, который её описывает.

Другими словами, взвешенный граф, построенный для такой сети, не является произвольным графом любого вида, а имеет некоторые особенности. В данной работе предлагаются алгоритмы, которые позволяют использовать эти особенности для практических целей.

Рассмотрим взвешенный неориентированный связный граф с неотрицательными весами рёбер. Нас интересуют такие его характеристики: центр, радиус и диаметр. Они определяются следующим образом. Эксцентриситет вершины — максимальное из расстояний от данной вершины до других вершин. Радиус графа — минимальный из эксцентриситетов вершин связного графа; вершина, на которой достигается этот минимум, называется центральной (центром графа). Диаметр графа — это максимум расстояния между вершинами для всех пар вершин. В большинстве случаев эти значения вычисляются вместе с матрицей кратчайших расстояний и соответственно ускорение их поиска достигается ускорением поиска матрицы кратчайших расстояний для различного рода графов [1 – 3]. Однако бывают ситуации, когда матрица кратчайших расстояний известна, а эти величины нет. Некоторые подобные случаи приведены ниже:

— изначально вычислять какую-то из характеристик не требовалось;
— матрица кратчайших расстояний графа каким-то образом изменяется, корректируется без пересчёта кратчайших расстояний между всеми парами вершин;
— требуется вычислять центр, радиус и диаметр не для всего графа, а для некоторого подграфа. При этом таких подграфов может быть несколько, и они могут изменять свой состав.

Тривиальный метод отыскания указанных характеристик состоит в просмотре всех элементов матрицы кратчайших расстояний и определении столбца, максимум значений которого минимален, и максимального элемента матрицы. Сложность нахождения характеристик таким способом составляет $O(n^2)$, где n — количество вершин в графе.

Если использовать особенности графа, построенного по реальной сети, поиск этих величин можно значительно ускорить, применяя следующие алгоритмы.

1. Поиск центра, радиуса и диаметра взвешенного графа

1.1. Алгоритм ускоренного поиска центра и радиуса

Входные данные алгоритма: матрица кратчайших расстояний $M = (m_{ij})$ взвешенного неориентированного связного графа с неотрицательными весами ребер.

Шаг 1. *Подготовка*

Ищется пара вершин (x, y), каждая из которых является самой удаленной для другой:

$$m_{xy} = \max_{i=\overline{1,n}} m_{iy} = \max_{i=\overline{1,n}} m_{ix}. \tag{1}$$

Для этого выбирается любая вершина графа p_1, далее ищется вершина p_2, для которой $m_{p_1p_2} = \max\limits_{i=\overline{1,n}} m_{p_1i}$. Поиск вершин p_j, $j = 2, 3, \ldots$, продолжается до тех пор, пока не будет выполнено равенство $p_{j-1} = p_{j+1}$, тогда $x = p_j$, $y = p_{j+1}$.

Шаг 2. *Поиск*

Рассматриваются все вершины, кроме x, y. Претендент на центр c ищется как вершина, максимум удаления которой от пары (x, y) минимален:

$$\max(m_{cx}, m_{cy}) = \min_{i=\overline{1,n}} (\max(m_{ix}, m_{iy})). \tag{2}$$

После нахождения c величина $r = \max(m_{cx}, m_{cy})$ является первым приближением радиуса графа.

Шаг 3. *Проверка*

Ищется периферийная вершина z, такая, что

$$m_{zc} = \max_{i=\overline{1,n}} m_{ic}. \tag{3}$$

Если $m_{zc} = r$, то, согласно утверждению 1, r — радиус, c — один из центров графа. Если $m_{zc} > r$, радиус R графа лежит в пределах $r \leqslant R \leqslant m_{zc}$, переходим к шагу 4.

Шаг 4. *Попытка замены вершин x и y*

Производится попытка ускорения поиска. Ищется вершина t, для которой $m_{zt} > m_{xy}$. Если такая вершина найдена, то полагаем $x = z$, $y = t$ и переходим к шагу 2. Иначе переходим к шагу 5.

Шаг 5. *Поиск нового претендента на центр*

Среди нерассмотренных претендентов на центр находится вершина d, для которой

$$\max(m_{dx}, m_{dy}, m_{dz}) = \min_{i=\overline{1,n}} (\max(m_{ix}, m_{iy}, m_{iz})).$$

Если максимальное расстояние от этой вершины до других меньше, чем текущая верхняя граница радиуса ($\max\limits_{i=\overline{1,n}} m_{di} < m_{zc}$), то d — новый претендент на центр; полагаем $c = d$, переходим к шагу 3. Если $\max\limits_{i=\overline{1,n}} m_{di} = m_{zc}$, то c — центр, r — радиус. Если $\max\limits_{i=\overline{1,n}} m_{di} > m_{zc}$, то помечаем вершину d как рассмотренную и снова выполняем шаг 5.

Утверждение 1. Пусть дан взвешенный неориентированный связный граф с n вершинами и неотрицательными весами рёбер, x и y — две его произвольные вершины, вершина c определяется по (2) и вершина z определяется по (3). Тогда если $m_{zc} = r = \max(m_{cx}, m_{cy})$, то r — радиус, а c — один из центров графа.

Доказательство. Дано: $\max\limits_{i=\overline{1,n}} m_{ic} = r = \min\limits_{i=\overline{1,n}} (\max(m_{ix}, m_{iy}))$. То есть r — максимальное расстояние от c до любой другой вершины, но в то же время r — минимальное расстояние до одной (а возможно, и обеих) из вершин x, y. Поэтому для любой другой вершины максимум расстояния до вершин из пары (x, y) не меньше r. Следовательно, c — центр (в общем случае не единственный). ■

1.2. Алгоритм быстрого поиска диаметра

Шаги 1 и 2 алгоритма совпадают с соответствующими шагами алгоритма поиска центра и радиуса.

Входные данные алгоритма: матрица кратчайших расстояний $M = (m_{ij})$ взвешенного неориентированного связного графа с неотрицательными весами рёбер.

Шаг 1. *Подготовка*

Ищется пара вершин (x, y), удовлетворяющая (1). Величина $d = m_{xy}$ при этом является текущим претендентом на диаметр.

Шаг 2. *Поиск опорной вершины*

Ищется вершина c, удовлетворяющая (2).

Шаг 3. *Поиск диаметра*

Выполняется поиск диаметра в столбцах матрицы, соответствующих только тем вершинам i, для которых выполнено неравенство $m_{ci} > d/2$.

Утверждение 2. Пусть дан взвешенный неориентированный связный граф с неотрицательными весами рёбер, вершины x, y определяются по (1), вершина c определяется по (2) и $d = m_{xy}$. Тогда диаметр графа равен d или находится в столбцах матрицы кратчайших расстояний тех вершин i, для которых выполнено неравенство $m_{ci} > d/2$.

Доказательство. Если $d = m_{xy}$ не является диаметром графа, то существует пара вершин z, t, такая, что $m_{zt} > m_{xy}$. Для матрицы кратчайших расстояний справедливо $m_{zt} \leqslant m_{zc} + m_{ct}$. Следовательно, $m_{xy} = d < m_{zc} + m_{ct}$, откуда $m_{zc} > d/2$ или $m_{ct} > d/2$. ■

2. Результаты исследования

Проведено сравнение предложенных алгоритмов быстрого нахождения (БН) с простым проходом по матрице кратчайших расстояний (ПМ). В качестве тестовых данных выступали графы, которые используются в транспортных компаниях для расчёта логистических операций и представляют реальные дорожные сети 20 крупнейших российских городов. Характеристики графов приведены в таблице. Производилось вычисление отдельно центра с радиусом, диаметра, а также их совместное нахождение. Вычисления производились несколько раз, в таблице представлены средние значения времени вычисления. Так как тестовые графы обладают небольшим количеством вершин, каждый метод выполнялся 1000 раз и замерялось общее время выполнения. Тестирование производилось на ПК с процессором Intel Core 2 Duo E8400.

Результаты испытаний и характеристики тестовых графов

Количество вершин	Центр, радиус		Диаметр		Центр, радиус и диаметр	
	ПМ, с	БП, с	ПМ, с	БП, с	ПМ, с	БП, с
528	1,4	0,015	0,64	0,015	1,5	0,016
814	3,5	0,047	1,5	0,031	3,6	0,031
1291	8,2	0,047	4,1	0,047	8,4	0,062
1302	8,5	0,047	4,1	0,031	8,7	0,062
1601	13	0,063	6,2	0,047	13	0,063
1641	13	0,23	6,5	0,078	13	0,26
1645	13	0,13	6,6	0,093	14	0,203
1870	18	0,11	8,7	0,078	18	0,13
2059	21	4,2	10	0,203	22	4,3
2150	23	0,094	11	0,062	23	0,094
2194	24	0,093	12	0,078	25	0,093
2280	24	0,16	13	0,109	25	0,203
2424	29	0,14	14	0,109	30	0,19
2484	30	0,81	15	0,109	32	0,83
2542	31	0,19	16	0,094	32	0,22
2896	42	0,45	20	0,14	44	0,5
2921	42	0,109	21	0,094	43	0,16
2964	42	1,7	21	0,13	43	1,8
3060	44	0,25	22	0,11	45	0,23
3364	55	0,23	27	0,204	56	0,28

Алгоритм быстрого поиска центра и радиуса справляется с задачей в среднем в 160 раз быстрее, чем простой проход по матрице кратчайших расстояний, ускорение варьируется от 5 до 400 раз. Для поиска диаметра эти цифры — в 100 раз, от 30 до 190 раз; для совместного поиска центра, радиуса и диаметра — в 130 раз, от 5 до 340 раз.

3. К вопросу об особенностях графов

Заметим, что все дорожные сети, на которых проводилось тестирование и проявился значительный эффект от использования алгоритма, представляют собой попытку охватить заданное количество географических объектов связями с наименьшей суммарной длиной. То есть особенности этих графов происходят из экономической выгоды, практического смысла, а также географического расположения и топологического строения области размещения сети. Очевидно, что необходима некоторая математическая формализация этих особенностей, которая не только ценна сама по себе, но и позволит, во-первых, получить критерий определения легко измеряемых графов, во-вторых, понять, за счёт каких свойств графа достигается эффективность алгоритмов.

Вынуждены признать, что такой формализации пока сделать не удалось. Уже ясно, что хорошо известные свойства графа, такие, как планарность и разреженность, не являются необходимыми. В частности, поиск в алгоритме происходит по матрице кратчайших расстояний, которая, кроме исходного графа, представляет полный граф с ребрами, веса которых соответствуют элементам матрицы. Аналогично, быстрое выполнение алгоритма для какого-либо графа вовсе не означает, что он является планарным. Достаточность этих свойств (планарности и разреженности) также пока обосновать не удалось.

Если анализировать выполнение алгоритма, то можно заметить, что скорость поиска характеристик зависит от выполнения следующего условия. Рассмотрим в исследуемом графе все четверки вершин (i, j, k, l) и все различные треугольники, обра-

зуемые каждыми тремя вершинами из этой четверки, $\triangle_{ijk}$, $\triangle_{ijl}$ и т. д. Назовем треугольник $\triangle_{ijk}$ невырожденным, если во всех комбинациях его вершин для элементов матрицы кратчайших расстояний выполняется $m_{ij} + m_{jk} > m_{ik}$. Рассмотрим четверки вершин (i, j, k, l), у которых все треугольники являются невырожденными. Пусть для площадей этих треугольников выполняются соотношения $S_4 \geqslant S_3 \geqslant S_2 \geqslant S_1$ и $S_4 + S_1 \approx S_3 + S_2$. Наличие в графе таких четвёрок вершин приводит к замедлению поиска, и чем таких четверок больше, тем медленнее происходит поиск. Соответственно условием быстрого поиска характеристик графа является наличие как можно меньшего числа такого рода четверок. К сожалению, анализ графа на выполнение этого условия весьма трудоемок и в вычислительном смысле многократно сложнее выполнения самого алгоритма.

Таким образом, вопрос строгого определения графовых особенностей, способствующих быстрой работе предложенных алгоритмов, остается открытым. На данный момент единственным конструктивным способом проверки графа на эффективность использования алгоритма является выполнение самого алгоритма.

Заключение

Предложены точные алгоритмы быстрого поиска центра, радиуса и диаметра взвешенного графа по матрице кратчайших расстояний. Представленные алгоритмы на тестовых графах нашли центр и радиус в среднем в 160 раз, диаметр в 100 раз, а совместно центр, радиус и диаметр в 130 раз быстрее в сравнении с нахождением этих величин простым проходом по матрице кратчайших расстояний.

Рассматриваемые алгоритмы не гарантируют ускорения вычисления указанных характеристик в общем случае. Используя условия из п. 3, можно построить граф, на котором ускорения не будет или эффект ускорения будет мал. Однако искусственность такого условия позволяет предполагать, что в реальности дорожная сеть, представленная таким графом, не имеет практического смысла.

ЛИТЕРАТУРА

1. *Johnson D. B.* Efficient algorithms for shortest paths in sparse graph // J. ACM. 1977. No. 24. P. 1–13.
2. *Galil Z. and Margalit O.* All pairs shortest distances for graphs with small integer length edges // Information and Computation. 1977. No. 134. P. 103–139.
3. *Zwick U. and Shoshan A.* All pairs shortest paths in undirected graphs with integer weights // Proc. of the 40th IEEE Symposium on Foundations of Computer Science. Washington: IEEE Computer Society Washington, 1999. P. 605–614.

ЛОГИЧЕСКОЕ ПРОЕКТИРОВАНИЕ ДИСКРЕТНЫХ АВТОМАТОВ

DOI 10.17223/20710410/16/10

УДК 518.5

НАХОЖДЕНИЕ РЕЖИМА МАКСИМАЛЬНОГО ЭНЕРГОПОТРЕБЛЕНИЯ ЛОГИЧЕСКОЙ СХЕМЫ

А. Д. Закревский

Объединённый институт проблем информатики НАН Беларуси, г. Минск, Беларусь

E-mail: zakrevskij@tut.by

Известно, что оценка максимального энергопотребления логической схемы позволяет обеспечить её надёжность. Получение этой оценки облегчается предварительным нахождением такого режима работы схемы, при котором её энергопотребление оказывается максимальным. Предлагается метод нахождения этого режима для случая, когда рассматриваемая схема реализует конечный автомат.

Ключевые слова: *логические схемы, режим максимального энергопотребления, логическое проектирование.*

Введение

При проектировании логической схемы важно оценивать максимальную затрату энергии при её функционировании, поскольку чрезмерные затраты энергии могут приводить к выходу схемы из строя. В основном эта энергия тратится при смене входных наборов — комбинаций значений входных булевых переменных схемы. Затраты энергии для каждой пары входных наборов (предшествующего и последующего) можно находить экспериментально или оценивать числом переключаемых транзисторов [1, 2].

Оценивание энергопотребления логической схемы может проводиться более успешно при учёте назначения схемы и способа её проектирования. Это позволяет значительно сократить число рассматриваемых входных наборов и переходов между ними, облегчая тем самым решение данной задачи. Ещё больший эффект достигается при предварительном нахождении режима максимального энергопотребления — соответствующей последовательности наборов значений входных переменных схемы.

Ниже эта задача решается для случая, когда схема синхронна и реализует конечный автомат. Тогда она представляет собой комбинационную схему с однотактной обратной связью, обеспечиваемой регистром. Значения переменных на отмеченных жирными линиями выходных полюсах в текущий момент времени становятся значениями соответствующих входных переменных в следующий такт. Пример такой схемы показан на рис. 1.

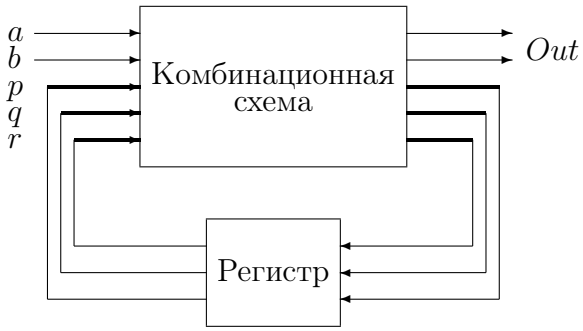


Рис. 1. Схемная реализация конечного автомата

Положим, что комбинационная схема имеет n входных переменных и они делятся на две части: u *свободных* переменных, принимающих произвольные наборы значений, и v *связанных* переменных, наборы значений которых представляют коды возможных состояний автомата ($u + v = n$). В примере на рис. 1 свободными являются переменные a, b, а связанными — переменные p, q, r.

Число w входных наборов схемы в целом равно $s2^u$, где s — число состояний автомата. Эти наборы представляются соответствующими значениями n-компонентного булева вектора $\mathbf{x} = (\mathbf{u}, \mathbf{v})$. Вектор $\mathbf{u}$ представляет набор значений свободных переменных и содержит u компонент. Вектором $\mathbf{v}$ представляется набор значений связанных компонент. Его длина v определяется числом состояний автомата и способом их кодирования.

Предлагаемый в работе метод нахождения режима максимального энергопотребления логической схемы основан на предположении, что чем больше переменных меняют свои значения при очередной смене значения вектора $\mathbf{x}$, тем больше тратится энергии на эту смену.

1. Пример конечного автомата

Рассмотрим в качестве примера автомат с шестью состояниями S_0, S_1, S_2, S_3, S_4, S_5 и двумя свободными переменными a и b. Возможные переходы между состояниями отобразим ориентированным графом переходов G (рис. 2).

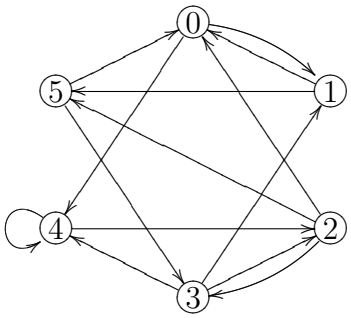


Рис. 2. Граф переходов G

Условия переходов между состояниями S_i и S_k представим соответствующими элементами таблицы. Например, автомат переходит из состояния S_1 в состояние S_0 только при одном наборе значений свободных переменных — когда $ab = 1$, а в состояние S_5

в трёх случаях — когда $\overline{a} \vee \overline{b} = 1$. Заметим, что пустые элементы таблицы соответствуют невозможным переходам.

Условия переходов в автомате

	S_0	S_1	S_2	S_3	S_4	S_5
S_0		a			$\overline{a}$	
S_1	ab					$\overline{a} \vee \overline{b}$
S_2	ab			$a\overline{b}$		$\overline{a}$
S_3		ab	$\overline{b}$		$\overline{a}b$	
S_4			b		$\overline{b}$	
S_5	$a\overline{b}$			$\overline{a} \vee b$		

Проектирование логической схемы, реализующей конечный автомат, начинается с кодирования его состояний наборами значений булевых переменных. Практически эффективные алгоритмы решения этой задачи рассмотрены в [3]. В работе [4] данная задача решается путём такого отображения графа переходов в булев граф, при котором как можно больше рёбер графа переходов отображается на рёбра булева графа. В результате существенно уменьшается число переменных, меняющих свои значения на переходах, и тем самым снижается энергопотребление схемы, реализующей автомат. Состояния автомата кодируются булевыми векторами, соответствующими тем вершинам булева графа, на которые они отобразились.

Так, в данном примере находятся коды состояний автомата, компонентами которых служат соответствующие значения трёх булевых переменных p, q, r. Эти коды представлены столбцами следующей матрицы:

	S_0	S_1	S_2	S_3	S_4	S_5
p	1	0	1	0	1	0
q	0	0	1	1	1	0
r	0	0	0	1	1	1

.

В полученном результате при большинстве переходов в данном автомате меняется значение лишь одной из кодирующих переменных p, q или r — соответствующие рёбра отмечены на рис. 3 одиночными линиями. При остальных переходах меняются значения двух либо трёх переменных — рёбра представлены двойными либо тройными линиями.

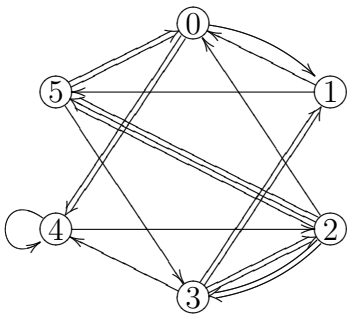


Рис. 3. Граф переходов с отмеченными рёбрами

Вместе с двумя входными переменными кодирующие переменные образуют пятикомпонентный входной булев вектор $\mathbf{x} = (a, b, p, q, r)$ комбинационной схемы, реализующей заданный автомат.

В данном примере автомата с шестью состояниями число возможных входных наборов схемы $w = 6 \cdot 2^2 = 24$, что несколько меньше числа всех наборов $2^5 = 32$. Значительно сильнее сокращается число переходов между наборами: из 1024 различных переходов оказываются возможными лишь $16 \cdot 14 = 224$. Действительно, число переходов в автомате равно 14 и при каждом из них значения переменных p, q, r меняются однозначно. Что же касается входных переменных a и b, то они могут принимать произвольные наборы значений в предшествующий и последующий такты.

2. Нахождение режима максимального энергопотребления схемы

Предлагаемый метод заключается в поиске такой циклической последовательности переходов, которая допустима в рамках используемой формальной модели (хотя и может быть маловероятной), может многократно повторяться и является наиболее энергоёмкой, приводя к максимальным энергозатратам. При решении этой задачи можно ограничиться рассмотрением *простых* (не самопересекающихся) циклов, поскольку любой цикл разлагается на простые и его энергоёмкость не может превысить максимальной энергоёмкости элементов разложения.

Например, простым является цикл из трёх переходов $0 \to 4$, $4 \to 2$ и $2 \to 0$ (см. рис. 3), который проходит через вершины 0, 4, 2 и обозначается 042. Для нашего примера простых циклов 15: 4, 01, 015, 01532, 015342, 042, 04231, 042315, 0425, 153, 042531, 23, 234, 253, 2534.

Для входящих в простые циклы переходов подсчитываются их веса — числа кодирующих переменных, меняющих свои значения. Среднее значение такого числа на переход (обозначим это среднее через $mean\_code$) определяет энергоёмкость цикла. Выбирается цикл, в котором оно максимально. Например, в цикле 042 на переходах $0 \to 4$, $4 \to 2$ и $2 \to 0$ меняют свои значения соответственно две, одна и одна переменная; следовательно, $mean\_code = (2 + 1 + 1)/3 = 4/3$.

Максимальное значение $mean\_code$, равное 2, достигается в циклах 2504, 23 и 325. Заметим, что эти циклы, содержащие переходы с большими весами, можно выявить визуально (см. рис. 3), используя эвристический метод, иллюстрируемый следующим примером.

Берется переход 25 с максимальным весом (3), к нему подсоединяется соседний переход 50 с весом 2, затем переход 04, также с весом 2. В результате получается композиция трёх переходов, последовательно проходящих через вершины 2, 5, 0, 4. Её можно замкнуть переходом 42 с весом 1, получив таким образом цикл 2504. Аналогично находится цикл 325: в этом случае к переходу 25 подсоединяется соседний слева переход 32, затем цикл замыкается переходом 53.

Для каждого перехода в выбранном цикле находятся допустимые наборы значений свободных переменных, т. е. такие наборы, которые обеспечивают выполнение перехода. Например, для переходов $2 \to 5$, $5 \to 0$, $0 \to 4$ и $4 \to 2$ цикла 2504 такие наборы переменных a, b образуют соответственно множества $\{00, 01\}$, $\{10\}$, $\{00, 01\}$ и $\{01, 11\}$. Последовательность этих множеств представим выражением

$$free = 00, 01/10/00, 01/01, 11.$$

Из допустимых наборов выбираются такие, при которых по возможности максимизируется число переменных, меняющих свои значения на переходе. В данном примере такие наборы образуют следующую последовательность значений вектора **u**: 01, 10, 01, 11. Объединяя полученные наборы свободных переменных с соответствующими наборами кодирующих переменных — последовательными значениями вектора **v**: 110,

001, 100, 111, находим искомую циклическую последовательность наборов всех входных переменных схемы — последовательность соответствующих значений вектора $\mathbf{x}$: 01110, 10001, 01100, 11111.

В заключение подсчитываются числа входных переменных комбинационной схемы, меняющих свои значения на переходах, и находится их среднее значение:

$$mean\_input = (5 + 4 + 3 + 2)/4 = 3{,}5.$$

Аналогично подсчитываются энергозатраты при двух других выбранных циклах:

$$\begin{aligned}
23: \quad & free = 10/00, 10; \\
& \mathbf{u}: 10, 00; \quad \mathbf{v}: 110, 011; \quad \mathbf{x}: 10110, 00011; \\
& mean_input = (3 + 3)/2 = 3; \\
325: \quad & free = 00, 10/00, 01/00, 01, 11; \\
& \mathbf{u}: 00, 01, 11; \quad \mathbf{v}: 011, 110, 001; \quad \mathbf{x}: 00011, 01110, 11001; \\
& mean_input = (3 + 4 + 3)/3 = 3{,}33\ldots
\end{aligned}$$

Как видно, максимальной энергоёмкостью обладает цикл 2504. Следовательно, режим максимального энергопотребления рассматриваемой схемы заключается в периодическом повторении простого цикла 2504, который инициируется повторяющейся четвёркой (01, 10, 01, 11) наборов значений свободных переменных a и b.

Заключение

Задача определения максимального энергопотребления спроектированной логической схемы упрощается, когда известно назначение схемы — например, когда она реализует конечный автомат. Известные методы её решения, основанные на подсчёте числа переключаемых логических элементов, довольно трудоёмки, будучи связаны с рассмотрением многочисленных вариантов последовательной смены значений входного вектора. Объём вычислений, производимый при решении этой задачи, значительно сокращается предлагаемым в данной работе методом нахождения режима максимального энергопотребления — соответствующей циклической последовательности значений входного вектора.

ЛИТЕРАТУРА

1. *Ghosh A., Devadas S., Keutzer K., and White J.* Estimation of Average Switching Activity in Combinational and Sequential Circuits // Proc. 29th ACM/IEEE Design Automation Conf. Anaheim, CA, 1992. P. 253–259.
2. *Marculescu D., Marculescu R., and Pedram M.* Theoretical Bounds for Switching Activity Analysis in Finite-State Machines // Proc. 1998 international symposium on low power electronics and design. NY, USA: ACM, 1998. P. 36–41.
3. *Закревский А. Д.* Алгоритмы энергосберегающего кодирования состояний автомата // Информатика. 2011. № 1(29). С. 68–78.
4. *Закревский А. Д.* Алгоритм матричного отображения графа на булев куб // Вестник Томского госуниверситета. Управление, вычислительная техника и информатика. 2011. № 3(16). С. 94–99.

ДИСКРЕТНЫЕ МОДЕЛИ РЕАЛЬНЫХ ПРОЦЕССОВ

DOI 10.17223/20710410/16/11

УДК 577.95

МАТЕМАТИЧЕСКИЕ МОДЕЛИ ИСТОРИЧЕСКИХ ПРОЦЕССОВ

В. А. Воробьев, Ю. В. Березовская

Северный (Арктический) федеральный университет им. М. В. Ломоносова, г. Архангельск, Россия

E-mail: vva100@atnet.ru

Для математического моделирования исторических процессов предлагаются и применяются каузальные модели (К-модели), адекватные свойствам исторических процессов. Рассмотрены базовые процессы истории: демографический процесс и этногенез. Дан обзор истории Западной Европы и её экстраполяция в ближайшее будущее.

Ключевые слова: *каузальная сеть, каузальная модель, математическое моделирование исторических процессов.*

1. Пролог математической истории

Исторические процессы характеризуются дискретными качественными и количественными параметрами. История — результат взаимодействия большого числа людей и других объектов техносферы и биосферы. Все они претерпевают скачкообразные изменения своих состояний, интенсивность которых зависит от численности этих объектов. Эта взаимосвязь и взаимопереход количества и качества давно замечена философами и усложняет математическое моделирование исторических процессов.

Математизация истории началась с моделей демографических процессов [1 – 3] и продолжилась в трудах синергетиков [4 – 7]. Естественным путём математизации истории является мобилизация сохранившихся статистических данных и использование достижений математической статистики. Более абстрактным является написание, исследование и решение подходящих дифференциальных уравнений для исторических процессов — путь, который так успешно использовала физика, теория биологических популяций [8] и труды Римского клуба [9]. На этом пути получены следующие результаты [1 – 3]:

1) Объяснён мировой демографический процесс ограничивающим действием *экологического барьера* на экспоненциальный рост населения Земли.
2) Открыта *экологическая пауза* конца XX века — отставание роста населения Земли от роста ёмкости техносферы — экологической ниши человечества.
3) Обнаружены причины снижения рождаемости по мере индустриального развития — *информационного барьера* — как следствия наложения возраста обучения на фертильный возраст женщин и повышенные нормы потребления в современном обществе.
4) Сделан *прогноз* будущего развития демографической ситуации на планете — рост населения Земли в середине XXI века останавливается и происходит его дальнейшее вымирание.

Пусть t — время, а T — дата от Рождества Христова (РХ). Зависимость ёмкости P (человек) экологической ниши человечества (техносферы) и численности N (человек) населения Земли от времени t описывается следующими уравнениями:

уравнение роста населения по Мальтусу (1798)	$dN/dt = N/\tau(t);$	(1)
экологический барьер (2003 [1])	$N = k \cdot P, k \leqslant 1;$	(2)
уравнение роста (2003 [1]) ниши (техносферы)	$dP/dt = NP/C;$	(3)
экологическое уравнение роста населения	$dN/dt = kNP/C.$	(4)

Характерное время $\tau(t)$ (время увеличения численности населения в e раз) стремительно уменьшалось во всём мире в XIX и XX веках. Напротив, одновременное увеличение величины $\tau(t)$ в развитых странах — это *демографический переход*, т. е. снижение темпов роста населения.

Из уравнений (1)–(4) непосредственно следует:

уравнение Капицы для населения Земли (1999 [5])	$dN/dt = N^2/C;$	(5)
формула фон Фёрстера для населения Земли (1960)	$N = C/(T_0 - T);$	(6)
уравнение годового прироста ниши (2007 [6])	$\Delta P = C/(T_0 - T)^2.$	(7)

Параметры уравнения (6) можно найти из данных С. П. Капицы [4, 5], построив линейный тренд функции $1/N$ с помощью системы Excel, например. Получается, что $C \cong 197{,}005$ млрд человеколет, а $T_0 \cong 2025$ год. Смысл величины C не раскрывается. Уравнение (2) говорит о том, что население, превышающее величину P, не может выжить и обречено на вымирание.

Демографические данные показывают, что уравнения (1)–(7) адекватны только в эпоху *экологического дефицита* (*экодефицита*) — с момента полного заселения экологической ниши где-то около 30 тыс. лет назад [4, 5] и до $\approx$ 1975 г. Избыток M населения должен был удаляться или даже не родиться из-за плохого состояния здоровья голодающих женщин. Уравнение гибели потенциальных и реальных людей от экодефицита имеет вид

$$dM/dt = N/\tau(t) - N^2/C. \tag{8}$$

Рис. 1 показывает, что ёмкость экологической ниши человечества растёт медленнее населения только до 70-х годов XX века. С этого времени действие экологического барьера прекращается.

За момент выхода из экодефицита примем дату $T_{\text{п}} = 1975$ г. Из тех же уравнений нетрудно получить, что в момент $T_{\text{п}}$ характерное время роста $\tau = \tau_{\text{п}} = T_0 - T_{\text{п}}$. Минимальное наблюдаемое время удвоения равно $\tau_{\text{удв}} \cong 19$ лет, а $\tau(t) = 1{,}44 \cdot \tau_{\text{удв}} \cong 27$ лет. Население Земли 1975 г. $\cong$ 4 млрд. Следовательно, в момент выхода из экологического дефицита время удвоения населения Земли составляло $\tau_{\text{удв}} \cong 35$ лет, что близко к действительности.

После 1975 г. наступила *экологическая пауза* (*экопауза*) [1 – 3], когда рост населения, согласно мальтузианскому уравнению (1), стал медленнее роста ёмкости экологической ниши, заданного уравнением (3). В экопаузе исторические явления уже не могут объясняться экологическим дефицитом. Они являются следствием *социального взаимодействия* людей и человеческих качеств, сформированных под давлением экологического барьера. Экопауза выявила *системный кризис* человечества [3], который проявляется прежде всего как *экологический кризис*. Особую тревогу вызывает чрезмерное снижение рождаемости в развитых странах, которое принято называть *демографическим переходом* (рис. 2).

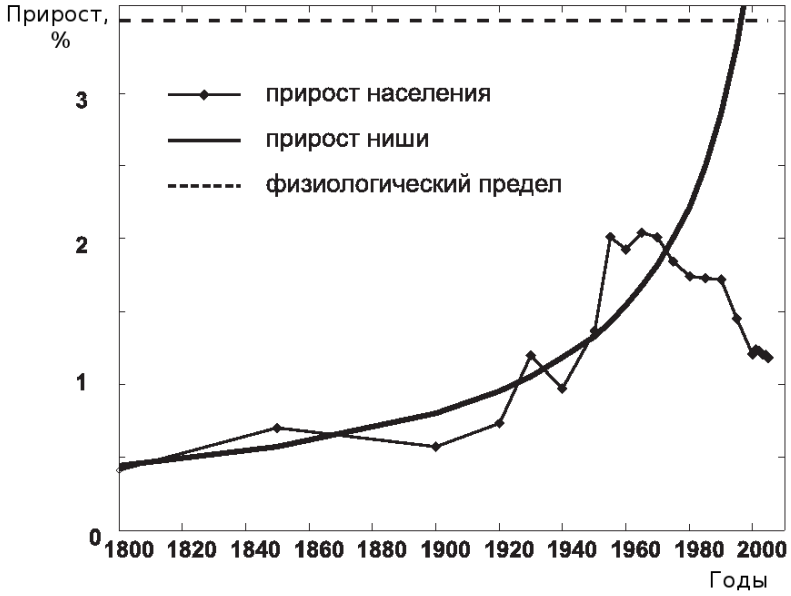


Рис. 1. Конец эпохи экодефицита и начало экологической паузы

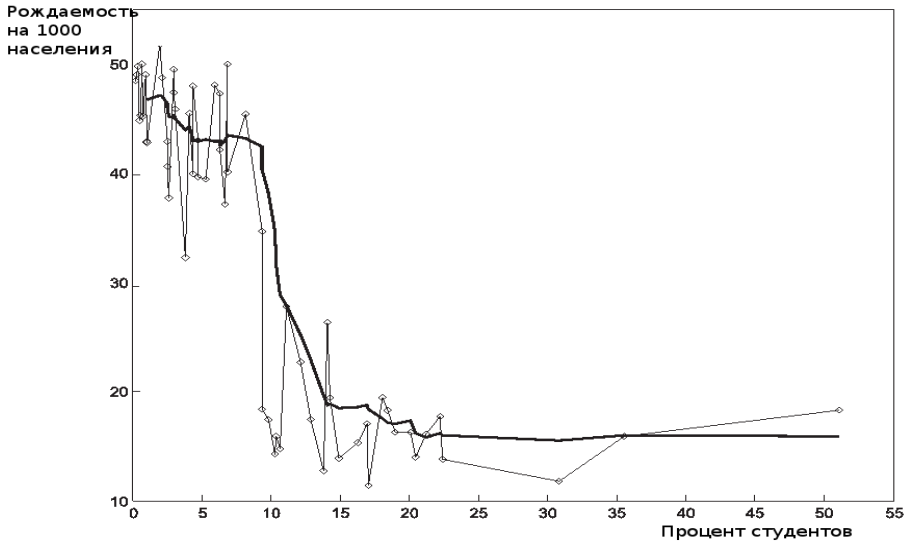


Рис. 2. Зависимость рождаемости от процента студентов. Мировая статистика 1960–1970 г. и её сглаживание

Причина демографического перехода — *информационный барьер*, состоящий в следующем. Демографические данные и социальная статистика [7, 10] выявили [3] явную связь между ростом образования и снижением рождаемости (рис. 2). Аналогичные результаты получены в [6]. Снижение рождаемости можно объяснить тем, что возраст профессионального становления в сложном современном обществе перекрывает репродуктивный возраст женщин. Наиболее образованная молодёжь — студенчество — является *референтной группой* для всей остальной молодёжи. Ценности и стиль жизни студенчества становятся образцом для всех остальных молодых людей, а из этого стиля выбираются наиболее доступные для большинства женщин гедонизм и отказ от

рождения детей. Это явление, грозящее демографической катастрофой всем развитым странам мира, и есть информационный барьер человечества. Действие информационного барьера ярко демонстрируют демографические процессы в странах т. н. *золотого миллиарда*. Если угодно, золотой миллиард вымирает от избытка.

Экстраполяция статистики прироста населения из [5, 7, 10] на ближайшие столетия позволила прогнозировать динамику роста населения Земли [3] и обнаружить, что с середины XX века проявляется тенденция к снижению темпов роста населения. В линейном приближении примерно к 2050 г. рост населения прекратится на уровне $\cong 8 - 9$ млрд, а далее начнётся вымирание человечества (рис. 3).

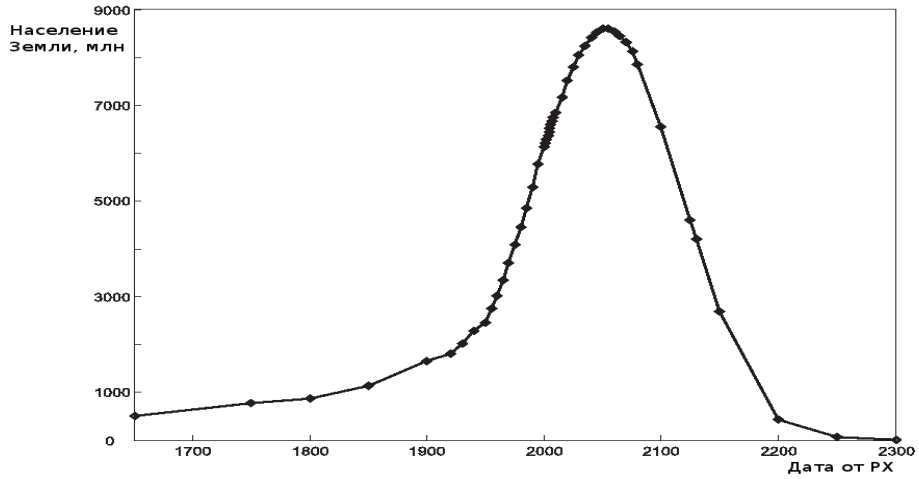


Рис. 3. Импульс населения технической цивилизации

До каких пределов будет происходить это вымирание, предсказать трудно. Можно полагать, что оно остановится на уровне начала индустриальной эпохи, т. е. не более 1 млрд человек, когда изменятся и ценности, и образ жизни человечества. Индустриальная и постиндустриальная эпохи, начавшиеся в конце XVIII — начале XIX века и сопровождавшиеся демографическим взрывом XX века, продлятся до 2200 г. При этом следующее XXII столетие будет ознаменовано стремительным снижением численности людей, и неизвестно, как это отразится на их психике и социальном поведении.

Если допустить, что уравнение (3) роста экологической ниши будет действовать и в экопаузе, пусть даже при уменьшении населения, то к концу XXI века получится совершенно нереальная для Земли ёмкость ниши в 400 млрд человек [3]. Получив этот результат, авторы [3] пришли к выводу, что это возможно, только если вся промышленная деятельность человечества будет вынесена в космос. Однако данные о развитии энергетики говорят о замедлении темпов её роста вместе с экономическим развитием. Более того, по разведанным запасам углеводородов прогнозируется снижение добычи энергоносителей (рис. 4), а атомная энергетика отвергается как экологически опасная.

Но главное возражение против математических моделей и прогнозов состоит в том, что мотивация людей никак не отражается в наших уравнениях. Феноменологические модели недостаточны для понимания и адекватного описания исторических процессов. Они не учитывают социального поведения людей. История человечества — это продукт совместной деятельности множества конкурирующих людей и в эпоху экодефицита, и в экологической паузе, и в преддверии близкой экологической катастрофы.

Рис. 4. Развитие мировой энергетики

2. Каузальное моделирование социальных систем

Итак, необходимо моделировать историю как продукт совместного действия множества объектов: людей, животных, биоты, экологических мест, технологий, идей и т. д. и т. п. Поведение каждого такого объекта в социуме можно представить вероятностным автоматом, переходы которого из состояния в состояние недетерминированы и неоднозначны. Это позволяет моделировать «свободу воли» людей и неопределённость поведения природных объектов. Кроме того, наши автоматы изменяют своё состояние не столько сами по себе, сколько под действием других элементов всей этой сложной системы. Взаимодействие состоит в том, что состояния «воздействующих» автоматов влияют на «изменяемые» автоматы и переводят их в новые состояния. Способ передачи воздействий и структура связей между автоматами не рассматриваются. Вместо этого принята *гипотеза сильного перемешивания* [8].

Метод каузального моделирования социальных процессов строго описан в [11, 12].

Пусть $Q = \{q_1, q_2, \ldots, q_m\}$ — упорядоченное множество имён состояний, в которых могут находиться взаимодействующие *автоматы* — элементы сложной системы: особи в популяции живых существ, устройства в технических системах, люди, организации, фирмы и т. д. Далее будем называть эту систему *популяцией автоматов*. Каждому состоянию $q_i \in Q$ сопоставлено действительное число m_i — *маркер состояния*, задающий число автоматов, находящихся в i-м состоянии. *Внешнее состояние* (окружающая среда) изображается звёздочкой «*», и ему соответствует неопределённый маркер. Общее число автоматов в популяции, не считая внешнего состояния, $N = \sum_{i=1}^{m} m_i$. Множество маркеров m_i, упорядоченное вместе с Q, образует *вектор-состояние* M_t системы в момент времени t. Появление действительных чисел здесь и далее оправдано тем, что количество автоматов (целое число) может быть задано в любых единицах: штуках (целое число), дюжинах, тысячах, миллиардах и т. д. Но тогда целое число «штук» будет задаваться десятичной дробью от тысяч или миллиардов, что и моделируется действительным числом.

Далее, как принято в математических формулах, упоминание имени состояния обозначает и само это состояние, и количество автоматов, находящихся в этом состоянии. Так, буква P есть имя физической величины «вес» и, одновременно, количество килограмм (тонн, грамм, пудов) веса. Заметим, что поведение системы не зависит от единиц измерения. Следовательно, модель системы должна быть *масштабируемой*.

Каузальная модель (К-модель) — это упорядоченное множество D переходов $q_i \to q_j$ автоматов из состояния q_i в состояние q_j под действием других автоматов, находящихся в состоянии q_k. Воздействующие состояния есть причины переходов, а новые состояния — следствия переходов. Отсюда и вытекает название метода моделирования — каузальное, т. е. *причинно-следственное*.

Пусть $d_i \in D$ — переход, тогда ${}^\bullet d_i$ — изменяемые входные состояния, необходимые для его исполнения, $d_i^\bullet$ — выходные состояния, возникающие в результате перехода d_i. Выражение ${}^\bullet d_i \to d_i^\bullet$ задаёт все переходы из ${}^\bullet d_i$ в $d_i^\bullet$ под действием ${}^\bullet d_i$. В частности, если множества ${}^\bullet d_i$ и $d_i^\bullet$ содержат по два состояния, то такой переход называется *простым*. Например, переход $q_k, q_i \to q_k, q_j$ меняет q_i на q_j, а состояние q_k только воздействует, но сохраняется. Используемые виды простых переходов перечислены в табл. 1.

Таблица 1

Различные виды простых переходов

1	Простой	$q_k, q_i \to q_k, q_j$	Состояние q_k переводит q_i в q_j
2	Удаляющий	$q_k, q_i \to q_j$	Состояния q_k, q_i удаляются и порождают q_j
3	Сохраняющий	$q_k, q_i \to q_i, q_k, q_j$	Состояния q_k, q_i сохраняются и порождают q_j
4	Остаточный	$q_k, q_i \to q_k, q_l$	Реализует переход $q_i \to q_l$ для тех автоматов, которые не выполнили его в переходах типа 1, 2, 3
5	Ингибиторный	${}^\wedge q_k, q_i \to q_j$	${}^\wedge q_k$ означает отсутствие ингибитора — состояния q_k, мешающего переходу $q_i \to q_j$

В общем случае на входе и выходе перехода может быть по нескольку автоматов в одном и том же состоянии. Зададим функции $In : Q \times D \to \{k_{ji}\}$ и $Out : D \times Q \to \{k_{ij}\}$, где k_{ji}, k_{ij} — действительные числа, указывающие для каждого перехода d_i число k_{ji} возбуждающих и число k_{ij} порождаемых автоматов. Функции In и Out задаются матрицами, строкам которых соответствуют имена состояний, столбцам — имена переходов, а элементы — числа k_{ji} и k_{ij}.

Пусть In_i — i-й столбец матрицы In, тогда переход d_i будет готов к срабатыванию тогда и только тогда, когда $M_t \geqslant In_i$. В каждом такте моделирования срабатывают не все готовые переходы d_i, а некоторое их число $R_i = p_i P_i(M_t({}^\bullet d_i))$. Здесь коэффициент p_i задаётся «руками» или извлекается из наблюдений, обычно это вероятность; $M_t({}^\bullet d_i)$ — маркировка входа в переход d_i; $P_i(M_t({}^\bullet d_i))$ — функция, зависящая от типа перехода. *Интенсивность* R_i — это число автоматов, изменивших состояние за один такт модельного времени. Множество интенсивностей R_i, упорядоченное вместе с D, есть *вектор интенсивностей* переходов R.

Функция $P_i(M_t({}^\bullet d_i))$ зависит от типа перехода d_i. Основные типы переходов — *линейные* и *нелинейные*. Линейные переходы соответствуют *дальнодействию* в системе, когда все всех «видят» и могут взаимодействовать. Нелинейные переходы соответствуют *близкодействию*: *раствор* — равномерному распределению автоматов по всей

системе мощности N (как молекулы в растворе) и *смесь* — собранию взаимодействующих автоматов в одном месте (как птичий базар). Поскольку линейные и нелинейные переходы возможны для всех пяти видов простого перехода, то получается 15 типов только простых переходов. Кроме того, в качестве действующих могут использоваться задержанные на τ тактов значения $M_{t-\tau}({}^\bullet d_i)$. Итак, в общем случае описание перехода — это выражение вида

$$ {}^\bullet d_i \to d_i^\bullet, p_i, \text{тип перехода}, \text{задержки } {}^\bullet d_i. $$

Для простых переходов вида $q_k, q_i \to q_k, q_j$ примем более простую запись

$$ q_k : q_j > q_l, p_i, \text{тип перехода}, \text{задержка } m_k, \text{задержка } m_j. $$

Для линейного перехода $P_i = \min\{M_t({}^\bullet d_i)\}$, для простого перехода в растворе $P_i = m_k m_j / N$, в смеси — $P_i = m_k m_j / (m_k + m_j)$.

Разность матриц $Out - In = D$ называется *девиатором*. В [12] показано, что динамика популяции автоматов задаётся системой дифференциальных уравнений (9) с нормировкой (10):

$$ dM/dt = D \cdot R; \tag{9} $$

$$ \sum_{i=1}^{n} m_{it} = N_t, \tag{10} $$

где $\cdot$ — матричное умножение; m_{it} — маркировка i-го состояния; N_t — общее число автоматов в момент времени t.

Для моделирования популяции следует задать начальную маркировку M_0 и побеспокоиться о единой размерности для состояний. Состояния, как было постулировано ранее, качественно различны, и их маркировки имеют разную размерность. Для согласования размерностей каждому автомату в состоянии q_i должен соответствовать один автомат в любом качественно ином состоянии q_j. Для этого следует соответствующим образом подбирать единицы измерения. Так, в следующих далее демографических и экологических моделях исторического процесса за единицу измерения принят один *человек*, а ему соответствует одно *место* проживания в экологической нише или техносфере, одно *трудовое* усилие для создания места проживания, одно *дитё* и т. д.

Ниже используется программа «Популяция», которая моделирует поведение системы по правилам срабатывания простых переходов в К-модели. На рис. 5 показан интерфейс программы при моделировании демографического взрыва согласно уравнению (5) С. П. Капицы [5].

Графическое представление К-моделей будем называть каузальными сетями (К-сетями). *К-сеть* — двудольный граф, подобный сети Петри [12]. Функционирование К-модели эквивалентно численному решению системы дифференциальных уравнений динамики средних. Поэтому они даже не выписываются. Впрочем, их тоже можно получить. К-модель тем точнее, чем меньше такт моделирования (интегрирования). Однако достаточно малый такт можно далее не уменьшать, так как это не меняет поведение модели. Таким образом, и в отношении единиц времени К-модель *масштабируема*.

Следует особо отметить, что состояние автомата в популяции соответствует не конкретному объекту, а некоторому поведению. Например, один человек может в разное время вести себя по-разному, т. е. иметь разные состояния: родитель, творец, потребитель и т. д.

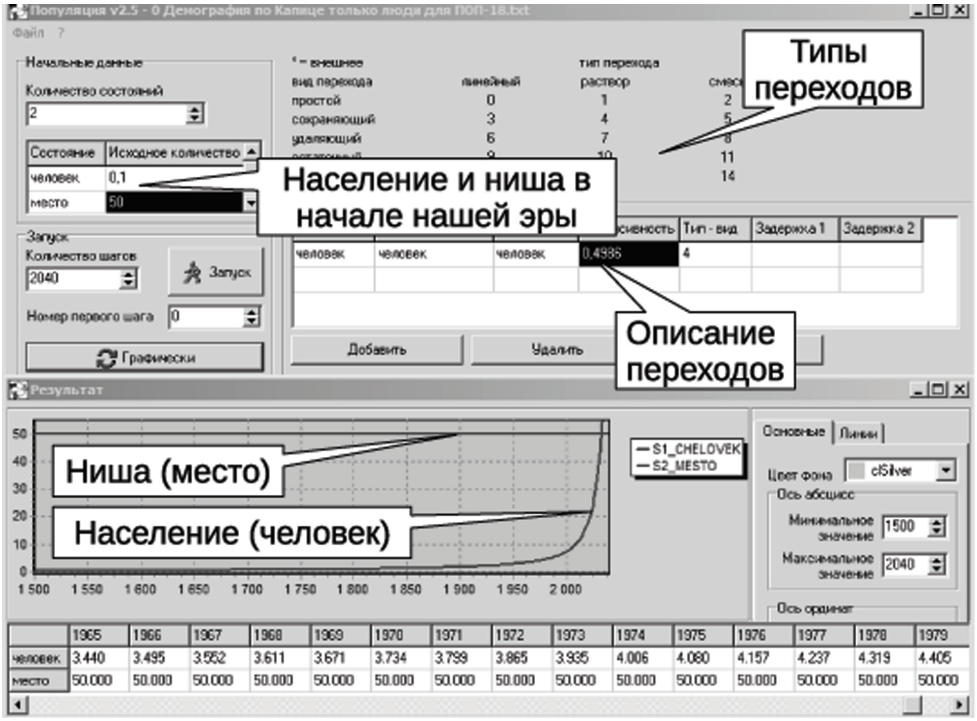


Рис. 5. Интерфейс программы ПОПУЛЯЦИЯ, демографический процесс с 1500 по 2040 г. и демографический взрыв индустриальной эпохи

3. K-модель демографического взрыва

Феноменологические уравнения фон Фёрстера (4) и Капицы (5) достаточно точно моделируют взрывной демографический процесс, но ничего не объясняют. Уравнения (4) и (5), в отличие от мальтузианского уравнения (1), парадоксальны, поскольку противоречат физиологии. Попытки Капицы и его последователей дать объяснение полученному результату следует признать неудачными. Каузальное моделирование по определению невозможно, если исследователь не укажет причинно-следственные связи между состояниями и событиями в системе. Поэтому мы вынуждены с самого начала принять хоть какую-то объяснительную гипотезу относительно этих связей. Таковая напрашивается исходя из экологических уравнений (2)–(4), выведенных в [1 – 3]. Делитель $C = 197{,}005$ млрд человеколет [челет] в уравнении С. П. Капицы — не что иное, как экологическая ёмкость биоты Земли, в которой растворены все люди и животные. Следовательно, их близкодействующие взаимодействия можно рассматривать как множество нелинейных переходов в растворе. И размерности всех величин в дальнейшем привязаны к величине C: [челет] или [человек] по смыслу переходов.

Результат K-моделирования по Капице представлен на рис. 5. K-модель содержит один переход и по-прежнему ничего не объясняет. И здесь тоже есть странность — ёмкость экологической ниши (место) получилась равной 50 млрд челет. При такой ёмкости K-модель достаточно точно отражает демографический процесс вплоть до начала экопаузы в 1975 г.

4. K-модели роста населения Земли в эпоху экодефицита

Роль биоты в K-модели на рис. 5 сведена к «растворителю» (М), в который погружены люди (Ч) и техносфера (М). Для моделирования функций биоты требуется осознать эмпирический смысл величин P и C, используемых в уравнениях (3)–(7) без

особых пояснений. Исходя из размерности C [челет], ясно, что ёмкость экологической ниши P — это количество человек, которое может обеспечить техносфера без дополнительных трудовых усилий. Ёмкость P измеряется в единицах [чел], а C — труд (Т), необходимый человеку, чтобы исчерпать всю биоту (Б) и превратить её в техносферу — место (М) проживания и пропитания человека. Ясно, что всю биоту исчерпать нельзя в силу законов экологии. В норме масса продуцентов должна составлять около 98 % массы биоты, а общая масса консументов и редуцентов должна быть около 2 % массы биоты. Тогда продуценты могут без ущерба прокормить и консументов, и редуцентов. Человек, как вид, является консументом и редуцентом одновременно, и мы вправе ожидать, что в эпоху экодефицита число экологических мест (М) и соответственно людей (Ч) существенно не превысит 2 % от мощности биоты. И действительно, $C \cong 200$ млрд челет, а годовой биологический ущерб от человечества в конце экодефицита составляет $N_{1975} \cong 4$ млрд челет, т. е. $\cong 2\,\%$. После 1975 г. деятельность человечества становится антиэкологичной и ведёт к экологической катастрофе, а уравнения (3)–(7) постепенно теряют свою адекватность. К моменту «обострения» в 2025 г. К-модель и уравнения (3)–(7) вообще теряют эмпирический смысл.

Моделировать демографический процесс будем по-прежнему с 0-го года н. э. Начальные условия: $\text{Ч}_0 = 0{,}1$ млрд; $\text{М}_0 = 0{,}1$ млрд; $\text{Б}_0 = 196{,}805$; $\text{Т}_0 = 0$. Здесь появляется новое, чисто человеческое состояние моделирующих автоматов — труд (Т), которое дало название К-модели. Отсюда следует, что $C = \text{Ч}_0 + \text{Б}_0 + \text{М}_0 + \text{Т}_0 = 197{,}005$ млрд челет.

К-модель на рис. 6 и сеть на рис. 7 предполагает, что трудозатраты человечества (Т) пропорциональны и числу людей (Ч), и числу мест (М), причём и люди, и места растворены в биоте мощности C (переход 1). Это относится и к числу вновь созданных мест, и к числу выживших людей, причём численность мест проживания людей не убывает ни от труда, ни от размножения (тип переходов d_1 и d_3 сохраняющий — 4). Размножение людей пропорционально и количеству мест, и количеству людей, растворённых во всём множестве автоматов. А вот приложение труда к биоте, происходящее в том же растворе, приводит к превращению и этих трудозатрат, и обработанной биоты в место техносферы (тип перехода d_2 линейный удаляющий — 6). Задержка на 23 года для размножения людей подобрана так, чтобы обеспечить хорошее совпадение с известными данными палеодемографии и демографии. Совпадение получилось очень точное.

К-сеть демографического процесса и её матричное представление изображены на рис. 7. В вершинах-переходах записаны интенсивности, т. е. среднее число переходов за единицу времени, равную одному году. В вершинах-позициях записаны переменные маркеры, т. е. число автоматов, находящихся в соответствующем состоянии. Справа на рис. 7 показаны матрицы $\boldsymbol{In}$, $\boldsymbol{Out}$, $\boldsymbol{D}$ и вектор-столбец $\boldsymbol{R}$ матричного описания К-сети.

Выполняя матричное умножение, согласно (9), можно получить дифференциальные уравнения для К-модели с нормировкой (10). Для всей области адекватного поведения К-модели $\text{Т} < \text{Б}$, т. е. $\min\{\text{Т}, \text{Б}\} = \text{Т}$, вследствие чего дифференциальные уравнения имеют вид

$$\begin{aligned} d\text{Ч}/dt &= \text{Ч} \cdot \text{М}/C, \\ d\text{М}/dt &= \text{Т}, \\ d\text{Б}/dt &= -\text{Т}, \\ d\text{Т}/dt &= \text{Ч} \cdot \text{М}/C - \text{Т}, \\ \text{Ч} + \text{М} + \text{Б} + \text{Т} &= C = 197{,}005. \end{aligned}$$

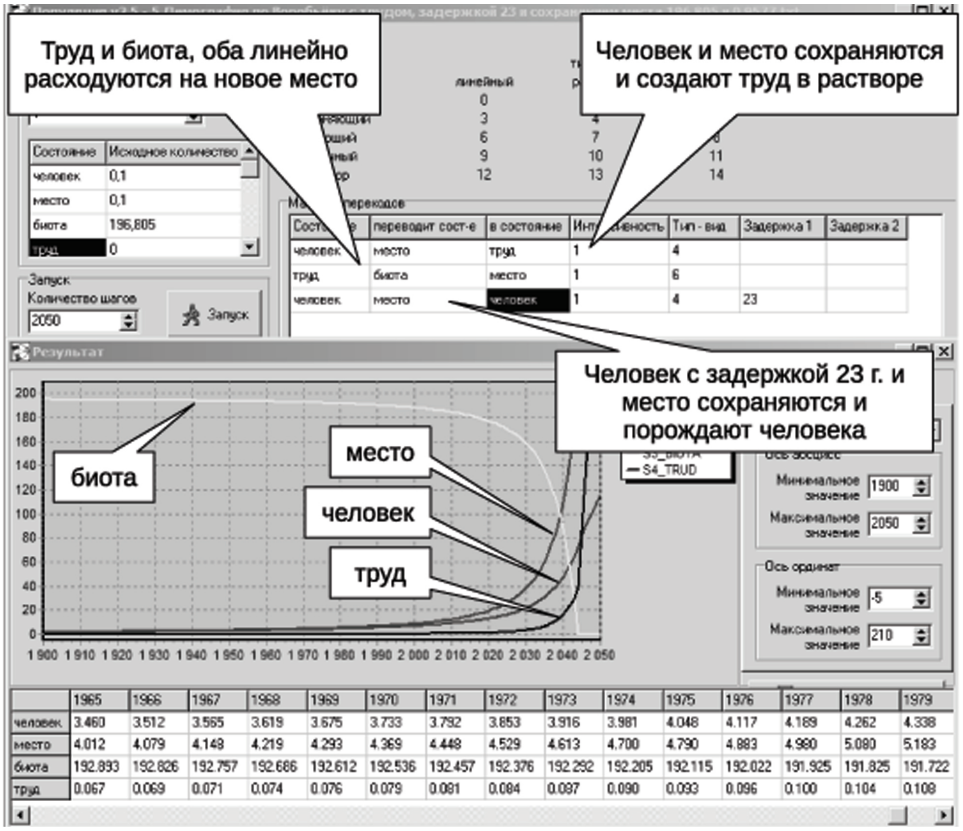
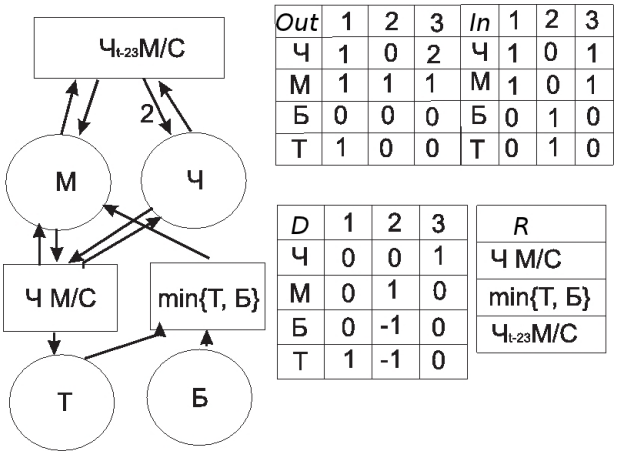


Рис. 6. Демографический процесс перехода в экопаузу. Трудовая K-модель



Out	1	2	3
Ч	1	0	2
М	1	1	1
Б	0	0	0
Т	1	0	0

In	1	2	3
Ч	1	0	1
М	1	0	1
Б	0	1	0
Т	0	1	0

D	1	2	3
Ч	0	0	1
М	0	1	0
Б	0	-1	0
Т	1	-1	0

R
Ч М/С
min{Т, Б}
Чt-23М/С

Рис. 7. K-сеть и матричное представление демографического процесса в биосфере

Впрочем, имеющаяся программа реализации K-моделей позволяет обойтись без составления и решения этих уравнений. Результаты моделирования приведены на рис. 6 для 1900–2050 г., что позволяет увидеть процесс потери адекватности K-модели.

Обратим внимание, что и эта K-модель не лишена странностей. Интенсивность рождения нового человека пропорциональна численности людей (Ч) с коэффициентом 1. Эта странность может быть объяснена тем, что делитель C в формулах (3)–(5) приравнен к ёмкости биоты. Для моделирования это допустимо, но на самом деле это не так. Ёмкость биоты должна оцениваться отдельно. Более того, нами построено

множество K-моделей демографического взрыва и экологической катастрофы, которые абстрагируются от каких-то аспектов реального демографического процесса: труда, задержек, смертности и т. п., но при этом учитывают иные аспекты реальности. Подробно рассматривать эти модели нет смысла в силу ограничения их адекватности 1975 годом, когда кончается экодефицит и начинается экопауза.

На рис. 8 показана K-модель демографического процесса в Эдеме — раю, где можно не трудиться. Дети (Д) заселяют готовое множество мест $M = 0{,}25C = 50$ млрд челет для пар родителей (Ч) в техносфере. Заселение Эдема происходит без наращивания техносферы. Демографический взрыв обеспечивается близкодействием и нелинейностью рождения детей в растворе М. K-модель показывает, что такой сценарий полностью исчерпывает техницированную биоту к 2030 г. Эта модель не отражает действительный демографический процесс, но показывает, что гиперболический рост населения возможен и в Эдеме.

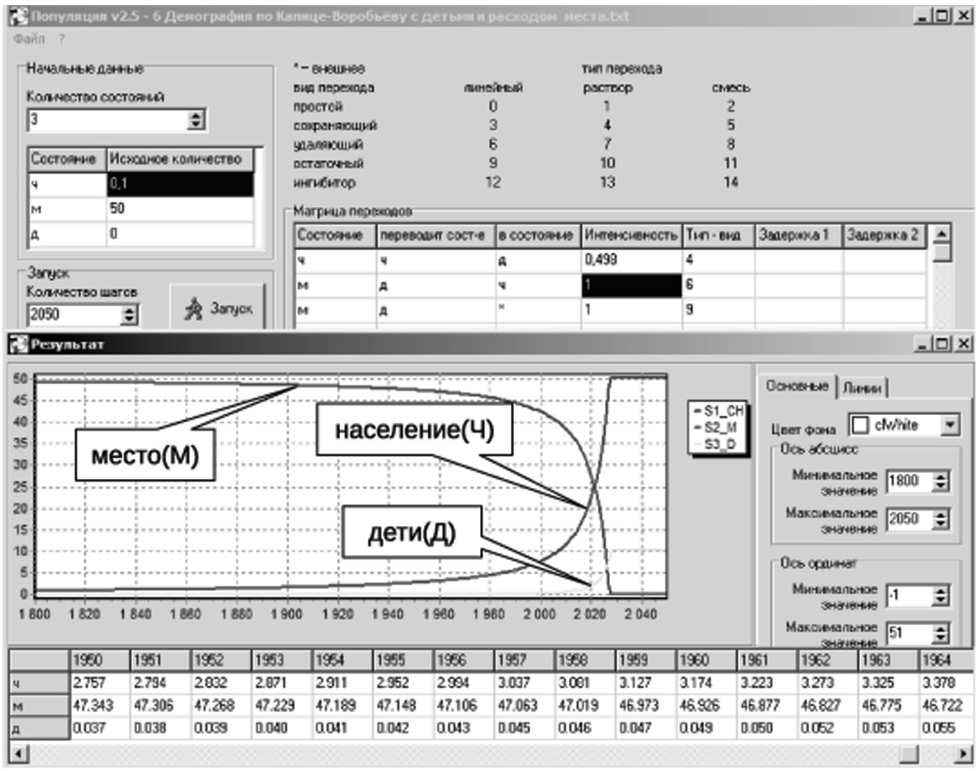


Рис. 8. Эдем. Готовая ниша заселяется людьми без труда. Дети, не нашедшие места, вымирают

5. K-модель демографического кризиса в экопаузе

С началом экологической паузы K-модели демографического взрыва постепенно теряют и адекватность, и эмпирический смысл, поскольку модельный рост населения в них ничем не ограничивается. Для того чтобы получить более адекватный результат, подобный тому, что показан на рис. 3, следует моделировать рост населения с учетом господствующей в обществе мотивации репродуктивного поведения. Разумеется, здесь тоже действуют общеизвестные естественные факторы: рождение детей (Д) из внешнего состояния (*); элиминация детей, не нашедших места, во внешнюю среду (*); смертность всех людей с освобождением места (М); рост ёмкости экологической ниши (М) в результате творческих усилий людей-творцов (Т). В индустриальном и постиндустриальном обществе добавляются ещё два фактора, замещающих репродуктивное

поведение: во-первых, обучение молодёжи и появление творцов (Т), во-вторых, появление избыточного места (М) и потребителей (П). Будем полагать, что творчество и потребление исключают размножение, но не исключают смертность. Основанием для такого суждения служат демографические данные, показанные на рис. 2 и 3. Ясно, что каждый человек может попеременно выполнять все три функции: репродуктивную, творческую и потребительскую, поэтому соответствующие численности Ч, Т и П относятся к людям, которые заняты данной функцией в текущий момент времени.

Демографическая статистика позволяет задать вполне правдоподобные коэффициенты рождаемости и смертности людей. Естественная рождаемость — 47,7 детей на 1000 человек населения. В нашей К-модели вся смертность сводится к детской смертности, а выжившие дети, став взрослыми людьми, живут до 77 лет, творцы — до 71 года. Это упрощение не слишком искажает результат. Средняя продолжительность жизни соответствует реальной. Неизвестными остаются только коэффициенты появления творческих людей и потребителей. Эти коэффициенты приходится подбирать, добиваясь наилучшего ретроспективного соответствия модельной численности населения и известных демографических данных. Вообще, проблема подбора неизвестных коэффициентов модели является отдельной задачей оптимального синтеза К-модели.

Варьируя коэффициенты К-модели, ответственные за появление творцов и потребителей, можно получить различные значения максимального населения Земли и сроки его достижения. Ближайшая к наблюдаемому [7, 10] мировому демографическому процессу К-модель с 1800 по 2500 г. показана на рис. 9.

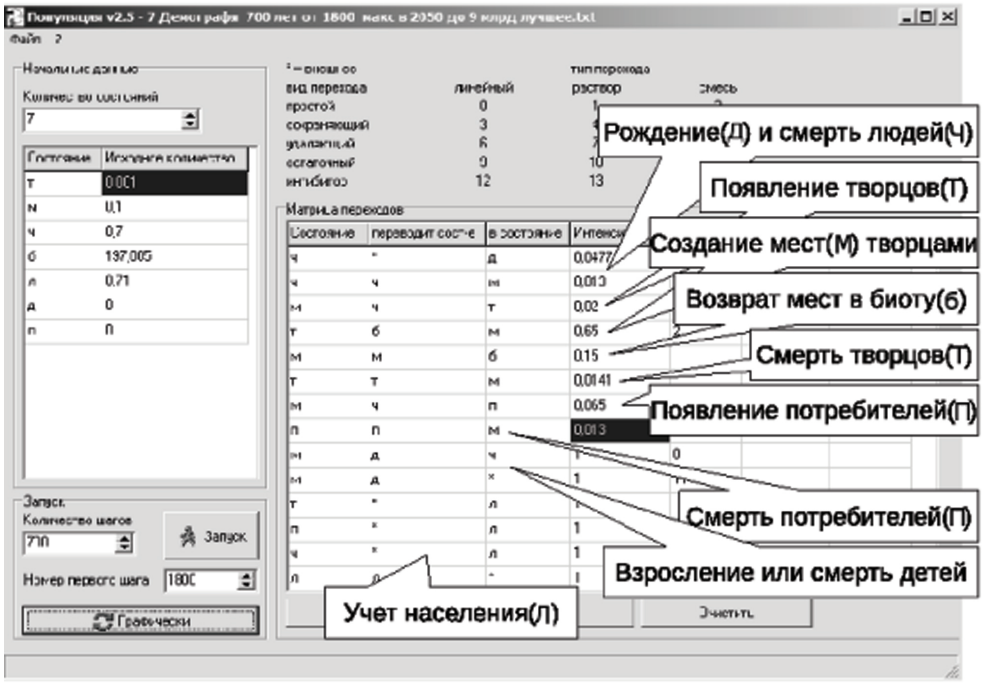
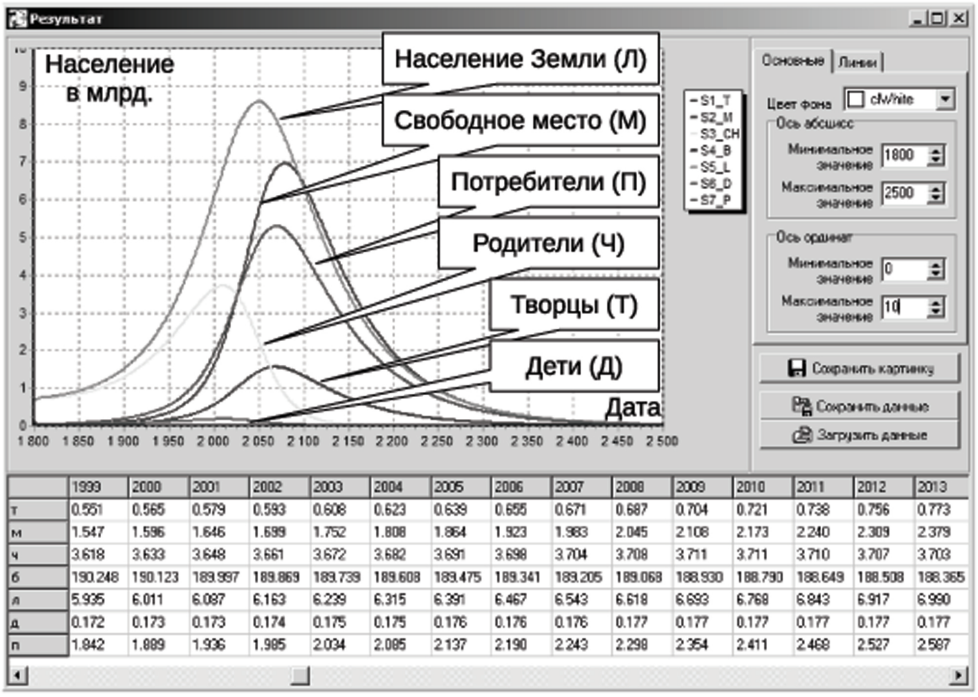


Рис. 9. К-модель демографического кризиса эпохи модерна с 1800 по 2500 г. с потерей адекватности в экологической паузе после 1975 г.

Результаты К-моделирования, представленные на рис. 10, показали, что если репродуктивное поведение людей не изменится, постиндустриальная цивилизация вымрет.



	1999	2000	2001	2002	2003	2004	2005	2006	2007	2008	2009	2010	2011	2012	2013
т	0.551	0.565	0.579	0.593	0.608	0.623	0.639	0.655	0.671	0.687	0.704	0.721	0.738	0.756	0.773
м	1.547	1.596	1.646	1.699	1.752	1.808	1.864	1.923	1.983	2.045	2.108	2.173	2.240	2.309	2.379
ч	3.618	3.633	3.648	3.661	3.672	3.682	3.691	3.698	3.704	3.708	3.711	3.711	3.710	3.707	3.703
б	190.248	190.123	189.997	189.869	189.739	189.608	189.475	189.341	189.205	189.068	188.930	188.790	188.649	188.508	188.365
л	5.935	6.011	6.087	6.163	6.239	6.315	6.391	6.467	6.543	6.618	6.693	6.768	6.843	6.917	6.990
д	0.172	0.173	0.173	0.174	0.175	0.175	0.176	0.176	0.176	0.177	0.177	0.177	0.177	0.177	0.177
п	1.842	1.889	1.936	1.985	2.034	2.085	2.137	2.190	2.243	2.298	2.354	2.411	2.469	2.527	2.587

Рис. 10. Демографический процесс эпохи модерна и постмодерна с 1800 по 2500 г. В строке **Л** дана модельная численность населения Земли, очень хорошо совпадающая с данными всемирного банка [7] и статистическим прогнозом (рис. 3)

6. К-модель этногенеза

Демографический процесс отражает человеческий способ приспособления к природной среде — расширение своей экологической ниши, т. е. техносферы.

Этногенез — продукт внутривидовой конкуренции за экологическую нишу. Ближайшей популяционной моделью такой конкуренции является модель «хищник — жертва».

Подробное описание этногенеза дано в работах Л. Н. Гумилёва [13]. Согласно Гумилёву, в социальной жизни происходит конкуренция трёх *социотипов* человека: *пассионариев* (*пасси*), *гармоничников* (*гарми*) и *суббиологичных* субъектов (*субби*). При этом пасси (П) вытесняют субби (С) как непосредственными притеснениями и убийствами, так и организацией военных и криминальных затей, где субби — расходный материал. Гарми (Г), в свою очередь, вытесняют пасси, как неудобных и асоциальных субъектов, плохих семьянинов и наиболее жертвенных воинов и диссидентов. И, наконец, субби живут за счёт сердобольных гарми, а зачастую просто грабят и убивают своих кормильцев.

Пасси стремятся к «иррациональным» ценностям духа — истине, красоте, справедливости, чести, социальному лидерству. При этом они тормозят (подавляют) свои биологические инстинкты вплоть до *жертвенности*. Это, в конечном счёте, и позволяет человечеству расширять свою экологическую нишу, творить техносферу. Гарми не против рационального труда, но они не приходят в противоречие с биологическими позывами души и тела. Субби живут только ради удовлетворения плотских биологических потребностей: еды, секса, сна, агрессии, лидирующих позиций в сообществе. Для этого субби извлекают из гарми *избыточность* (И) — те материальные блага, которые

гармы могут пожертвовать или лишиться насильственно без очевидного ущерба для своей численности. А поскольку избыточность создают пасси, то такое положение дел сохраняется, пока есть эти самые пасси. Как только пасси практически уничтожены, эксплуатация гармоничного сообщества со стороны субби приобретает хищнический характер, что и приводит к обскурации, депопуляции и гибели этноса.

У этих трёх социотипов различно и экологическое поведение. Пасси обеспечивают рост техносферы, гармы — хранители техносферы, а субби — её разрушители. Если пасси и гармы оставляют потомкам благоустроенное место жительства (М), то субби оставляют после себя даже не биоту, а пустыню, которая постепенно возвращается в биоту.

Л. Н. Гумилёв считает гармы основным социотипом человека, появление пасси он объясняет генетическим процессом мутации непонятного (космического?) происхождения, а субби — исчезновением гена пассионарности. Эта теория похожа на объяснение альтруизма и эгоизма сочетанием рецессивного гена альтруизма с доминантным геном эгоизма подобно известной теории пола как сочетания X и Y хромосом. Однако всё не так просто. Л. Н. Гумилёв полагает, что пасси, гармы и субби — это типы социального поведения, которые определяются не только наследственными признаками, но и *индукцией* социальных установок людей. В нашей модели, представленной на рис. 11, пасси, гармы и субби являются таковыми от рождения, по наследству от родителей. Индукция, пассионарная или субпассионарная, не моделируется.

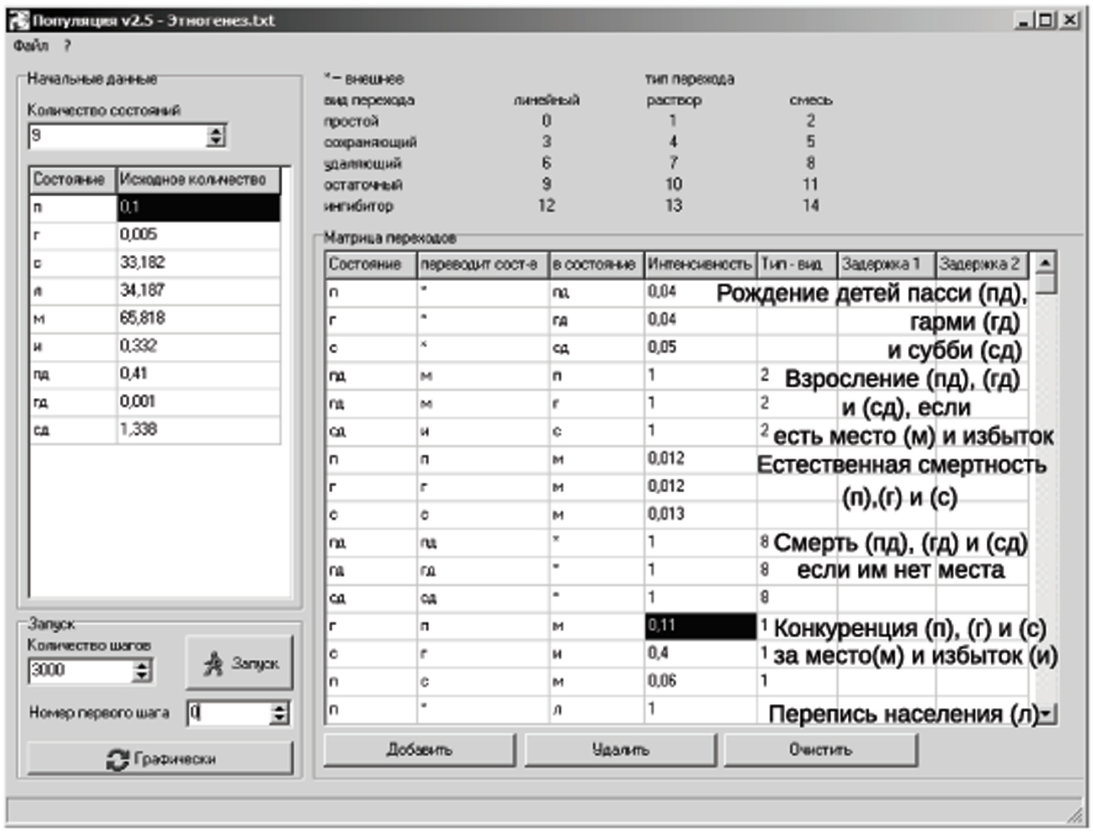


Рис. 11. К-модель этногенеза как конкуренции пасси (П), гармы (Г) и субби (С)

Как и в модели «хищник — жертва», в модели этногенеза наблюдается чередование фаз с разными доминантами поведения. На рис. 12 можно видеть *волны* преобладания

пасси, гармы и субби. В фазах *подъёма* и *акме* этнос формирует и активно распространяет свою культуру, религию и стереотипы поведения, расширяет свою территорию, воюет с соперниками. Этому соответствует пассионарная доминанта поведения людей. В фазах *надлома* и *инерции* происходит переход к универсальной светской государственности (собственно в этом и состоит надлом) и расцвет цивилизации, поскольку растёт исполнительская дисциплина и основное внимание окрепшее государство уделяет благоустройству жизни. *Инерция* — это эпоха гармоничной доминанты поведения. Наконец, в фазах *обскурации* и *депопуляции* полностью исчезают пасси, затем вымирают гармы, а субби, оставшиеся без контроля от пасси и покровительства от гармы, быстро разрушают свой кормящий ландшафт и инфраструктуру техносферы. В результате субби-доминанта приводит к разрушению цивилизации, исчезновению этноса, депопуляции и *реликтовому* состоянию, т. е. к равновесию с окружающей средой, как это обычно бывает в биологической популяции. Такой этнос обычно становится жертвой *нашествия варваров* [14].

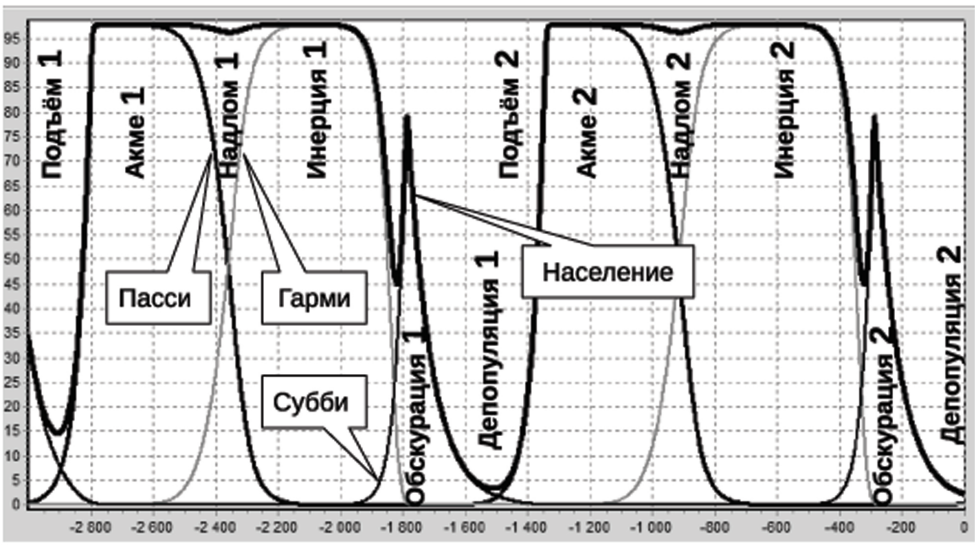


Рис. 12. Два классических цикла этногенеза в Древнем Египте

На рис. 12 показаны два классических цикла этногенеза на территории Египта. Для того чтобы читатель мог увидеть фазы этногенеза содержательно, в табл. 2 приводятся знаковые события истории Древнего Египта за период 3000–0 г. до н.э. При этом коэффициенты строк 13, 14 и 15 в графе «интенсивность», ответственные за конкуренцию, подобраны так, что совпадение основных решающих исторических событий с фазами этногенеза просто удивительное. Кроме того, видно, что длительной фазы реликта в Древнем Египте не было. Остатки пасси и нашествие пассионарных варваров (по Тойнби [14]) восстановили пассионарность населения Египта. В отсутствие гармы пасси стали быстро размножаться, эксплуатировать, истреблять и расходовать субби в новых войнах. При этом и характер исторических событий стал не таким, как в Древнем и Среднем Царствах. После римской аннексии Египта самостоятельный этногенез там прекратился.

На рис. 13 эти же волны этногенеза наложены на историю Западной Европы от основания Рима до Европейского Союза. В данном случае известно, что второй цикл европейского этногенеза происходит с другим расовым и этническим составом населения Европы в результате вторжения варваров с Севера. Древний Рим — это италики средиземноморской расы, а Северная Европа — скандинавы, франки и германцы

Таблица 2

Хронология Древнего Египта. Знаковые события

Годы до н. э.	Этапы	Основные события
3000–2700	**Подъём**, 300 л.	3000 Объединение Египта. Древнее царство
2700–2400	**Акме**, 300 л. Власть религии	2700–2400 Джосер, Хеопс и др. Пирамиды 2640 Имхотеп. Первые своды наставлений 2575 Снофру. Новая техника строительства пирамид
2400–2150	**Надлом**, 250 л. Политический кризис	2465 Реформы. Усиление власти на местах 2325 Усиление власти номархов 2200 Плавильное дело, использование бронзы 2155 Распад Древнего царства 2154 Начало первого переходного периода в Египте
2150–1800	**Инерция**, 350 л. Империя. Борьба за расширение и укрепление империи	2040 Среднее царство. Объединение Египта 2000 Планиметрия и пространственные отношения 1962 Хети написал поучения Аменемхета I 1926 Появилась История Синухета 1890–1800 Первые математические папирусы 1878 Сенусерт III. Расширение границ Египта 1841 Аменемхет III. Регулирование уровня воды
1800–1700	**Обскурация**, 100 л.	1785 Распад Среднего царства
1700–1551	**Депопуляция**, 150 л.	1650 Начало второго переходного периода в Египте
1551–1300	**Подъём**, 251 л. Восстановление империи и начало её нового расширения. Кризис религии. Победа старого культа и укрепление государства	1551 Яхмос I. Победа над гиксосами, XVIII династия 1505 Тутмос I. Начало экспансии Египта 1500 Составлен «папирус Эберса» 1490 Хатшепсут. Государственный переворот 1461 Тутмос III. Экспансия Египта 1403 Тутмос IV. Мир с государством Митанни 1361 Аменхотеп IV. Культ Эхнатона и новая столица 1347 Тутанхамон. Отмена культа Эхнатона. Перенос столицы в Мемфис
1300–1100	**Акме**, 200 л. Экспансия Египта. Политические интриги	1286 Рамсес II. Битва при Кадеше 1270 Рамсес II и Хаттусили III. Соглашение о Сирии 1184 Рамсес III. Разгром ливийцев и народов моря 1153 Убийство фараона Рамсеса III
1100–900	**Надлом**, 200 л.	1070 Начало третьего переходного периода в Египте 945 Шешонк I основал XXII династию
900–350	**Инерция**, 550 л. Внешние вторжения и борьба за сохранение единой империи	728 Пианхи. Начало правления нубийской династии 681 Асархаддон. Экспансия Персии в Египет 664 Псамметих I. Объединение Египта 589 Борьба за Палестину 404 Амиртей изгнал персов 380 Неферит II. Последняя самостоятельная династия
350–200	**Обскурация**, 150 л. Недееспособность	343 Артаксеркс III. Завоевание Египта персами 332 Александр Македонский. Александрия
200–000	**Депопуляция**, 200 л.	47 Частично гибнет Александрийская библиотека 30 Клеопатра. Аннексия Египта Римом

центрально- и североевропейских рас. Основные события истории Западной Европы хорошо известны и наложены прямо на волны европейских этногенезов. Совпадение и здесь очень точное.

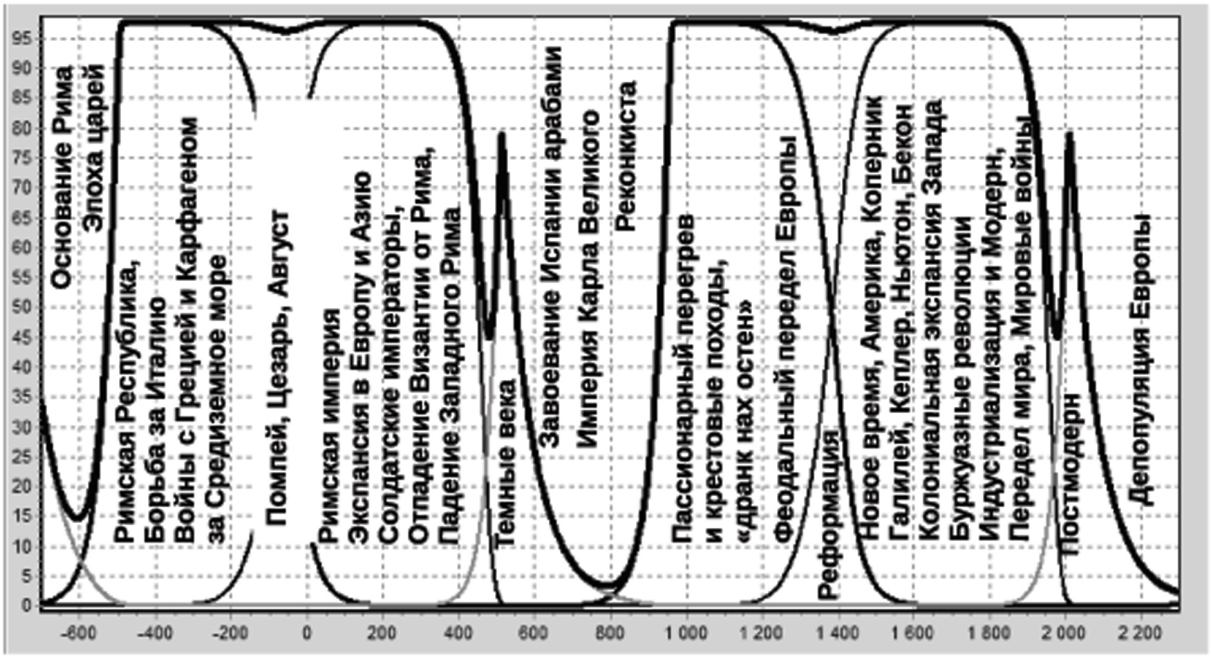


Рис. 13. Этногенез в Западной Европе от Рима до Европейского Союза. Та же последовательность этапов этногенеза, что и в Египте (рис. 12)

Та же кривая, наложенная на хронологию христианской России, тоже даёт точное совпадение событий с фазами этногенеза. Результат показан на рис. 14. В данном случае этнический состав населения также изменился по сравнению со славянами первого тысячелетия нашей эры. Произошла метисация славян, убегавших от печенегов и половцев, с угрофинскими и тюркскими народами Севера России и Поволжья, что и породило великорусский суперэтнос — московитов, поморов, волгарей и т. д.

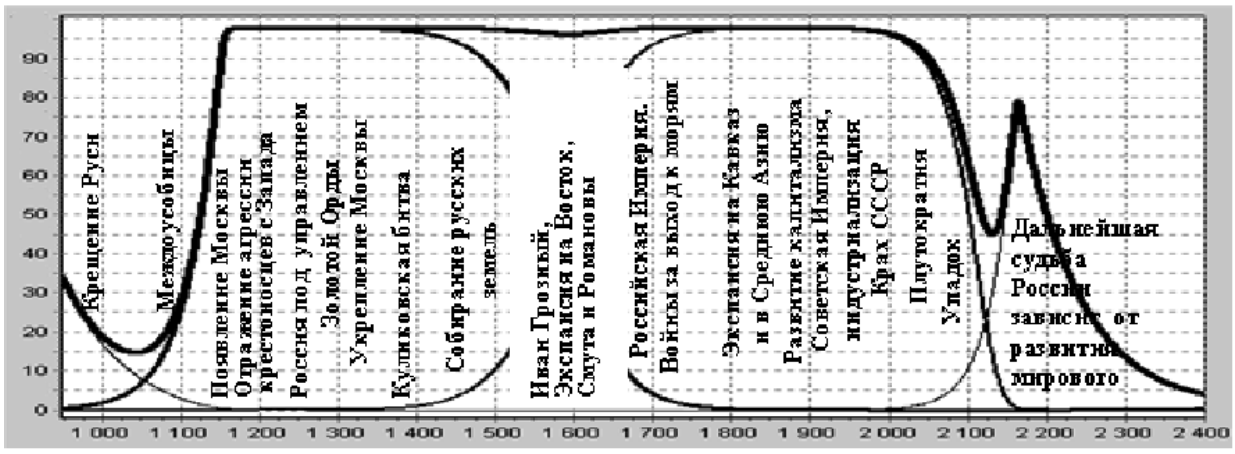


Рис. 14. Этногенез христианской России с 950 по 2400 г.

.

Рис. 14 очевидно противоречит мнению Л. Н. Гумилёва и А. Тойнби о том, что русский этногенез начался в XIV веке одновременно с Польшей и Турцией. Дело в том,

что яркие события в этногенезе происходят с интервалом приблизительно в 350 лет, при смене фазы этногенеза. Поэтому сдвиг начала этногенеза на этот срок также даёт правдоподобную картину. Если верить Гумилёву, то следует считать за начало фазы подъёма в России 1250 г. — сразу после татаро-монгольского нашествия. Акме — от Куликовской битвы до Отечественной войны 1812 г., надлом начинается выступлением декабристов в 1825 г. и заканчивается распадом СССР и 2000 годом. Эта картина российского этногенеза по Л. Н. Гумилёву показана на рис. 15.

Рис. 15. Этногенез в России по Л. Н. Гумилёву с 1250 по 2700 г.

Какая из моделей адекватна? Модель на рис. 14 хорошо согласуется с представлениями многих представителей Русской Православной Церкви (РПЦ), с предшествующими результатами, с наблюдениями Н. Я. Данилевского [15], да и с современным состоянием дел в России. Надлом превращает Этнос в Цивилизацию и Империю, после чего следует внешняя колониальная экспансия, что мы и видим на рис. 14. В современной России мы наблюдаем депопуляцию, связанную либо с обскурацией, либо с мировым кризисом цивилизации. Модель Гумилёва тоже хорошо описывает события последних веков.

К сожалению, исторического материала, позволяющего сделать выбор между моделями РПЦ и Гумилёва, у авторов пока нет. Необходимо хорошее описание истории России за 1500 лет до крещения Руси. Тогда можно было бы увидеть два цикла этногенеза (как в Египте и Европе) и верифицировать K-модель. Кроме того, судя по результатам моделирования демографического кризиса в экопаузе, представленные модели этногенеза теряют адекватность уже в XXI веке. История человечества приобретает новый характер и новый смысл.

Обратим внимание, что максимальная численность населения везде равна 100. Дело в том, что наша K-модель не учитывает расширения техносферы, а её ёмкость — это число мест (M). В силу теоремы масштабируемости [12] можно взять любой масштаб для задания начального состояния и подобрать необходимую численность населения. В данном случае выбран такой масштаб, что число людей с разными типами пове-

дения совпадает с процентом этих людей. Адекватная каузальная модель этногенеза с расширением техносферы не является предметом данной работы. Важен принципиальный качественный результат.

Ещё одна деталь К-модели этногенеза — депопуляция этноса почти до нулевой численности. В реальности пассионарное и гармоничное население не вымирает, а *замещается* или *отпадает* от этноса подобно отпадению христиан от язычников, Византии от Рима, великороссов от Киевской Руси, староверов от несториан и т. д. *Раскол этнического поля*, бегство из этноса (территориальное и/или социальное) сохраняет пассионарный генофонд для последующего *пассионарного толчка* на периферии этноса.

7. Результат исследования

Итак, разработан удобный метод быстрого получения и проверки математических моделей исторического процесса — К-моделирование. Построенные модели исходят из предположения о благополучном развитии земной цивилизации без катастроф и мировых войн. Пока получены только предварительные результаты математического моделирования истории. К сожалению, эти результаты не могут порадовать читателей оптимистическими прогнозами. Человечество вступило в эпоху глобального кризиса. Этот кризис сильнее всего затрагивает продвинутые технические цивилизации — Запад и Россию, но за ними неизбежно последуют и развивающиеся страны. Из исторических наблюдений, математических и каузальных моделей демографии и этногенеза вытекают следующие предварительные выводы.

1. Современное человечество прошло три крупных этапа антропогенеза: 1) эпоха расселения по планете Земля [5]; 2) эпоха экологического дефицита и этногенеза, как результата конкуренции людей за экологическую нишу на Земле [13]; 3) эпоха экологической паузы и мирового экологического кризиса [1 – 3]. Экопауза завершается в XXI веке, после чего начинается вымирание постиндустриальной цивилизации и, возможно, замена современного *homo sapiens* новым видом разумных людей.

2. Репродуктивное поведение человека, как биологического вида, мотивируется сексуальной потребностью. Появление потомства при этом является не целью, а побочным результатом сексуальности. Если репродуктивное поведение людей не изменится, постиндустриальная потребительская цивилизация вымрет. Возможно, что это обусловлено не только социокультурными, но и генетическими факторами, сформированными в эпоху экологического дефицита. Современные люди вымирают в условиях экологической паузы, экологического кризиса и изобилия.

3. Исторический процесс в эпоху экодефицита детерминирован, в основном, этногенезом — объединением людей, родственных генетически, психологически и социокультурно, в борьбе за экологическую нишу на Земле [13].

4. Конкуренция происходит не только между этносами, но и внутри этносов, как минимум, между тремя генотипами людей: пасси, гарми и субби. Ближайшей математической моделью этой конкуренции является модель «хищник — жертва».

5. Основные исторические повороты происходят при смене этапов этногенеза, когда меняется доминанта социального поведения. С небольшими вариациями коэффициентов это наблюдается на моделях этногенеза и в Египте, и в Европе, и в России, и в Древней Греции, и в Византии. Это, очевидно, общая закономерность этногенеза.

6. Хорошее совпадение волн этногенеза в Египте, Западной Европе, России, Древней Греции и Византии (это одна и та же кривая) говорит о том, что этногенез практически не зависит от климата, расы, культуры, календарного времени и уровня техно-

логии. Это значит, что этногенез, скорее всего, не социокультурное и не экономическое, а природное явление, связанное с генетикой человека как биологического вида.

7. Для появления носителей пассионарной и иной генетики нет необходимости в каком-то космическом вмешательстве. Волны этногенеза возникают, как и в модели «хищник — жертва», из-за нелинейности процессов конкуренции людей.

8. Исторический процесс объективен, закономерен и практически не зависит от желаний и идей отдельных людей или социальных групп. Никакие социальные, просветительские или воспитательные меры не могут изменить ход истории. Люди могут предлагать самые разумные и прогрессивные идеи или социальные институты, но тщетно. Всякая идея будет продуктивна тогда и только тогда, когда для её восприятия и реализации появится достаточное число генетически подходящих людей — пасси, гарми или даже субби. И в этом трагедия гениев и пророков.

9. На волне *подъёма* и *акме* доминирует самая жёсткая религиозная социокультурная установка, которая невыносима для возрастающей массы гарми, а потом и субби. В результате происходит *надлом*: смута, гражданская война или революция, смягчение нравов в пользу витальных инстинктов гарми и субби, установление социального равновесия и законности, а точнее — безразличия к высшим сакральным ценностям. Этот результат развития пассионарной культуры есть, собственно, *цивилизация*, как и утверждал О. Шпенглер [16].

10. В своём развитии цивилизация становится всё более агрессивной по отношению к культуре, породившей эту цивилизацию. Это приводит к *обскурации* — разгулу субби, отрицанию и осмеянию пассионарной *классической культуры*, её вытеснению массовой *поп-культурой*. После этого этнос, потерявший пассионарность, базовые ценности и мотивы для продуктивной деятельности, гибнет. Начинается *депопуляция*, беспокоящая деятелей культуры — так называемых *интеллектуалов*.

11. Интеллектуалы не знают ни фундаментальных законов истории, ни математики. Поэтому они не в состоянии понять происходящие процессы и горячо обсуждают разные *благоглупости* (Салтыков-Щедрин), окончательно разлагающие гибнущий этнос.

12. Теория этногенеза по Л. Н. Гумилёву в целом выдержала проверку математическим моделированием. Качественная картина этногенеза хорошо подтверждается. Однако если сменить некоторые коэффициенты в К-модели, можно получать различные сценарии этногенеза с пролонгированной фазой реликта, с различными длительностями волн этногенеза, с равновесными реликтовыми состояниями даже без тех или иных социотипов человека. Всё это, впрочем, получается так же, как и на более простой модели «хищник — жертва».

13. Живая и богатая событиями История получается в узком коридоре параметров, в которые человечество попадает не так уж часто. Недаром Н. Я. Данилевский [15], О. Шпенглер [16] и А. Тойнби [14] насчитывают в истории едва ли два десятка заметных культур и цивилизаций.

Заключение

Жизненно необходимо математическое и компьютерное моделирование исторических процессов современности. Численные результаты демографии и исторической статистики не дают полного понимания мотивов людей и, следовательно, самой истории. Для получения моделей, достаточно точных для научного исторического прогнозирования, метод К-моделирования должен быть развит и реализован полностью.

Но что особенно важно, так это понимание социального поведения людей, их мотивов и деяний. Именно мотивы и деяния легли в основу дискретных К-моделей, представленных выше: рождение детей и их вымирание от недостаточной производительности техносферы, творчество и аскеза творцов, потребительство и гедонизм, как основа снижения рождаемости и творческой активности, одичание брошенного ландшафта. Всё это лежит в основе К-моделей и демографического процесса, и этногенеза. Выяснение и описание гуманитарных мотивов— дело профессиональных психологов, антропологов и историков. А вычисления и прогнозы обеспечиваются математической историей.

ЛИТЕРАТУРА

1. *Воробьев В. А., Воробьева Т. В.* Экологический императив и демографический процесс // Вестник Поморского университета. Сер. Естественные и точные науки. 2003. №1(3) С. 122–131.
2. *Воробьев В. А., Воробьева Т. В.* Демографический парадокс, экология и религия // Свеча-2003: Наука и Религия: сб. научных и методических работ по религиоведению и культурологии / под ред. Е. И. Аринина. Архангельск: Поморский государственный университет им. М. В. Ломоносова, 2003.
3. *Воробьев В. А., Воробьева Т. В.* Экологическая пауза — системный кризис человечества // Исследования в области глобального катастрофизма / под ред. В. К. Журавлёва. Вып. 1. Новосибирск: Редакционно-издательский центр НГУ, 2006. С. 69–109.
4. *Капица С. П., Курдюмов С. П., Малинецкий Г. Г.* Синергетика и прогнозы будущего. 2-е изд. М.: Эдиториал УРСС, 2001. 288 с.
5. *Капица С. П.* Сколько людей жило, живет и будет жить на Земле. Очерк теории роста Человечества. М.: Международная программа образования, 1999. 240 с.
6. *Малков А. С., Коротаев А. В., Халтурина Д. А.* Математическая модель роста населения Земли, экономики, технологии и образования // Новое в синергетике, новые проблемы, новое поколение / под ред. Г. Г. Малинецкого. М.: Наука, 2007. С. 148–186.
7. Народонаселение стран мира. Справочник / под ред. Б. Ц. Урланиса. 2-е изд. М.: Статистика, 1978.
8. *Базыкин А. Д.* Нелинейная динамика взаимодействующих популяций. М.; Ижевск: Институт компьютерных исследований, 2003. 368 с.
9. *Форестер Дж.* Мировая динамика. М.: Наука, 1978. 314 с.
10. Страны и регионы. Статистический справочник Всемирного банка: пер. с англ. М.: Весь мир, 1999–2005.
11. *Воробьев В. А.* Метод моделирования популяции автоматов // Современные достижения в науке и образовании: математика и информатика. Материалы междунар. научн.-практич. конф. Архангельск, 1–5 февраля 2010. С. 16–22.
12. *Воробьев В. А., Березовская Ю. В.* Популяции взаимодействующих автоматов // Прикладная дискретная математика. 2011. №4. С. 89–104.
13. *Гумилев Л. Н.* Этносфера: история людей и история природы. М.: АСТ, 2004. 575 с.
14. *Тойнби А. Дж.* Постижение истории. `http://www.hrono.ru/index.php`
15. *Данилевский Н. Я.* Россия и Европа. Взгляд на культурные и политические отношения Славянского мира к Германо-Романскому. СПб.: ГЛАГОЛЪ, СПбГУ, 1995. 501 с. `http://www.booksite.ru/fulltext/yev/rop/ada/nil/index.htm`
16. *Шпенглер О.* Закат Европы. Новосибирск: Наука, 1993. 592 с.

СВЕДЕНИЯ ОБ АВТОРАХ

БЕРЕЗОВСКАЯ Юлия Владимировна — старший преподаватель кафедры программирования и высокопроизводительных вычислений Северного (Арктического) федерального университета им. М. В. Ломоносова, г. Архангельск. E-mail: **myumla.myu@gmail.com**

БЫКОВА Валентина Владимировна — доцент, кандидат технических наук, профессор Института математики Сибирского федерального университета, г. Красноярск. E-mail: **bykvalen@mail.ru**

ВОРОБЬЕВ Владимир Анатольевич — профессор, доктор технических наук, профессор кафедры программирования и высокопроизводительных вычислений Северного (Арктического) федерального университета им. М. В. Ломоносова, г. Архангельск. E-mail: **vva100@atnet.ru**

ЖАРКОВА Анастасия Владимировна — аспирантка Саратовского государственного университета им. Н. Г. Чернышевского, г. Саратов. E-mail: **VAnastasiyaV@gmail.com**

ЗАКРЕВСКИЙ Аркадий Дмитриевич — член-корр. НАН Беларуси, доктор технических наук, главный научный сотрудник Объединённого института проблем информатики НАН Беларуси, г. Минск. E-mail: **zakrevskij@tut.by**

КАРМАНОВА Евгения Олеговна — аспирантка Саратовского государственного университета им. Н. Г. Чернышевского, г. Саратов. E-mail: **janekao@mail.ru**

КЯЖИН Сергей Николаевич — студент факультета кибернетики и информационной безопасности Национального исследовательского ядерного университета МИФИ, г. Москва.
E-mail: **litota975@mephist.ru**

РОМАНЬКОВ Виталий Анатольевич — профессор, доктор физико-математических наук, заведующий кафедрой Омского государственного университета им. Ф. М. Достоевского, г. Омск.
E-mail: **romankov48@mail.ru**

САЛИЙ Вячеслав Николаевич — кандидат физико-математических наук, заведующий кафедрой, профессор Саратовского государственного университета им. Н. Г. Чернышевского, г. Саратов.
E-mail: **SaliiVN@info.sgu.ru**

СЕМЕНОВА Наталья Андреевна — аспирантка Московского государственного института электроники и математики (ТУ), г. Москва. E-mail: **natasha_sem@inbox.ru**

СТОЛОВ Евгений Львович — профессор, доктор технических наук, профессор кафедры системного анализа Казанского федерального университета, г. Казань. E-mail: **yevgeni.stolov@ksu.ru**

ТИМЕРЯЕВ Тимофей Валерьевич — аспирант Уфимского государственного авиационного технического университета, г. Уфа. E-mail: **timeryaev@yandex.ru**

УРАКОВ Айрат Ренатович — кандидат физико-математических наук, доцент Уфимского государственного авиационного технического университета, г. Уфа. E-mail: **urakov@ufanet.ru**

ФОМИЧЕВ Владимир Михайлович — доктор физико-математических наук, профессор кафедры криптологии и дискретной математики Национального исследовательского ядерного университета МИФИ, г. Москва. E-mail: **fomichev@nm.ru**

АННОТАЦИИ СТАТЕЙ НА АНГЛИЙСКОМ ЯЗЫКЕ

Kjazhin S. N., Fomichev V. M. **ABOUT PRIMITIVE SYSTEMS OF NATURAL NUMBERS.** The set structure of primitive systems of natural numbers is described, and the main properties of such systems are installed. An algorithm for enumerating primitive systems of numbers not exceeding a given number m is constructed using the concepts of deadlockness and k-minimalities of primitive systems. Also, some algorithms are offered for determining the primitiveness index of a finite directed graph by means of depth-first search and the exponentiation of the vertex adjacency matrix. Computational complexity of the algorithms is estimated.
Keywords: *primitive system of natural numbers, primitive matrix, primitive graph, exponent, subexponent.*

Romankov V. A. **DIOPHANTINE CRYPTOGRAPHY OVER INFINITE GROUPS.** Here, a short survey is presented on the group-based cryptography that is a contemporary area of an activity which explores how abstract infinite groups can be used as platforms for constructing cryptographic primitives, systems and protocols. Studying in the area is developed by the methods of group theory, complexity theory and theory of computations. Undecidable and allegebly hard algorithmic problems from group theory are the base of the constructing. Different complexity aspects of the algorithmic problems and the corresponding search problems are discussed. The universality of the diophantine language in cryptography is explained, and its combining role is noted. The free metabelian groups as platforms for constructing cryptographic systems and protocols is suggested. Some reasons for the suggestion including a decidability of the word problem and existing normal forms of elements are stated. One more reason is a non-decidability of the equation solvability problem and non-decidability of the endomorphism problem in these groups that follow from a non-decidability of the 10-th Hilbert Problem. It is promised that the considerations of the paper will be used in the forthcoming paper by the author and S. Y. Erofeev for constructing a possibly one-way function and an autentification protocol with zero knowledge on a free metabelian group.
Keywords: *group-based cryptography, algorithmic problem, search problem, generic complexity, diophantine language, diophantine cryptography, free metabelian group, endomorphism problem.*

Stolov E. L. **MATHEMATICAL MODEL OF RANDOM GENERATOR BASED ON TERNARY LOGIC.** A mathematical model of physical generator that is based on ternary logic is suggested. A few combinational units form a ring circuit. Each unit realizes the same ternary logical function, and the circuit works as a chaotic generator. It is proved that signal on the output of any unit has uniform distribution.
Keywords: *asynchronous generator,ternary logic, Markov process.*

Semenova N. A. **SEMANTIC ROLE BASED ACCESS CONTROL MODEL.** This paper describes extension of the standard RBAC model with a semantic context that can be used in corporative information systems to express the dependence between employees' work responsibilities and system roles assigned to corresponding users. The main difference of the proposed model is that it allows to automate role assignment and revocation. This

article contains also a safety proof for proposed semantic model.
Keywords: *RBAC, automated role management.*

Bykova V. V. **FPT-ALGORITHMS ON GRAPHS OF LIMITED TREEWIDTH.** A method for designing FPT-algorithms by means of dynamic programming based on the tree decomposition is investigated. Some problems limiting the application of this method in practice are pointed. The problem of memory is solved by using a binary tree decomposition of the separator, which reduces the theoretical and the actual size of the dynamic programming tables. The technique of tables in the language of relational algebra is described.
Keywords: *graph algorithms, tree decomposition, dynamic programming.*

Zharkova A. V. **INDICES IN DYNAMIC SYSTEM OF BINARY VECTORS ASSOCIATED WITH CYCLES ORIENTATIONS.** An algorithm is proposed for computation of indices in dynamic system of binary vectors associated with cycles orientations. Evolutionary function of the system transforms vectors according to the following rules: if both the initial component is 0 and the final one is 1 they are replaced by 1 and 0 respectively and all digrams 10 are replaced simultaneously by 01. Maximal index of the subsystem formed by vectors of a given dimension is found.
Keywords: *finite dynamic system, evolutionary function, binary vectors, index, cycles.*

Karmanova E. O. **CONGRUENCE RELATIONS OF PATHS: SOME COMBINATORIAL PROPERTIES.** A congruence relation of a path is an equivalence relation on the set of its vertices all of whose classes are independent subsets. It is proved (theorem 1) that the number of all congruence relations of a path with m edges equals to the number of all equivalence relations on a m-element set. For a given connected graph G theorem 2 determines the length of the shortest path whose quotient-graph is G.
Keywords: *path, congruence relation, equivalence relation, quotient-graph, Bell number.*

Salii V. N. **THE SYSTEM OF ABSTRACT CONNECTED SUBGRAPHS OF A LINEAR GRAPH.** A linear graph is a graph obtained from a path by some orientation of its edges. The set of all connected graphs that can be embedded in a given linear graph L is ordered by embedding relation. Conditions on L are found under which this ordered set is a lattice.
Keywords: *path, linear graph, abstract subgraph of a graph, ordered set, lattice, binary vector, duality.*

Urakov A. R., Timeryaev T. V. **USING WEIGHTED GRAPHS FEATURES FOR FAST SEARCHING THEIR PARAMETERS.** In this paper, some algorithms are presented for the fast search of center, radius and diameter of weighted graphs on all-pairs shortest path matrix, using features of real-world road networks graphs. They are compared with algorithms searching these parameters by simple pass through elements of matrix.
Keywords: *graph center, graph radius, graph diameter, all-pairs shortest path matrix, graph features, weighted graph.*

Zakrevskij A. D. **SEARCH FOR CONDITIONS OF MAXIMAL ENERGY CONSUMPTION IN LOGIC CIRCUIT.** It is known that estimation of maximal energy consumption in a logic circuit allows to secure its reliability. Getting that estimation is considerably facilitated by preliminary finding such a regime of the circuit working at which its energy consumption is maximal. A method for such finding is suggested in this paper

for the case when the regarded circuit implements a finite automaton.
Keywords: *logical circiut, maximal energy consumption, logic design.*

Vorob'ev V. A., Berezovska Yu. V. **THE MATHEMATICAL MODELS OF HISTORICAL PROCESSES.** The article deals with the causal models (C-models) for the mathematical modelling historical processes. Ways to apply the causal modelling method to the investigations of historical processes are offered. The history of Western Europe and its extrapolation in the near future is reviewed.
Keywords: *causal net, causal model, mathematical modelling historical processes.*

Журнал «Прикладная дискретная математика» включен в перечень ВАК рецензируемых российских журналов, в которых должны быть опубликованы основные результаты диссертаций, представляемых на соискание учёной степени кандидата и доктора наук, а также в перечень журналов, рекомендованных УМО в области информационной безопасности РФ в качестве учебной литературы по специальности «Компьютерная безопасность».

Журнал «Прикладная дискретная математика» распространяется по подписке; его подписной индекс 38696 в объединённом каталоге «Пресса России». Полнотекстовые электронные версии вышедших номеров журнала доступны на его сайте vestnik.tsu.ru/pdm и на Общероссийском математическом портале www.mathnet.ru. На сайте журнала можно найти также и правила подготовки рукописей статей в журнал.

Тематика публикаций журнала:

- *Теоретические основы прикладной дискретной математики*
- *Математические методы криптографии*
- *Математические методы стеганографии*
- *Математические основы компьютерной безопасности*
- *Математические основы надежности вычислительных и управляющих систем*
- *Прикладная теория кодирования*
- *Прикладная теория автоматов*
- *Прикладная теория графов*
- *Логическое проектирование дискретных автоматов*
- *Математические основы информатики и программирования*
- *Вычислительные методы в дискретной математике*
- *Дискретные модели реальных процессов*
- *Математические основы интеллектуальных систем*
- *Исторические очерки по дискретной математике и ее приложениям*