МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

ПРИКЛАДНАЯ ДИСКРЕТНАЯ МАТЕМАТИКА

Научный журнал

2019 № 44

Зарегистрирован в Федеральной службе по надзору
в сфере связи и массовых коммуникаций

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

Подписной индекс в объединённом каталоге «Пресса России» 38696

УЧРЕДИТЕЛЬ
Томский государственный университет

Адрес редакции и издателя: 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 надёжности, интеллектуальных системах.

Периодичность выхода журнала: 4 номера в год.

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

Подписано к печати 20.06.2019. Формат $60 \times 84\frac{1}{8}$. Усл. п. л. 14,8. Тираж 300 экз.
Заказ № 3847. Цена свободная. Дата выхода в свет 27.06.2019.

Отпечатано на оборудовании
Издательского Дома Томского государственного университета
634050, г. Томск, пр. Ленина, 36
Тел.: 8(3822)53-15-28, 52-98-49

СОДЕРЖАНИЕ

CONTENTS

ТЕОРЕТИЧЕСКИЕ ОСНОВЫ ПРИКЛАДНОЙ ДИСКРЕТНОЙ МАТЕМАТИКИ

УДК 519.7

СВОЙСТВА КОМПОНЕНТ НЕКОТОРЫХ КЛАССОВ ВЕКТОРНЫХ БУЛЕВЫХ ФУНКЦИЙ[1]

И. А. Панкратова

Национальный исследовательский Томский государственный университет, г. Томск, Россия

В классе обратимых векторных булевых функций от n переменных с координатными функциями, существенно зависящими от всех переменных, рассматриваются подклассы $\mathcal{K}_n$ и $\mathcal{K}'_n$. Функции этих классов получены с помощью n независимых транспозиций соответственно из тождественной подстановки и из подстановки, каждая координатная функция которой существенно зависит от одной переменной. Приводятся некоторые свойства компонент функций из этих классов.

Ключевые слова: *векторная булева функция, обратимые функции, нелинейность векторной булевой функции, компонентная алгебраическая иммунность.*

DOI 10.17223/20710410/44/1

PROPERTIES OF COMPONENTS FOR SOME CLASSES OF VECTORIAL BOOLEAN FUNCTIONS

I. A. Pankratova

National Research Tomsk State University, Tomsk, Russia

E-mail: pank@mail.tsu.ru

In the class of invertible vectorial Boolean functions in n variables with coordinate functions depending on all variables, we consider the subclasses $\mathcal{K}_n$ and $\mathcal{K}'_n$, where the functions are obtained using n independent transpositions, respectively, from the identity permutation and from the permutation with coordinate functions essentially dependent on exactly one variable. We show that, for any $F = (f_1 \dots f_n) \in \mathcal{K}_n \cup \mathcal{K}'_n$ and $i \in \{1, \dots, n\}$, the coordinate function f_i has a single linear variable, each component function vF with vector $v \in \mathbb{F}_2^n$ of a weight greater than 1 has no fictitious and linear variables , the nonlinearity N_F, the degree $\deg F$, and the component algebraic immunity $\mathrm{AI}_{\mathrm{comp}}(F)$ are 2, $n-1$, and 2 respectively.

Keywords: *vectorial Boolean functions, invertible functions, nonlinearity, component algebraic immunity.*

[1]Работа поддержана грантом РФФИ, проект № 17-01-00354.

Введение

Векторные булевы функции широко используются при построении симметричных и асимметричных криптосистем [1 – 3]. К этим функциям предъявляются различные требования: они должны быть обратимы, обладать хорошими криптографическими свойствами, допускать компактное представление, быстро вычисляться. Один из способов удовлетворить последним двум требованиям — ограничение на число существенных переменных координатных функций. В [1, 4] описаны методы получения обратимых векторных булевых функций от n переменных, каждая координата которых существенно зависит ровно от k переменных, $k < n$. Отправной точкой для этих методов служат функции от k переменных, каждая координата которых существенно зависит от всех переменных. В данной работе рассматриваются два класса таких функций — $\mathcal{K}_n$ и $\mathcal{K}'_n$, устанавливаются некоторые криптографические свойства функций в них, в том числе значения для их нелинейности, степени и компонентной алгебраической иммунности.

1. Описание классов $\mathcal{K}_n$ и $\mathcal{K}'_n$

Для $n \in \mathbb{N}$ рассмотрим обратимые векторные булевы функции $F = (f_1 \dots f_n)$ на $\mathbb{F}_2^n$, такие, что координатные функции $f_i : \mathbb{F}_2^n \to \mathbb{F}_2$, $i = 1, \dots, n$, существенно зависят от всех переменных $x_1, \dots, x_n$; обозначим класс таких функций $\mathcal{F}_n$. В [5] предложен алгоритм 1 построения некоторой такой функции, который состоит в следующем: стартуя с тождественной подстановки $G : \mathbb{F}_2^n \to \mathbb{F}_2^n$, на i-м шаге, $i = 1, \dots, n$, выбираем множество $M_i = \{a, b\}$, где $a, b \in \mathbb{F}_2^n$, a и b отличаются только в i-й координате (соседние по ней) и не принадлежат M_j для $j < i$, и меняем местами значения $G(a)$ и $G(b)$. Таким образом, по построению, $M_i \cap M_j = \varnothing$ для всех $i, j = 1, \dots, n$, $i \neq j$, и для любого $a = a_1 \dots a_n \in \mathbb{F}_2^n$ имеет место

$$f_i(a) = \begin{cases} a_i \oplus 1, & \text{если } a \in M_i, \\ a_i & \text{иначе.} \end{cases} \tag{1}$$

Обозначим класс функций, которые можно получить алгоритмом 1 (при всевозможных способах выбора пар a, b), через $\mathcal{K}_n$. В [5] доказано, что $\mathcal{K}_n \neq \varnothing$ для всех $n > 2$; в [6] описаны некоторые свойства координат функций из $\mathcal{K}_n$.

Для любой рассматриваемой далее функции $F \in \mathcal{K}_n$ и любого $i \in \{1, \dots, n\}$ под M_i подразумевается та пара векторов в $\mathbb{F}_2^n$, соседних по i-й координате, которую алгоритм 1 выбирает на своём i-м шаге.

В [5] предложена модификация алгоритма 1 построения функций из класса $\mathcal{K}_n$, состоящая в том, что отправной точкой алгоритма является не обязательно тождественная подстановка G, а такая, что каждая координатная функция существенно зависит ровно от одной переменной, т. е. $G(x) = (g_1(x) \dots g_n(x))$, $x = x_1 \dots x_n$, $g_i = x_{j_i}^{\sigma_i}$, где $\{j_1, \dots, j_n\} = \{1, \dots, n\}$, $\sigma_i \in \{0, 1\}$ и $x_i^0 = \overline{x_i}$, $x_i^1 = x_i$, $i = 1, \dots, n$. Будем называть эту модификацию алгоритмом $1'$, а класс функций, которые можно таким образом получить, обозначим $\mathcal{K}'_n$.

В [4, утверждение 1] доказано, что обратимых функций на $\mathbb{F}_2^n$, каждая координатная функция которых существенно зависит от двух переменных, не существует ни при каких $n \in \mathbb{N}$. Таким образом, $\mathcal{F}_2 = \varnothing$, следовательно, $\mathcal{K}_2 = \mathcal{K}'_2 = \varnothing$ ввиду $\mathcal{K}_n \subseteq \mathcal{K}'_n \subseteq \mathcal{F}_n$.

Основные результаты данной работы относятся к свойствам компонент (линейных комбинаций координат) функций из классов $\mathcal{K}_n$ и $\mathcal{K}'_n$.

2. Свойства функций класса $\mathcal{K}_n$

Определение 1. Переменная x_i называется *линейной* (или *фиктивной*) для булевой функции $f(x)$, если $f(a) \neq f(b)$ (или соответственно $f(a) = f(b)$) для каждой пары (a, b) соседних по i-й координате векторов.

Утверждение 1. Пусть $F = (f_1 \dots f_n) \in \mathcal{K}_n$. Тогда для каждого $i = 1, \dots, n$ функция f_i имеет единственную линейную переменную — x_i.

Доказательство. Пусть $a, b \in \mathbb{F}_2^n$ — произвольные соседние по i-й координате векторы. Поскольку множество M_i содержит два соседних по i-й координате вектора, имеет место $a, b \in M_i$ или $a, b \notin M_i$. Отсюда ввиду $a_i \neq b_i$ и (1) имеем $f_i(a) \neq f_i(b)$. Значит, x_i — линейная переменная функции f_i.

Для $k \neq i$ выберем соседние по k-й координате векторы $c, d \in M_k$. Тогда $c, d \notin M_i$ ввиду $M_i \cap M_k = \varnothing$. Получим $f_i(c) = c_i = d_i = f_i(d)$. Следовательно, x_k не является линейной переменной для f_i. ■

Пусть $v = (v_1 \dots v_n) \in (\mathbb{F}_2^n)^* = \mathbb{F}_2^n \setminus \{00 \dots 0\}$. *Компонентой функции* $F = (f_1 \dots f_n)$ называется скалярное произведение $vF : \mathbb{F}_2^n \to \mathbb{F}_2$, $vF(x) = \bigoplus\limits_{i=1}^{n} v_i f_i(x) = \bigoplus\limits_{v_i=1} f_i(x)$. Через $\mathrm{w}(v)$ обозначим вес вектора v (количество единиц в нём).

Лемма 1. Пусть $M = \{a_1, a_2, \dots, a_{2k}\} \subseteq \mathbb{F}_2^n$, $k \geqslant 2$ и векторы a_{2i-1}, a_{2i} соседние по j_i-й координате, $i = 1, \dots, k$, $1 \leqslant j_1 < j_2 < \dots < j_k \leqslant n$. Тогда для любого $i \in \{1, \dots, n\}$ множество M нельзя разбить на пары векторов, соседних по i-й координате.

Доказательство. Заметим, что два соседних по j_i-й координате вектора имеют вид $x0y, x1y$ для некоторых $x \in \mathbb{F}_2^{j_i - 1}$, $y \in \mathbb{F}_2^{n - j_i}$. Тогда сумма по модулю 2 всех векторов из M равна $b = e_{j_1} \oplus e_{j_2} \oplus \dots \oplus e_{j_k}$, где e_t — булев вектор длины n и веса 1, имеющий 1 в t-й координате.

Если множество M можно разбить на k пар векторов, соседних по i-й координате, то

$$\bigoplus_{j=1}^{2k} a_j = \begin{cases} e_i, & \text{если } k \text{ нечётное}, \\ 00 \dots 0, & \text{если } k \text{ чётное}, \end{cases}$$

что противоречит равенству $\bigoplus\limits_{j=1}^{2k} a_j = b$ при $k \geqslant 2$. ■

Утверждение 2. Пусть $F = (f_1 \dots f_n) \in \mathcal{K}_n$. Тогда для каждого $v = v_1 \dots v_n \in \in \mathbb{F}_2^n$, такого, что $\mathrm{w}(v) \geqslant 2$, компонентная функция vF не имеет фиктивных и линейных переменных.

Доказательство. Обозначим $V = \{k : v_k = 1\}$, $M = \bigcup\limits_{k \in V} M_k$. Тогда по построению функции F для любого $a = a_1 \dots a_n \in \mathbb{F}_2^n$ имеет место

$$vF(a) = \begin{cases} \bigoplus\limits_{k \in V} a_k \oplus 1, & \text{если } a \in M, \\ \bigoplus\limits_{k \in V} a_k, & \text{если } a \notin M. \end{cases}$$

Докажем, что для любого $i = 1, \dots, n$ переменная x_i не является ни фиктивной, ни линейной для функции vF, т. е. найдутся соседние по i-й координате векторы a, b, такие, что $vF(a) \neq vF(b)$, и найдутся соседние по i-й координате векторы c, d, такие, что $vF(c) = vF(d)$.

С л у ч а й 1: $v_i = 1$. Выберем $a, b \in M_i \subseteq M$. Тогда

$$vF(a) = \bigoplus_{k\in V} a_k \oplus 1 = \bigoplus_{k\in V\setminus\{i\}} a_k \oplus a_i \oplus 1,$$
$$vF(b) = \bigoplus_{k\in V} b_k \oplus 1 = \bigoplus_{k\in V\setminus\{i\}} b_k \oplus b_i \oplus 1 = \bigoplus_{k\in V\setminus\{i\}} a_k \oplus a_i \neq vF(a),$$

последнее равенство здесь верно ввиду того, что a и b соседние по i-й координате.

Заметим, что множество M есть объединение векторов, образующих пары, соседние по координатам с номерами из V. По лемме 1 множество M нельзя разбить на пары векторов, соседних по i-й координате; другими словами, найдётся такой вектор $c \in M$, что соседний ему по i-й координате вектор d не принадлежит M. Получим

$$vF(c) = \bigoplus_{k\in V} c_k \oplus 1 = \bigoplus_{k\in V\setminus\{i\}} c_k \oplus c_i \oplus 1,$$
$$vF(d) = \bigoplus_{k\in V} d_k = \bigoplus_{k\in V\setminus\{i\}} d_k \oplus d_i = \bigoplus_{k\in V\setminus\{i\}} c_k \oplus c_i \oplus 1 = vF(c).$$

С л у ч а й 2: $v_i = 0$. В этом случае пары векторов a, b и c, d меняются местами, а именно: для $c, d \in M_i \not\subseteq M$ получим

$$vF(c) = \bigoplus_{k\in V} c_k, \quad vF(d) = \bigoplus_{k\in V} d_k = \bigoplus_{k\in V} c_k = vF(c).$$

Выберем соседние по i-й координате векторы a, b так, чтобы $a \in M$, $b \notin M$; запишем

$$vF(a) = \bigoplus_{k\in V} a_k \oplus 1, \quad vF(b) = \bigoplus_{k\in V} b_k = \bigoplus_{k\in V} a_k \neq vF(a).$$

Утверждение доказано. ■

3. Свойства функций класса $\mathcal{K}'_n$

Утверждение 3. Пусть $F = (f_1 \dots f_n) \in \mathcal{K}'_n$ и F получена алгоритмом $1'$ из начальной подстановки $G(x) = (x_{j_1}^{\sigma_1} \dots x_{j_n}^{\sigma_n})$. Тогда f_i имеет единственную линейную переменную — x_{j_i}, $i = 1, \dots, n$.

Доказательство. Рассуждения аналогичны приведённым в доказательстве утверждения 1. Для любого $a = a_1 \dots a_n \in \mathbb{F}_2^n$ имеет место

$$f_i(a) = \begin{cases} a_{j_i}^{\sigma_i} \oplus 1, & \text{если } a \in M_{j_i}, \\ a_{j_i}^{\sigma_i} & \text{иначе.} \end{cases}$$

Тогда для всех пар (a, b) соседних по j_i-й координате векторов получаем $a, b \in M_{j_i}$ или $a, b \notin M_{j_i}$, отсюда x_{j_i} — линейная переменная функции f_i; для соседних по j_k-й координате векторов $c, d \in M_{j_k}$ при $k \neq i$ получим $f_i(c) = f_i(d)$, значит, x_{j_k} не является линейной переменной для f_i. ■

Утверждение 4. Пусть $F = (f_1 \dots f_n) \in \mathcal{K}'_n$. Тогда для всех $v = v_1 \dots v_n \in \mathbb{F}_2^n$, таких, что $\mathrm{w}(v) > 2$, компонентная функция vF не имеет фиктивных и линейных переменных.

Доказательство. Аналогично доказательству утверждения 2, только в качестве множества M выберем $M = \bigcup\limits_{k\in V} M_{j_k}$, где $G = (x_{j_1}^{\sigma_1} \dots x_{j_n}^{\sigma_n})$ — начальная подстановка в алгоритме $1'$, из которой получена функция F. ■

Очевидно, что $\mathcal{K}_n \subseteq \mathcal{K}'_n$; в [5] доказано, что алгоритм $1'$ (и, тем более, алгоритм 1) не обладает свойством полноты в том смысле, что существуют функции класса $\mathcal{F}_n$, которые невозможно получить с его помощью. В [6] приведены экспериментальные данные для мощности $|\mathcal{K}_n|$ при $n = 3, \ldots, 6$.

Обозначим $\mathrm{d}(f, g)$ расстояние между булевыми функциями f и g от n переменных: $\mathrm{d}(f, g) = |\{a \in \mathbb{F}_2^n : f(a) \neq g(a)\}|$.

Утверждение 5. $|\mathcal{K}'_n| = 2^n\, n!|\mathcal{K}_n|$.

Доказательство. Поскольку $\mathcal{K}'_2 = \mathcal{K}_2 = \varnothing$, считаем, что $n > 2$.

Множество возможных начальных подстановок алгоритма $1'$ вида $G = \left(x_{j_1}^{\sigma_1} \ldots x_{j_n}^{\sigma_n}\right)$ образует группу Джевонса, порядок которой равен $2^n\, n!$ [7, с. 129]. Для доказательства утверждения надо показать, что алгоритм $1'$, стартуя с разных начальных подстановок, не получит одинаковых функций класса $\mathcal{K}'_n$.

Пусть $F_1, F_2 \in \mathcal{K}'_n$, F_1 получена из подстановки $G_1 = \left(x_{j_1}^{\sigma_1} \ldots x_{j_n}^{\sigma_n}\right)$, F_2 — из подстановки $G_2 = \left(x_{t_1}^{\delta_1} \ldots x_{t_n}^{\delta_n}\right)$ и $G_1 \neq G_2$, т. е. $j_i \neq t_i$ или $\sigma_i \neq \delta_i$ для некоторого $i \in \{1, \ldots, n\}$.

Предположим, что $F_1 = F_2$. По утверждению 3, i-я координата F_1 имеет единственную линейную переменную x_{j_i}, i-я координата F_2 — единственную линейную переменную x_{t_i}, следовательно, $j_i = t_i$, $i = 1, \ldots, n$.

По построению в алгоритме $1'$, для f_{1i} (i-й координаты функции F_1) имеем $\mathrm{d}(f_{1i}, x_{j_i}^{\sigma_i}) = 2$, для f_{2i} (i-й координаты функции F_2) — $\mathrm{d}(f_{2i}, x_{j_i}^{\delta_i}) = 2$. В случае $F_1 = F_2$ и $\sigma_i \neq \delta_i$ получаем $f_{1i} = f_{2i}$ и $\mathrm{d}(f_{1i}, x_{j_i}) = \mathrm{d}(f_{1i}, \bar{x}_{j_i}) = 2$. По неравенству треугольника

$$2^n = \mathrm{d}(x_{j_i}, \bar{x}_{j_i}) \leqslant \mathrm{d}(f_{1i}, x_{j_i}) + \mathrm{d}(f_{1i}, \bar{x}_{j_i}) = 4,$$

что невозможно при $n > 2$. ■

Приведём, следуя [8], определения некоторых криптографических характеристик векторных булевых функций $F : \mathbb{F}_2^n \to \mathbb{F}_2^m$.

Определение 2. *Нелинейностью функции* $F : \mathbb{F}_2^n \to \mathbb{F}_2^m$ называется минимальная нелинейность её компонент:

$$N_F = \min_{v \in (\mathbb{F}_2^m)^*} N_{vF} = \min_{v \in (\mathbb{F}_2^m)^*} \mathrm{d}(vF, \mathcal{A}_n) = \min_{v \in (\mathbb{F}_2^m)^*} \min_{g \in \mathcal{A}_n} \mathrm{d}(vF, g),$$

где $\mathcal{A}_n = \{ax \oplus b : a \in \mathbb{F}_2^n, b \in \mathbb{F}_2\}$ — класс всех аффинных функций от n переменных.

Степень функции F — максимальная степень её компонент (совпадает с максимальной степенью координатных функций):

$$\deg F = \max_{v \in (\mathbb{F}_2^m)^*} \deg vF.$$

Аннигилятором булевой функции $f(x)$ называется булева функция $g(x) \neq \mathrm{const}\, 0$, такая, что $f(x)g(x) = \mathrm{const}\, 0$; класс всех аннигиляторов функции f обозначается $\mathrm{AN}(f)$; алгебраическая иммунность функции f — минимальная степень среди всех аннигиляторов функций f и $f \oplus 1$:

$$\mathrm{AI}(f) = \min_{g \in \mathrm{AN}(f) \cup \mathrm{AN}(f \oplus 1)} \deg g.$$

Компонентная алгебраическая иммунность функции F — минимальная алгебраическая иммунность её компонент:

$$\mathrm{AI}_{\mathrm{comp}}(F) = \min_{v \in (\mathbb{F}_2^m)^*} \mathrm{AI}(vF).$$

Утверждение 6. Для функции $F \in \mathcal{K}'_n$ выполняются следующие свойства:

1) $N_F = 2$;
2) $\deg F = n - 1$;
3) $\mathrm{AI}_{\mathrm{comp}}(F) = 2$;
4) если $v \in \mathbb{F}_2^n$ и $\mathrm{w}(v) \leqslant 2^{n-3}$, то нелинейность компонентной функции vF равна $N_{vF} = 2\mathrm{w}(v)$; в частности, это верно для любого v при $n \geqslant 6$; для всех $v \neq 11111$ при $n = 5$ и т. д.

Доказательство.

1) В [6] доказано, что $N_{f_i} = 2$ для всех координат f_i функции F, поэтому $N_F \leqslant 2$; $N_F \neq 0$, так как по утверждению 4 компонентные функции vF не имеют линейных и фиктивных переменных, следовательно, не являются аффинными; $N_F \neq 1$, так как все компонентные функции обратимой функции F являются уравновешенными [8, Proposition 2], т. е. имеют вес 2^{n-1} — чётное число при $n > 1$, вес функций класса $\mathcal{A}_n$ также чётный, а две функции чётного веса не могут отличаться друг от друга на одном наборе.

2) Следует из доказанного в [6] равенства $\deg f_i = n - 1$ для $i = 1, \ldots, n$.

3) В [6] доказано, что $\mathrm{AI}(f_i) = 2$, $i = 1, \ldots, n$. Докажем, что $\mathrm{AI}(vF) \neq 1$ для всех $v \in (\mathbb{F}_2^n)^*$. Предположим противное: для функции $h \in \{vF, vF \oplus 1\}$ существует аннигилятор $g \in \mathcal{A}_n$. Ввиду уравновешенности функций h и g и равенства $gh = \mathrm{const}\, 0$ получаем, что $h = \overline{g}$, т. е. $h \in \mathcal{A}_n$, что неверно.

4) По построению $\mathrm{d}(vF, vx) = \left| \bigcup\limits_{v_k=1} M_{j_k} \right| = 2\mathrm{w}(v)$, где $vx = \bigoplus\limits_{i=1}^{n} v_i x_i$ и $G = (x_{j_1}^{\sigma_1} \ldots x_{j_n}^{\sigma_n})$ — начальная подстановка в алгоритме $1'$, из которой получена функция F. Поэтому $N_{vF} \leqslant 2\mathrm{w}(v)$. Предположим, что существует функция $g \in \mathcal{A}_n$, $g \neq vx$, такая, что $\mathrm{d}(g, vF) < 2\mathrm{w}(v)$. По неравенству треугольника получим

$$2^{n-1} = \mathrm{d}(g, vx) \leqslant \mathrm{d}(g, vF) + \mathrm{d}(vF, vx) < 2\mathrm{w}(v) + 2\mathrm{w}(v) = 4\mathrm{w}(v),$$

что неверно при $\mathrm{w}(v) \leqslant 2^{n-3}$.

Утверждение доказано. ■

Замечание 1. Эксперименты показывают, что равенство $N_{vF} = 2\mathrm{w}(v)$ выполняется и для $n = 5$ и $v = 11111$.

Автор выражает благодарность Е. Е. Трифоновой и Н. М. Киселевой, чьи компьютерные эксперименты помогли сформулировать свойства функций класса $\mathcal{K}'_n$.

ЛИТЕРАТУРА

1. *Agibalov G. P.* Substitution block ciphers with functional keys // Прикладная дискретная математика. 2017. № 38. С. 57–65.
2. *Agibalov G. P. and Pankratova I. A.* Asymmetric cryptosystems on Boolean functions // Прикладная дискретная математика. 2018. № 40. С. 23–33.
3. *Agibalov G. P.* ElGamal cryptosystems on Boolean functions // Прикладная дискретная математика. 2018. № 42. С. 57–65.
4. *Панкратова И. А.* Об обратимости векторных булевых функций // Прикладная дискретная математика. Приложение. 2015. № 8. С. 35–37.
5. *Pankratova I. A.* Construction of invertible vectorial Boolean functions with coordinates depending on given number of variables // Материалы Междунар. науч. конгресса по информатике: Информационные системы и технологии. Республика Беларусь, Минск, 24–27 окт. 2016. Минск: БГУ, 2016. С. 519–521.

6. *Карпова Л. А., Панкратова И. А.* Свойства координатных функций одного класса подстановок на $\mathbb{F}_2^n$ // Прикладная дискретная математика. Приложение. 2017. № 10. С. 38–40.
7. *Логачев О. А., Сальников А. А., Ященко В. В.* Булевы функции в теории кодирования и криптологии. М.: МЦНМО, 2004.
8. *Carlet C.* Vectorial Boolean Functions for Cryptography. Cambridge: Cambridge University Press, 2010. 93 p.

REFERENCES

1. *Agibalov G. P.* Substitution block ciphers with functional keys. Prikladnaya Diskretnaya Matematika, 2017, no. 38, pp. 57–65.
2. *Agibalov G. P. and Pankratova I. A.* Asymmetric cryptosystems on Boolean functions. Prikladnaya Diskretnaya Matematika, 2018, no. 40, pp. 23–33.
3. *Agibalov G. P.* ElGamal cryptosystems on Boolean functions. Prikladnaya Diskretnaya Matematika, 2018, no. 42, pp. 57–65.
4. *Pankratova I. A.* Ob obratimosti vektornykh bulevykh funktsiy [On the invertibility of vector Boolean functions]. Prikladnaya Diskretnaya Matematika. Prilozhenie, 2015, no. 8, pp. 35–37. (in Russian)
5. *Pankratova I. A.* Construction of invertible vectorial Boolean functions with coordinates depending on given number of variables. Proc. CSIST'16, Minsk, BSU Publ., 2016, pp. 519–521.
6. *Karpova L. A. and Pankratova I. A.* Svoystva koordinatnykh funktsiy odnogo klassa podstanovok na $\mathbb{F}_2^n$ [Properties of coordinate functions for a class of permutations on $\mathbb{F}_2^n$]. Prikladnaya Diskretnaya Matematika. Prilozhenie, 2017, no. 10, pp. 38–40. (in Russian)
7. *Logachev O. A., Sal'nikov A. A., and Yashchenko V. V.* Bulevy funktsii v teorii kodirovaniya i kriptologii [Boolean Functions in Coding Theory and Cryptology]. Moscow, MCCME Publ., 2004, 472 p. (in Russian)
8. *Carlet C.* Vectorial Boolean Functions for Cryptography. Cambridge: Cambridge University Press, 2010. 93 p.

УДК 519.115.1

ПРЕОБРАЗОВАНИЕ ДОНАХЬЮ: КАРУСЕЛЬНЫЕ ЭФФЕКТЫ И РУЧНЫЕ КОМПОНЕНТЫ

И. А. Пушкарев, В. А. Бызов

Вятский государственный университет, г. Киров, Россия

Изучается динамическая система, действующая на комбинаторных интерпретациях чисел Каталана. Описаны два новых карусельных эффекта (для триад и ростков), первый из которых использован для перехода от локального свойства деревьев (отсутствие триад определённых типов) к некоторым глобальным свойствам орбит. Показано, что отмеченный эффект носит массовый характер. Построено несколько новых ручных классов орбит, в том числе не только с постоянными длинами орбит, но и с растущими как $\Theta(n^2)$. Решены соответствующие перечислительные задачи.

Ключевые слова: *числа Каталана, плоские кубические деревья, преобразование Донахью, орбиты, карусельный эффект, ручные компоненты.*

DOI 10.17223/20710410/44/2

DONAGHEY'S TRANSFORMATION: CAROUSEL EFFECTS AND TAME COMPONENTS

I. A. Pushkarev, V. A. Byzov

Vyatka State University, Kirov, Russia

E-mail: vbyzov@yandex.ru

In this paper, Donaghey's transformation is investigated. It is a combinatorial dynamical system, based on the combinatorial interpretations of Catalan numbers, with the transition function of it being the composition of the direct and the inverse bijections between cubic and non-cubic trees. The dynamical system under investigation operates on a finite set and is inversible; therefore it is a mere permutation of trees. The properties of the cycles of this permutation, called the orbits, are studied in terms of permutation of structural elements of trees. In the course of studies, the systematic initiation of particular effects is indicated. These particular effects are referred to as "carousel": it is the movement of objects from one classification category to another, typical of natural classifications. In this form, they look as an indicator of system complexity. Two new carousel effects for structural elements, referred to as triads and germs, are described. The carousel effect for triads is used for the detection of two families of trees; the lengths of all tree arcs in the first family are equal to one; in the second family, they are equal to two. Here, the term "the arcs of the orbit" is used to denote its fragments between two trees, which have no left subtrees. Therefore, the properties of the arcs are the global properties of the orbits, and the carousel effects are local. The corresponding enumeration problems are solved: it is demonstrated that the number of trees of the first family increases as $\dfrac{C}{n^{3/2}}\left(\dfrac{5}{2^{4/3}-2^{2/3}+1}\right)^n$, the number of trees of the second family — as $\dfrac{3^{n+1/2}}{n^{3/2}\sqrt{\pi}}$ (n is the number of triads).

The paper presents the family of cycles with the length 6, which are different from the ones discovered by L. Shapiro, the number of which increases as $\Theta(n^2)$, and the family of cycles with length 9, the number of which increases as $\Theta(2^{n/2})$. A class of orbits with the lengths growing up as $\Theta(n^2)$ is detected.

Keywords: *Catalan numbers, plane cubic trees, Donaghey's transformation, carousel effect, tame components.*

Введение

Плоские деревья с висячим корнем и плоские кубические деревья с висячим корнем [1] — две хорошо изученные комбинаторные интерпретации чисел Каталана.

Напомним основные определения.

Определение 1 [2]. Деревом, растущим из корня r, называется слабосвязный орграф, в котором полустепень входа каждой вершины, кроме r, равна 1, а полустепень входа вершины r равна нулю.

В [2] показано, что условие слабой связности в данном случае равносильно отсутствию ориентированных циклов и что в любую некорневую вершину ведёт единственный ориентированный путь из корня. Все деревья, рассматриваемые в данной работе, являются конечными, то есть имеют конечное множество вершин (из чего следует, что и все остальные множества структурных элементов таких деревьев конечны).

Определение 2.

1. Пусть u и v — две различные вершины дерева, соединённые ориентированным ребром $e = (u, v)$. Вершина v называется *сыном* вершины u, а вершина u — *отцом* вершины v.
2. Дерево, растущее из корня r, называется *деревом с висячим корнем*, если множество сыновей корня r состоит ровно из одной вершины.
3. Вершины дерева, имеющие непустое множество сыновей, называются *нелистьями*. Вершины дерева, не имеющие сыновей, называются *листьями*.

Определение 3. Дерево, растущее из корня r, называется *плоским*, если множество сыновей каждой его вершины линейно упорядочено. В силу конечности можно считать, что сыновья каждой вершины занумерованы некоторым начальным отрезком натурального ряда: первый сын, второй сын и т. д.

Замечание 1. Термин «плоское» в данном случае не имеет никакого отношения к планарности, его следует понимать в смысле «изображённое в плоскости». При изображении плоского дерева на плоскости сыновей конкретной фиксированной вершины, как правило, располагают под ней примерно на одной высоте слева направо в порядке возрастания номеров.

Определение 4. Плоское дерево с висячим корнем называется *кубическим*, если множество сыновей любого нелиста, кроме корня, состоит ровно из двух элементов. В этом случае термин «первый сын» принято заменять термином «левый сын», а «второй сын» — термином «правый сын».

Определение 5. Плоское дерево называется *бинарным*, если множество сыновей любого нелиста, кроме корня, состоит не более чем из двух элементов; для множества сыновей каждого нелиста возможен один из трёх альтернативных вариантов:

1) вершина имеет только левого сына;
2) вершина имеет только правого сына;

3) вершина имеет ровно двух сыновей, один из них левый, а другой — правый.

Замечание 2. При этом типы сыновей существенны: например, два дерева, имеющие одно и то же множество вершин, у одного из которых вершина v является единственным (левым) сыном вершины u, а у другого — правым, различны.

Замечание 3. Бинарные деревья получаются из кубических удалением всех листьев. Очевидно, что соответствующее отображение является биекцией.

Известно, что количество плоских деревьев с висячим корнем и $n+1$ некорневыми вершинами и количество плоских кубических деревьев с висячим корнем и $n+1$ листьями равны числу Каталана $C_n = \dfrac{1}{n+1}\dbinom{2n}{n}$.

Существуют два варианта построения биекции (перехода от кубического дерева к некубическому) — левый и правый. При правом варианте вершинами некубического дерева становятся правые цепи (потомков). Опишем его подробно.

Определение 6. Пусть $u = u_1$ — вершина плоского кубического дерева, не являющаяся правым сыном никакой вершины (то есть являющаяся либо левым сыном некоторой вершины, либо сыном корня). Обозначим u_2 её правого сына, u_{k+1} — правого сына вершины u_k, $k = 2, 3, \ldots$ (до тех пор, пока правые сыновья существуют). *Правой цепью* (потомков вершины u) называется максимальная (возможно, одноэлементная, если u является листом) цепь вершин, которые можно пронумеровать таким образом. В силу максимальности последняя вершина такой цепи — обязательно лист. Вершинами некубического дерева являются правые цепи потомков.

Потомки вершины w, отождествлённой с данной цепью, определяются следующим образом. Все вершины правой цепи, кроме последней, являются нелистьями. Следовательно, кроме правых сыновей, они имеют и левых, каждый из которых является началом аналогичной цепи правых потомков. Соответствующие цепи правых потомков, по определению, являются сыновьями w (следовательно, количество её сыновей на единицу меньше длины соответствующей ей правой цепи). При этом порядок сыновей совпадает с порядком вершин в цепи, от которых они ответвляются.

Сыном корня некубического дерева, соответствующего данному кубическому, является правая цепь потомков сына корня, называемая *старшей правой цепью.*

Тем самым правый вариант перехода от кубического дерева T к некубическому $r(T)$ описан полностью. Он проиллюстрирован на рис. 1, где несколько правых цепей обведены пунктиром. Левый вариант l определяется аналогично.

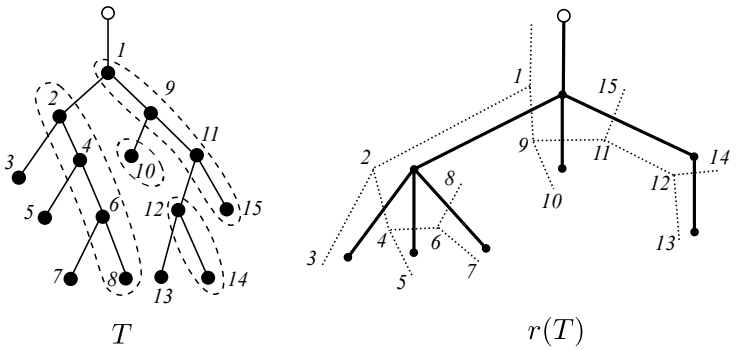

Рис. 1. Правая биекция r

Обратный переход (от некубического дерева к кубическому) также имеет левый и правый варианты. Опишем подробно правый вариант — отображение, обратное к r.

При этом переходе вершинам кубического дерева в некубическом соответствуют объекты, называемые *уголками*.

Определение 7. Пусть u — вершина плоского дерева. *Уголком* называется тройка (u, e_1, e_2), где e_1 и e_2 — два (необязательно различных) ребра, смежных с вершиной u. Уголок (u, e_1, e_2) может быть:

— *открывающим*, если e_1 — ребро, входящее в вершину u, e_2 — первое ребро, выходящее из вершины u;
— *закрывающим*, если e_2 — ребро, входящее в вершину u, e_1 — последнее ребро, выходящее из вершины u;
— *регулярным*, если оба ребра e_1 и e_2 выходят из вершины u и при этом являются соседними в смысле линейного порядка на сыновьях вершины u, то есть имеют номера n и $n+1$;
— *развёрнутым*, если оба ребра $e_1 = e_2$ совпадают с ребром, входящим в вершину u (которая в этом случае обязана быть листом);
— *корневым*, если $u = r$ — корень дерева и оба ребра $e_1 = e_2$ совпадают с ребром, выходящим из вершины r.

Все уголки, рассматриваемые в работе, принадлежат к одному из пяти указанных типов. На рис. 2 уголок *1* является корневым, уголки *2* и *6* — открывающими, *3*, *4* и *7* — регулярными, *5* и *8* — закрывающими, *9* — развёрнутым.

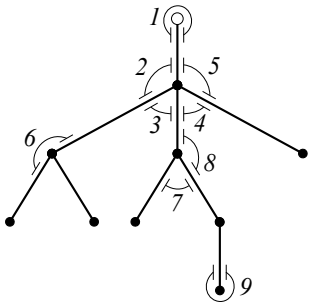

Рис. 2. Уголки в некубическом дереве

На множестве уголков плоского некубического дерева зададим структуру плоского кубического дерева, тем самым определяя его.

Определение 8.

1. Пусть (r, e, e) — корневой уголок плоского дерева. Он является корнем искомого кубического дерева. Его единственным сыном называется уголок (v, e, e_1), где v — сын корня r; $e = (r, v)$; e_1 — первое ребро, выходящее из вершины v, если такое существует, или $e_1 = e$, если таких нет (в этом случае сын корня оказывается развёрнутым уголком, а дерево тривиальное — состоит только из корня и единственной некорневой вершины).

2. Пусть (u, e_1, e_2) — уголок плоского дерева, не являющийся ни развёрнутым, ни закрывающим. Его правым сыном называется уголок (u, e_2, e_3), где e_3 — ребро, выходящее из вершины u, следующее за ребром e_2, если e_2 — не последнее ребро, выходящее из вершины u, или ребром, входящим в вершину u, если e_2 — последнее такое ребро (в этом случае правый сын оказывается закрывающим уголком).

3. Пусть (u, e_1, e_2) — уголок плоского дерева, не являющийся развёрнутым. Его левым сыном называется уголок (v, e_2, e_3), где v — конец ребра e_2; e_3 — первое ребро, выходящее из вершины v, если такое есть, или $e_3 = e_2$, если таких нет (в последнем случае левый сын оказывается развёрнутым уголком).

Иными словами, переход к левому сыну соответствует переходу по ребру вниз, а к правому сыну — переходу через ребро направо.

Считая для определённости, что обе биекции l и r преобразуют кубическое дерево в некубическое, рассмотрим композицию $\tau = l^{-1} \circ r$ (используется правая запись: сначала выполняется r, а затем l^{-1}).

Определение 9. Отображение τ (биекция множества кубических деревьев с n листьями на себя) называется *преобразованием Донахъю* множества плоских кубических деревьев.

Исключив из конструкции некубические деревья, можно получить непосредственное описание преобразования Донахью на языке кубических деревьев, а именно:

1) Дерево разбивается на правые цепи, как в описании биекции r.
2) Каждая правая цепь становится левой, порядок в ней инвертируется. В частности, старшим элементом становится тот, который в исходной цепи являлся листом.
3) Линейный порядок на сыновьях рассматриваемой цепи сохраняется.

Замечание 4. Несмотря на сохранение порядка, цепи, являющиеся сыновьями, после преобразования становятся смежны не с той же самой вершиной цепи-отца, с которой были смежны перед преобразованием, а с той, которая была её правым сыном (рис. 3).

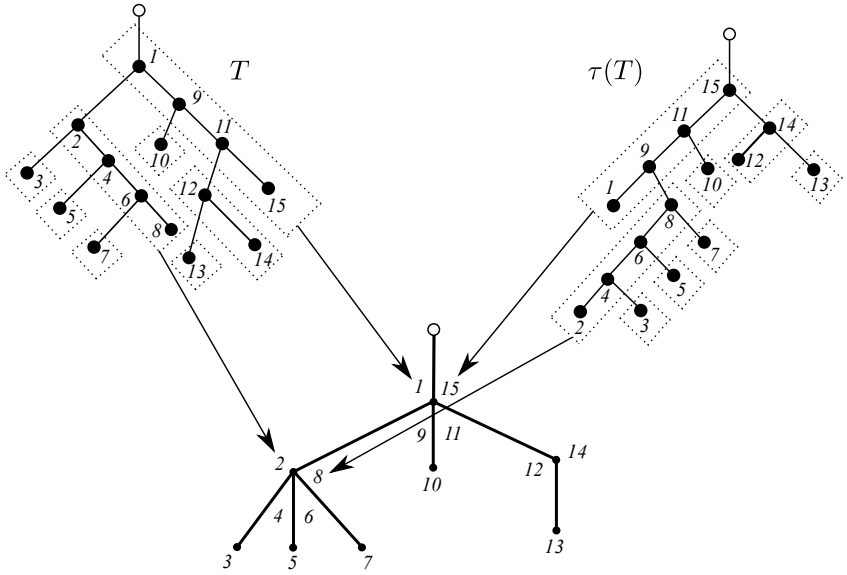

Рис. 3. Преобразование Донахью кубических деревьев

Приведённое определение преобразования Донахью не является единственно возможным, имеется большое количество довольно разнообразных его равносильных определений. Источником таковых, во-первых, является большое количество комбинаторных интерпретаций чисел Каталана, каждая из которых позволяет некоторым образом определить преобразование. В [3] приводится небольшой обзор такого рода определений на языках наиболее распространённых интерпретаций (положительные пути хромого короля, правильные расстановки скобок, триангуляции правильного многоугольника и т. д.). Во-вторых, даже на языке деревьев можно дать несколько определений, равносильность которых не вполне тривиальна. Дело в том, что описание плоского дерева как графа, определённым образом вложенного в плоскость, не

является единственно возможным. Например, полезным является рекурсивное определение плоского кубического дерева с висячим корнем.

Определение 10. Плоским кубическим деревом с висячим корнем называется объект, состоящий из следующих частей:

1) выделенной вершины r — висячего корня;
2) вершины u_0, являющейся левым сыном корня;
3) цепочки правых потомков $u_1, u_2, \ldots, u_k$ вершины u_0, в которой вершина u_{i+1}, $i = 1, \ldots, k-1$, является правым сыном вершины u_i, а вершина u_k не имеет сыновей (является листом);
4) набора *дочерних* деревьев $T_0, T_1, \ldots, T_{k-1}$, корни которых отождествляются с вершинами $u_0, u_2, \ldots, u_{k-1}$ соответственно.

Из симметричности понятия кубического дерева следует, что существует симметричное рекурсивное определение — в терминах цепочек левых потомков. Эти определения имеют один легкоустранимый дефект: в одном (приведённом) сын корня является левым сыном, в другом (симметричном) — правым. Поэтому в них разумно внести поправку, поясняющую, что «первоначальный» сын корня является одновременно и левым и правым сыном.

В терминах рекурсивных определений плоского кубического дерева с висячим корнем нетрудно дать формально другое определение преобразования Донахью.

Определение 11. Пусть T — плоское кубическое дерево с висячим корнем, сыном корня u_0, цепочкой правых потомков $u_1, u_2, \ldots, u_k$ (u_k — лист) и дочерними деревьями $T_0, T_1, \ldots, T_{k-1}$, имеющее n вершин. Пусть для всех деревьев, имеющих меньшее количество вершин, преобразование Донахью определено, в частности, пусть $D_0, D_1, \ldots, D_{k-1}$ — результаты преобразований деревьев $T_0, T_1, \ldots, T_{k-1}$ соответственно. Результатом преобразования дерева T называется дерево с корнем r, сыном корня u_k, цепочкой левых потомков $u_{k-1}, u_{k-2}, \ldots, u_0$ и дочерними поддеревьями $D_0, D_1, \ldots, D_{k-1}$, корни которых отождествлены с вершинами $u_1, u_2, \ldots, u_k$ соответственно.

Теорема 1 (равносильность определений преобразования Донахью).
Определения 9 и 11 равносильны.

Доказательство. Индукция по числу вершин некубического дерева.

Б а з а. Для тривиального некубического дерева с единственным некорневым листом утверждение очевидно.

И н д у к т и в н ы й п е р е х о д. Пусть дерево T имеет n некорневых вершин и для всех некубических деревьев с меньшим количеством некорневых вершин утверждение доказано.

Предположим, что сын корня u дерева T имеет m сыновей. Тогда правая обратная биекция r^{-1} превращает u в цепочку правых потомков $s_1, s_2, \ldots, s_{m+1}$ длины $m+1$, а левая обратная l^{-1} — в цепочку левых потомков $t_1, t_2, \ldots, t_{m+1}$ той же длины.

Пусть $T_1, T_2, \ldots, T_m$ — дочерние некубические деревья, корень которых отождествлён с сыном корня u, под действием правой обратной биекции они переходят в кубические деревья $R_1, R_2, \ldots, R_m$, а под действием левой обратной биекции — в деревья $L_1, L_2, \ldots, L_m$ соответственно. Тогда все переходы от деревьев R_i к деревьям L_i являются преобразованиями Донахью в смысле обоих определений (которые для дочерних деревьев равносильны по предположению индукции).

Корни кубических деревьев $R_1, R_2, \ldots, R_m$ отождествляются с вершинами s_1, s_2, $\ldots, s_m$, а корни деревьев $L_1, L_2, \ldots, L_m$ — с вершинами $t_2, t_3, \ldots, t_{m+1}$, что согласуется с обоими определениями. ■

Рекурсивное определение также не является единственной альтернативой на языке деревьев. Дерево можно строить из элементов, не являющихся вершинами, и каждое такое определение дерева позволяет дать своё собственное определение преобразования Донахью.

Примером альтернативного типа элементов, при помощи которого можно описать структуру дерева, являются *юниты*, неформально говоря — «половинки рёбер». При этом каждое ребро распадается на два юнита, верхний и нижний, верхний смежен с двумя другими юнитами в вершине, из которой выходит ребро, а нижний — в вершине, в которую ребро входит. Не останавливаясь на юнитах, перейдём сразу к производным от них более удобным элементам, называемым *триадами*. Триады в первом приближении соответствуют вершинам бинарного дерева, получающегося из кубического удалением листьев.

Определение 12. Сопоставим каждому нелисту u плоского кубического дерева одноимённую триаду — четвёрку (u, s_1, s_2, s_3), где элементы s_i — суть юниты (понимаемые просто как формальный атрибут), смежные с этой вершиной (альтернативно: между собой в этой вершине). Для определённости: s_1 — верхний юнит, s_2 — левый, s_3 — правый. Юниты триады могут быть *свободными* или *связанными*:

1. Верхний юнит s_1 является свободным, если триада соответствует сыну корня, и связанным, если триада соответствует нелисту, являющемуся сыном другого нелиста.
2. Левый юнит s_2 является свободным, если левым сыном нелиста u является лист, иначе является связанным. Аналогично определяется свободность или связанность правого юнита s_3.

Например, если нелист v является левым сыном нелиста u, то левый юнит s_2 триады (u, s_1, s_2, s_3) с верхним юнитом t_1 триады (v, t_1, t_2, t_3) являются связанными (и образуют «полноценное» ребро).

Триаду, содержащую сына корня, назовём *корневой*. Заметим, что на множестве триад дерева можно рассматривать инцидентности аналогично вершинам. Так, например, на рис. 4 слева у триады *7* левым сыном является триада *8*, а правым — триада *9*. Триада *1* является корневой. Дерево, присоединённое к корневой триаде слева, будем называть *левым поддеревом*, справа — правым поддеревом.

Дадим определение преобразования Донахью на языке триад.

Определение 13.

1. Определим максимальную цепочку правых потомков для триад аналогично тому, как это сделано для вершин. В частности, она начинается с триады, не являющейся ничьим правым потомком, — либо с левого сына, либо с корневой триады. Аналогично определяются максимальные цепочки левых потомков.

2. Операцией *переклейки цепочки* назовём следующее преобразование дерева, сформированного из триад:

— выбирается максимальная цепочка правых потомков;
— выделяются первая и последняя (не имеющая правого потомка) триады этой цепочки; заметим, что верхний юнит первой триады является связанным (кроме случая корневой триады), а правый юнит последней триады — свободным;

— правый юнит последней триады связывается с тем же самым юнитом, с которым был связан верхний юнит первой триады; если первая триада была корневой, то правый юнит последней триады становится верхним юнитом корневой триады;
— верхний юнит первой триады освобождается;
— все триады цепочки подвергаются однотипному переименованию юнитов: правый становится верхним, верхний — левым, левый — правым.

Заметим, что в результате этого преобразования максимальная цепочка правых потомков становится максимальной цепочкой левых потомков, при этом порядок старшинства триад инвертируется.

3. Преобразованием Донахью дерева называется процедура, состоящая из переклейки максимальной цепочки правых потомков сына корня и последующего аналогичного преобразования всех дочерних деревьев.

Следующая теорема очевидна по построению.

Теорема 2 (равносильность определений преобразования Донахью-2).
Определения 11 и 13 равносильны.

Определение 13 проиллюстрировано рис. 4.

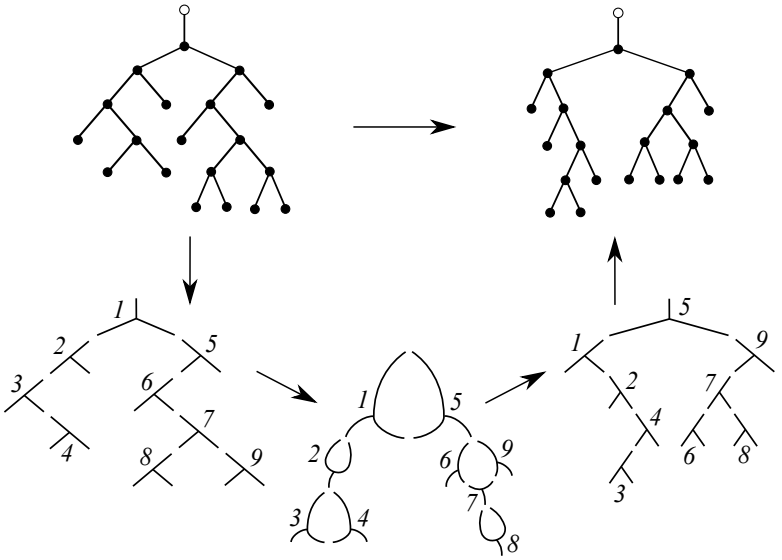

Рис. 4. Преобразование Донахью для дерева, разбитого на триады

Преобразование, описанное выше, впервые, по-видимому, рассмотрено в работе Р. Донахью [4]. Кроме того, оно или его аналоги для лесов рассматривались в [5, 6] и финским математиком А. Карттуненом, опубликовавшим некоторые гипотезы о характеристиках этого преобразования в «Энциклопедии числовых последовательностей» [7]. Первый из авторов настоящей работы (независимо) в работе [8] дал определение преобразования Донахью под названием «фрактальный поворот плоских кубических деревьев» (название предложено К. П. Кохасем в личной беседе).

Преобразование Донахью представляется важным объектом исследования по той причине, что выглядит сравнительно сложным (представляется уместной метафора «комбинаторный аналог турбулентности»), при том, что является просто семейством перестановок (каждая из которых — обратимое преобразование деревьев определённого размера). Наиболее естественной исследовательской задачей в отношении этого преобразования представляется изучение циклической структуры соответствующей перестановки кубических деревьев, и эта задача оказывается чрезвычайно сложной даже

в простейших частных случаях, например — описание всех циклов длины 2 этой перестановки. Циклы перестановки деревьев назовём *орбитами* преобразования Донахью.

Повторимся: числа Каталана имеют огромное количество комбинаторных интерпретаций [1] и на языке каждой из них можно дать собственное определение преобразования Донахью. Во всех случаях описание преобразования Донахью оказалось довольно сложным. Однако многообразие комбинаторных интерпретаций чисел Каталана требует рассмотрения возможности того, что на языке одной из них преобразование Донахью вовсе не выглядит «хаотичным». Наличие даже одной подобной достаточно простой комбинаторной интерпретации могло бы лишить смысла теоретическое изучение преобразования Донахью (назовём это *проблемой поиска тривиальной интерпретации*).

Формально некоторый вариант тривиальной интерпретации действительно существует, а именно: рассмотрим объекты, которые назовём *точками орбиты* — орбиты преобразования Донахью, на которых выделено ровно одно дерево. Каждое дерево принадлежит ровно одной орбите, поэтому количество точек орбиты равно числу Каталана. С другой стороны, применение к точке орбиты преобразования есть просто переход к той же самой орбите, на которой отмечено следующее дерево.

Однако заметим, что для построения описанной комбинаторной интерпретации чисел Каталана требуется предварительно знать все орбиты преобразования — то есть в данном случае преобразование выглядит тривиально на комбинаторной интерпретации, которая крайне нетривиально устроена сама. Попросту, нетривиальность преобразования формально перенесена в интерпретацию. Таким образом, построенная интерпретация не обессмысливает изучение преобразования Донахью, но иллюстрирует некоторые существенные формальные трудности, возникающие при обсуждении проблемы поиска тривиальной интерпретации.

1. Специальные классы деревьев и вспомогательные утверждения

Для изложения свойств орбит преобразования Донахью необходимо определить несколько специальных классов деревьев и сделать одно предварительное замечание.

Определение 14. Правой (соответственно левой) *гребёнкой* назовём дерево, все некорневые триады которого входят в цепочку правых (соответственно левых) потомков корневой триады.

При преобразовании Донахью правая гребёнка переходит в левую, а левая — в правую. То есть длина орбиты гребёнок равна двум (рис. 5).

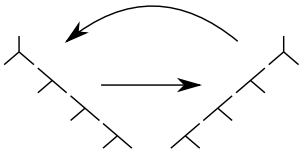

Рис. 5. Правая и левая гребёнки

Определение 15. Дерево, состоящее из n триад $\alpha_1, \alpha_2, \ldots, \alpha_n$, назовём *зигзагом*, если выполняются следующие условия:

1) корневой триадой является триада α_1;
2) при $i \in \{1, 2, \ldots, n-1\}$ единственным сыном триады α_i является триада α_{i+1};
3) если триада α_i является левым сыном, то триада α_{i+1} является правым сыном, и наоборот ($i \in \{2, 3, \ldots, n-1\}$).

Если n нечётно, зигзаг назовём *зигзагом первого рода*, если чётно — *зигзагом второго рода*. Если корневая триада зигзага имеет правого сына, то такой зигзаг назовём *правым*, если левого сына — *левым*. Триады α_1 и α_n назовём *конечными* триадами зигзага.

Для зигзагов первого рода длина цикла преобразований равна трём. Пример орбиты такого дерева приведён на рис. 6. Зигзаги второго рода отличаются от зигзагов первого рода тем, что после трёх преобразований Донахью переходят в симметричные деревья (рис. 7). Таким образом, длина орбиты таких деревьев равна шести.

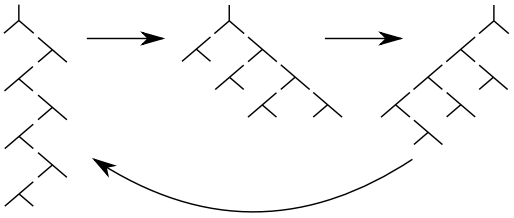

Рис. 6. Орбита зигзага первого рода

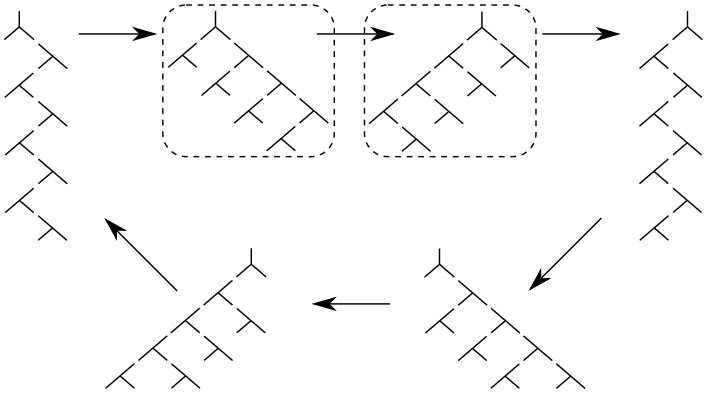

Рис. 7. Орбита зигзага второго рода

Определение 16. *Двухуровневой* правой (соответственно левой) гребёнкой назовём дерево, получающееся при присоединении ко всем триадам правой (соответственно левой) гребёнки левого (соответственно правого) сына.

В правую двухуровневую гребёнку переходит при преобразовании правый зигзаг второго рода. Правая двухуровневая гребёнка при преобразовании переходит в левую двухуровневую гребёнку. То есть двухуровневые гребёнки принадлежат орбите зигзага второго рода. На рис. 7 они выделены пунктиром.

Определение 17. Правой огибающей цепью называется последовательность триад дерева, полученная по следующим правилам:

1) первым элементом этой последовательности становится корневая триада;
2) если у последнего добавленного в последовательность элемента есть правый сын, то этот сын становится следующим элементом последовательности;
3) если у последнего добавленного в последовательность элемента нет правого сына, но есть левый сын, то этот левый сын становится следующим элементом последовательности;
4) если у последнего добавленного в последовательность элемента нет ни правого, ни левого сыновей, то формирование правой огибающей цепи завершается.

Лемма 1.

1) Дерево $\tau(T)$ имеет пустое правое поддерево тогда и только тогда, когда старшая правая цепь дерева T заканчивается триадой, не имеющей левого сына.
2) Дерево $\tau^2(T)$ имеет пустое правое поддерево тогда и только тогда, когда правая огибающая цепь дерева T заканчивается триадой, являющейся левым сыном.

Доказательство.

1. Справедливость данного утверждения относительно легко проверяется непосредственным вычислением и проиллюстрирована на рис. 8 слева (треугольниками на этом рисунке обозначены возможные поддеревья). Пусть α — триада, являющаяся концом старшей правой цепи дерева T. Тогда α становится корневой триадой дерева $\tau(T)$. Из определения преобразования у триады α нет левого сына в дереве T тогда и только тогда, когда в дереве $\tau(T)$ у α нет правого сына, то есть $\tau(T)$ имеет пустое правое поддерево. Таким образом, п. 1 леммы доказан.

2. Если дерево T устроено так, как показано на рис. 8 справа, то $\tau(T)$ имеет вид, изображённый слева, в этом нетрудно убедиться прямой проверкой. При этом триады правой огибающей цепи дерева T, не имеющие правых сыновей, образуют в $\tau(T)$ старшую правую цепь. В этом случае триада β перейдет в триаду α, и из п. 1 следует, что дерево $\tau^2(T)$ имеет пустое правое поддерево.

С другой стороны, если правая огибающая цепь дерева T заканчивается правым сыном, то максимальная цепь правых потомков, начинающаяся с триады β, содержит больше одной триады. Эта цепь перейдёт в цепь левых потомков триады α. ■

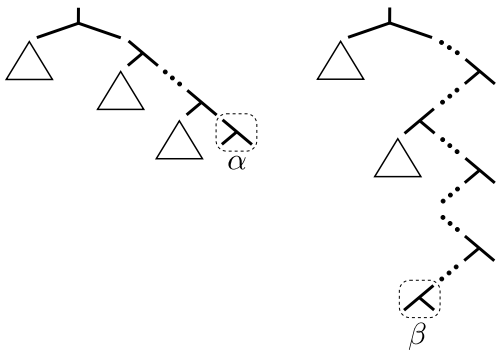

Рис. 8. Иллюстрация к доказательству леммы 1

2. Помеченные варианты преобразования Донахью и карусельные эффекты

Как уже отмечалось, один из возможных подходов к описанию преобразования Донахью на языке деревьев (как, впрочем, и на языке любой другой комбинаторной интерпретации чисел Каталана) — разбить дерево на части некоторого вида, связанные инцидентностью некоторого стандартного вида (например, на вершины или триады) и описать, как изменяется инцидентность под действием преобразования. Это приводит к возможности наблюдать за перемещениями отдельных элементов дерева внутри этого дерева (которое тоже изменяется). Фактически определение на рис. 3 и есть такое описание на языке вершин кубического дерева.

Заметим, что каждая вершина плоского кубического дерева относится к одному из пяти типов: она может быть листом или нелистом, независимо от этого может быть правым или левым сыном и, кроме того, может оказаться сыном корня. Обозначим

эти типы аббревиатурами ПЛ (правый лист), ЛЛ (левый лист), ПН (правый нелист), ЛН (левый нелист) и СК (сын корня).

Теорема 3 (карусельный эффект для вершин [3, теорема 4.1]).

Под действием преобразования Донахью вершина кубического дерева сменяет типы строго по схеме рис. 9.

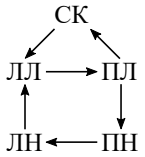

Рис. 9. Карусельный эффект для вершин

Заметим, что вершина не просто сменяет по циклу четыре основных типа, но время от времени может изменять тип, проходя через СК. Тем самым преобразование Донахью становится комбинаторной моделью динамических обменных процессов некоторого специального вида.

Кроме того, если бы каждая вершина просто меняла типы по одному и тому же циклу («тривиальный» карусельный эффект) и разбиение вершин на типы было инвариантно, то это автоматически означало бы, что преобразование не является очень сложным, поэтому нетривиальный карусельный эффект можно рассматривать как один из атрибутов сложного преобразования. В связи с этим возникает содержательный аналог проблемы поиска тривиальной интерпретации — поиск структурного описания деревьев, не подчинённого никакому нетривиальному карусельному эффекту. Никаких подходов к решению данной проблемы на данный момент не известно.

В свою очередь, триады можно разделить на одиннадцать типов по формальным признакам наличия свободных юнитов (рис. 10).

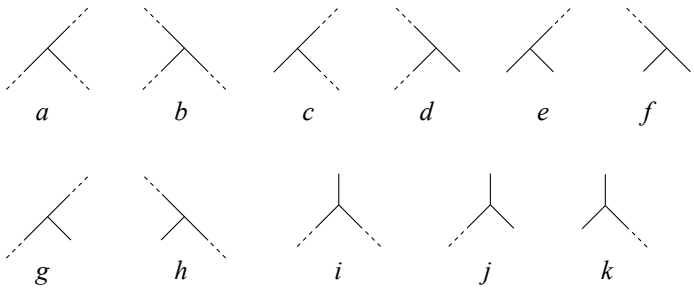

Рис. 10. Классификация триад

Это разбиение приводит к более содержательному карусельному эффекту: имеет место следующая

Теорема 4 (карусельный эффект для триад). Под действием преобразования Донахью триада кубического дерева сменяет типы строго по схеме рис. 11.

Доказательство. Справедливость правил изменения типа триад непосредственно следует из определения преобразования Донахью и проверяется большим количеством непосредственных вычислений. Неоднозначность типа образа триад типов f и d обусловлена тем, что триада, имеющая тип f или d, может находиться в конце старшей правой цепи дерева, поэтому после преобразования она может стать корневой. ■

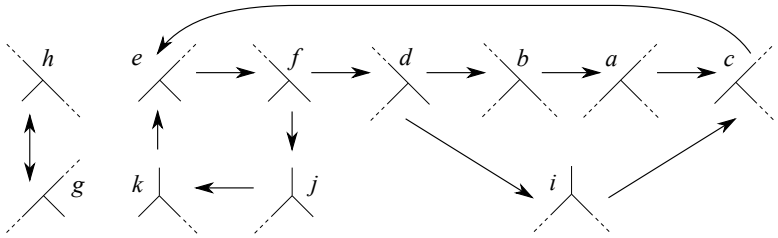

Рис. 11. Карусельный эффект для триад

Замечание 5. В данном случае орграф, описывающий карусельный эффект, состоит из двух компонент связности. Левая тривиальная компонента соответствует замеченной Р. Донахью в [4] «инвариантности гребёнок».

Замечание 6. Правая нетривиальная компонента содержит две точки ветвления и основной цикл длины 6. Эмпирическое изучение преобразования Донахью показало, что среди его циклов, по крайней мере, для деревьев небольшого размера, преобладают циклы длины 6. Возможно, это не случайное совпадение.

3. Одно приложение карусельного эффекта для триад

Карусельные эффекты могут использоваться не только как характерный признак сложности системы, но и для выяснения некоторых глобальных свойств орбит для деревьев специального вида. Подробно разберём пример такого рода.

Определение 18. Для дерева T с пустым левым поддеревом назовём *дугой* такую последовательность деревьев $T, \tau(T), \tau^2(T), \ldots, \tau^k(T)$, что $\tau^k(T)$ обладает пустым правым поддеревом и k — минимально возможное. Число k назовём длиной дуги. Дерево с пустым правым поддеревом $\tau^k(T)$ при однократном преобразовании переходит в дерево $\tau^{k+1}(T)$ с пустым левым поддеревом. Дугу дерева $\tau^{k+1}(T)$ назовём второй дугой дерева T. Аналогично определяются третья дуга дерева, четвертая и т. д.

Листовой назовём триаду, не имеющую сыновей (то есть со свободными вторым и третьим юнитами). Свойство дерева, состоящее в том, что все его листовые триады являются левыми сыновьями, назовём свойством lleaves; свойство, состоящее в том, что все листовые триады являются правыми сыновьями, — свойством rleaves. Если в дереве отсутствует триады типа x, то скажем, что оно обладает свойством $\mathrm{miss}(x)$ (здесь вместо x может быть любой из 11 типов триад на рис. 10).

Теорема 5.

1) Если дерево с пустым левым поддеревом обладает свойствами rleaves, $\mathrm{miss}(a)$ и $\mathrm{miss}(d)$, то длины всех его дуг равны 1.
2) Если дерево с пустым левым поддеревом обладает свойствами lleaves и $\mathrm{miss}(a)$, то длины всех дуг этого дерева равны 2.

Доказательство.

1. Пусть дерево T с пустым левым поддеревом удовлетворяет свойствам rleaves, $\mathrm{miss}(a)$ и $\mathrm{miss}(d)$. Покажем, что длина его первой дуги равна 1 и дерево $\tau^2(T)$ тоже удовлетворяет свойствам rleaves, $\mathrm{miss}(a)$ и $\mathrm{miss}(d)$. Так как в T нет триад типа d, старшая правая цепь этого дерева заканчивается триадой, не имеющей левого сына. Поэтому длина его дуги действительно равна 1.

Сохранение свойств rleaves, $\mathrm{miss}(a)$ и $\mathrm{miss}(d)$ следует из карусельного эффекта для триад дерева. Действительно, если в $\tau^2(T)$ есть листовая триада, являющаяся левым сыном, то в дереве T ей соответствует триада типа a, i или j, а триад этих типов

в дереве T нет по условию. Если в $\tau^2(T)$ есть триада типа a, то в T должна быть триада типа d. Если в $\tau^2(T)$ есть триада типа d, то в T должна быть листовая триада, являющаяся левым сыном.

2. Доказательство основано на следующем соображении: длина дуги дерева равна 2 тогда и только тогда, когда конец правой огибающей цепи дерева — триада, являющаяся левым сыном (см. лемму 1). У дерева T все листовые триады — левые сыновья, поэтому длина его первой дуги равна 2. Значит, $\tau^3(T)$ обладает пустым левым поддеревом.

Дерево T обладает свойствами lleaves и miss(a). Покажем, что дерево $\tau^3(T)$ тоже обладает этими свойствами.

Пусть дерево $\tau^3(T)$ не обладает свойством lleaves, то есть содержит листовую триаду, являющуюся правым сыном (тип f). Из карусельного эффекта следует, что в дереве T эта триада α может иметь тип a, i или j. Но триад таких типов в дереве T нет.

Пусть дерево $\tau^3(T)$ не обладает свойством miss(a), то есть содержит триаду типа a. Из карусельного эффекта в дереве T эта триада обладает типом f, что противоречит свойству lleaves дерева T. ■

Покажем, что отмеченный эффект является массовым. Для этого вычислим количество деревьев, подчиняющихся этому эффекту.

Обозначим через s_n количество деревьев с пустым левым поддеревом и n некорневыми триадами, удовлетворяющих свойствам rleaves, miss(a) и miss(d). Заметим, что $s_1 = s_2 = s_3 = 1$.

Пусть T — такое дерево, α — триада, являющаяся правым сыном корневой триады T, l — старшая левая цепь дерева с корнем α, β — конечная триада этой цепи. Возможны два случая:

1) цепь l состоит из одной триады ($\alpha = \beta$);
2) длина l больше 1.

В первом случае дерево T имеет вид рис. 12 слева (треугольниками обозначены поддеревья, формулы внутри треугольников — количество триад соответствующих деревьев). Здесь T_1 удовлетворяет свойствам rleaves, miss(a) и miss(d), поэтому количество таких деревьев равно s_{n-1}.

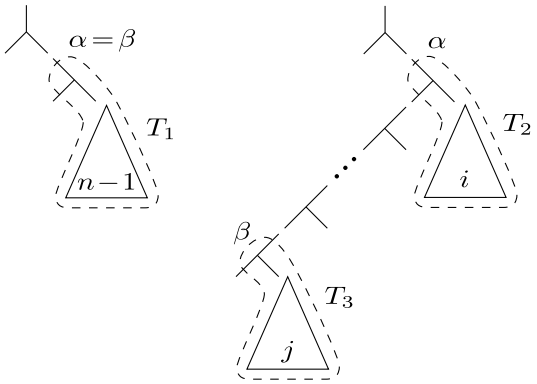

Рис. 12. Деревья, удовлетворяющие свойствам rleaves, miss(a) и miss(d)

Во втором случае непустые поддеревья могут быть присоединены только к триадам α и β, как показано на рис. 12 справа, иначе в дереве T образуется триада типа a. Дерево с корнем α обозначено T_2, с корнем β — T_3. Пусть T_2 содержит i некорневых

триад, $T_3 - j$ некорневых триад. У деревьев T_2 и T_3 не может быть пустых правых поддеревьев, потому что тогда нарушаются условия rleaves и miss(d). Поэтому сумма $i+j$ может принимать все значения от 2 до $n-2$. Таким образом, получаем рекуррентное соотношение

$$s_n = s_{n-1} + \sum_{k=2}^{n-2} \left(\sum_{i+j=k} s_j s_j \right), \tag{1}$$

где внутренняя сумма берётся по всем упорядоченным разложениям k в сумму двух натуральных слагаемых.

Рассмотрим производящую функцию $S(x) = \sum\limits_{n=1}^{\infty} s_n x^n$. Преобразуем её с учётом (1) и того, что $s_1 = s_2 = s_3 = 1$:

$$\begin{aligned} S(x) = \sum_{n=1}^{\infty} s_n x^n = x + s_1 x^2 + s_2 x^3 + (s_3 + s_1 s_1) x^4 + \\ + (s_4 + s_1 s_1 + s_1 s_2 + s_2 s_1) x^5 + \ldots = \\ = x + xS(x) + x^2 S(x)^2 + x^3 S(x)^2 + \ldots = x + xS(x) + x^2 \frac{S(x)^2}{1-x}. \end{aligned} \tag{2}$$

Рассматривая соотношение (2) как квадратное уравнение относительно $S(x)$, получаем

$$S(x) = \frac{(1-x)^2 - \sqrt{(1-x)(1-3x+3x^2-5x^3)}}{2x^2}. \tag{3}$$

В числителе (3) перед корнем стоит знак «минус». Это следует, например, из того, что в разложении $\sqrt{(1-x)(1-3x+3x^2-5x^3)}$ в ряд Маклорена коэффициент при x^2 равен 1, а $s_0 = 0$. Из теоремы Дарбу [9, с. 252] получаем асимптотическую оценку

$$s_n \sim \frac{C}{n^{3/2}} \left(\frac{5}{2^{4/3} - 2^{2/3} + 1} \right)^n,$$

где C — некоторая константа.

Обозначим через t_n количество деревьев с пустым левым поддеревом и n некорневыми триадами, удовлетворяющих свойствам *lleaves* и *miss*(a). Заметим, что $t_1 = 0$.

Пусть T — такое дерево. Воспользуемся введёнными обозначениями. Как и в случае с последовательностью s_n, возможны два варианта (рис. 12):

1) цепь l состоит из одной триады ($\alpha = \beta$);
2) длина l больше 1.

В первом случае количество рассматриваемых деревьев равно t_{n-1}. Перейдём к рассмотрению второго случая. Если T_2 и T_3 обладают непустыми правыми поддеревьями, то сумма $i + j$ может принимать все значения от 2 до $n-2$. Тогда количество подсчитываемых деревьев равно $\sum\limits_{k=2}^{n-2} \left(\sum\limits_{i+j=k} t_i t_j \right)$, где внутренняя сумма берётся по всем упорядоченным разложениям k в сумму двух натуральных слагаемых.

Если ровно одно из деревьев T_2 или T_3 обладает пустым правым поддеревом, то количество рассматриваемых деревьев равно $2 \left(\sum\limits_{k=1}^{n-2} t_k \right)$. Ещё одно дерево получается, если $i = j = 0$. Таким образом, справедливо следующее рекуррентное соотношение:

$$t_n = t_{n-1} + \sum_{k=2}^{n-2} \left(\sum_{i+j=k} t_i t_j \right) + 2 \sum_{k=1}^{n-2} t_k + 1. \tag{4}$$

Рассмотрим производящую функцию $T(x) = \sum\limits_{n=1}^{\infty} t_n x^n$. Преобразуем её с учётом формулы (4) и того, что $t_1 = 0$:

$$\begin{gathered} T(x) = \sum_{n=1}^{\infty} t_n x^n = t_1 x + (t_1+1)x^2 + (t_2 + 2t_1 + 1)x^3 + (t_3 + t_1 t_1 + \\ +2(t_1+t_2)+1)x^4 + \ldots = (t_1 x^2 + t_2 x^3 + t_3 x^4 + \ldots) + ((t_1 t_1)x^4 + \\ +(t_1 t_1 + t_1 t_2 + t_2 t_1)x^5 + \ldots) + 2(t_1 x^3 + (t_1+t_2)x^4 + \ldots) + \\ +(x^2 + x^3 + \ldots) = xT(x) + x^2(T(x)^2 + xT(x)^2 + x^2 T(x)^2 + \ldots) + \\ +2(t_1(x^3 + x^4 + \ldots) + t_2(x^4 + x^5 + \ldots) + \ldots) + \frac{x^2}{1-x} = \\ = xT(x) + x^2 \frac{T(x)^2}{1-x} + 2x^2 \frac{T(x)}{1-x} + \frac{x^2}{1-x}. \end{gathered} \tag{5}$$

Рассматривая (5) как квадратное уравнение относительно $T(x)$, получаем

$$T(x) = \frac{1 - 2x - x^2 - \sqrt{(1-3x)(1-x)^2(1+x)}}{2x^2}. \tag{6}$$

В (6) перед корнем стоит знак «минус». Это объясняется, например, тем, что в разложении в ряд Маклорена $\sqrt{(1-3x)(1-x)^2(1+x)}$ коэффициент при x^2 равен -1, а $t_0 = 0$. Из теоремы Дарбу получаем следующую асимптотическую оценку:

$$t_n \sim \frac{3^{n+1/2}}{n^{3/2}\sqrt{\pi}}.$$

4. Ростки

Эффект инвариантности гребёнок, описанный в [4], и соответствующая попытка свести преобразование больших деревьев к преобразованиям меньших приводят к понятию ростка, расположенного на ребре кубического дерева.

Определение 19.

1) Пусть T — кубическое дерево, e — правое ребро этого дерева, соединяющее вершину a с сыном b. Рассмотрим новое дерево $T(e)$, которое имеет следующие отличия от дерева T:
   а) правым сыном вершины a является новая вершина c;
   б) правым сыном вершины c является вершина b;
   в) левым сыном вершины c становится новая вершина d, которая является листом (см. рис. 13, треугольниками обозначены поддеревья).
2) Переход от дерева T к дереву $T(e)$ назовём операцией *разбиения ребра*. Новое ребро, соединяющее вершины c и d, назовём *ростком*.
3) Аналогичным образом определяется операция разбиения левого ребра. Будем считать, что для ребра, соединяющего корень с сыном корня, операция разбиения не определена.

Назовём росток, расположенный на ребре, идущем к листу, *внешним*, а на ребре, ведущем к нелисту, — *внутренним*. Заметим, что разбиение ребра (теперь уже не (a,b), а (c,b)) можно повторить, получая гребёнку, этот процесс назовём *реализацией ростка*. В [3] отмечен эффект разрушения внешнего ростка, когда под действием преобразования Донахью внешний росток превращается во всё более и более сложный

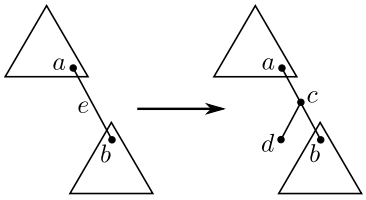

Рис. 13. Разбиение ребра

элемент структуры дерева, и высказана гипотеза о том, что список элементов структуры, в которые может превратиться внешний росток, является бесконечным. Однако имеет место

Теорема 6. Если дерево T_1 получается из дерева T реализацией некоторого внутреннего ростка, то и дерево $\tau(T_1)$ получается из дерева $\tau(T)$ реализацией некоторого внутреннего ростка. Иными словами, внутренние ростки сохраняются под действием преобразования Донахью.

По существу, это свойство было замечено в работе Р. Донахью [4].

Замечание 7. Росток — возможность разбить ребро, то есть он является формальным атрибутом ребра. Поэтому тот факт, что под действием преобразования Донахью росток иногда перемещается не так, как ребро, на котором он расположен (перемещение ростка нетрудно отследить, сравнивая результаты преобразований левого и правого деревьев (рис. 13)), или, иначе говоря, что росток может перемещаться с одного ребра на другое, удивителен.

Аналогично предыдущему, можно выделить 28 типов внутренних ростков (рис. 14) и описать соответствующий этой классификации карусельный эффект (рис. 15), который по сложности заметно превосходит все предыдущие.

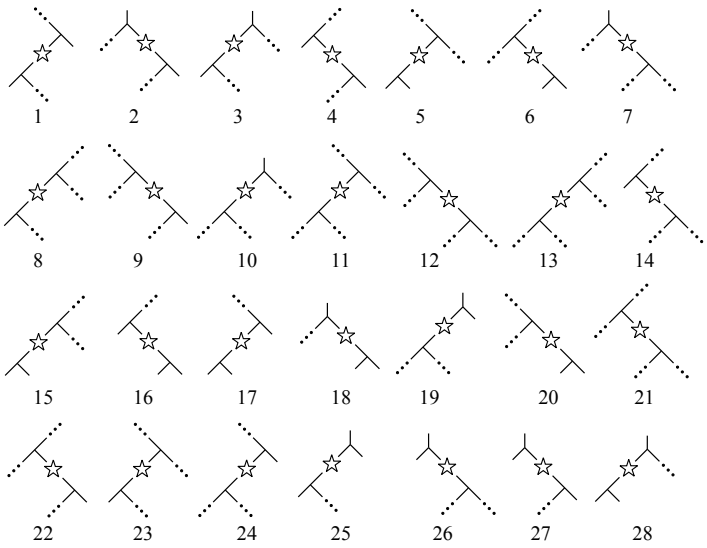

Рис. 14. Классификация внутренних ростков

Замечание 8. Существует дерево, один из внутренних ростков которого в процессе итераций преобразования Донахью проходит через все 28 типов (рис. 16).

Замечание 9. В графе, описывающем карусельный эффект, существует маршрут, по которому не проходит ни один росток, например, $28 \to 16 \to 17 \to 18 \to 25 \to$ $\to 26$.

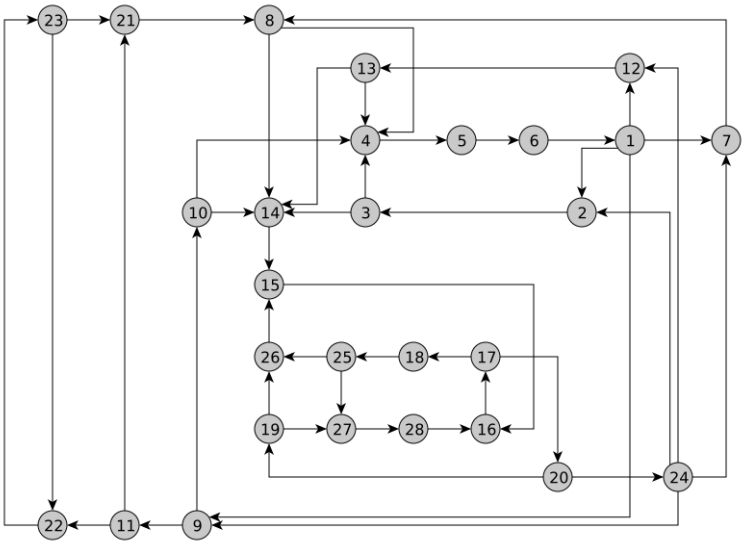

Рис. 15. Карусельный эффект для внутренних ростков

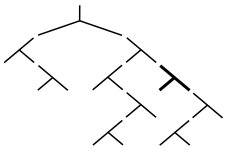

Рис. 16. Росток, проходящий через все типы

5. Ручные компоненты: большие семейства коротких орбит

Естественный вопрос, касающийся преобразования Донахью, — как растёт количество орбит с ростом размера деревьев. С этим вопросом тесно связан вопрос о причинах заметного преобладания (по крайней мере, для деревьев небольшого размера) циклов длины 6.

Первый результат такого типа получен ещё до выхода основополагающей работы [4]: Л. Шапиро в [10] построил семейство орбит длины 6 преобразования Донахью, количество которых растёт как $\Theta(2^n)$ (в обозначениях [11]). Пример такой орбиты приведён на рис. 17. Назовём эти семейства *шапировскими* семействами циклов.

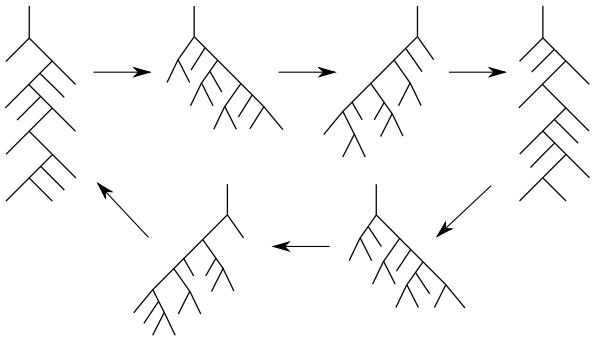

Рис. 17. Пример шапировского цикла

Из отмеченного факта тривиально следует

Теорема 7. Количество циклов преобразования Донахью для кубических деревьев с n листьями растёт не медленнее чем $\Theta(2^n)$.

Оценка теоремы 7 улучшает оценку [3]. Авторам не удалось улучшить оценку теоремы 7, однако некоторый прогресс в этом направлении всё же достигнут.

Во-первых, удалось построить не являющееся шапировским семейство циклов длины 6, следовательно, не все циклы длины 6 являются шапировскими. Структура такого цикла показана на рис. 18. Деревья этого цикла состоят из зигзагов, выражения показывают количества триад в зигзагах, $k, l, m \in \mathbb{N}$. Количество таких циклов растёт как $\Theta(n^2)$, где n — количество триад.

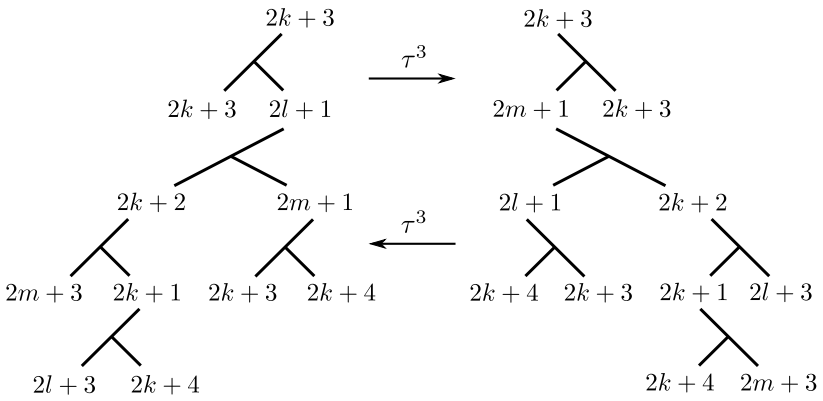

Рис. 18. Пример нешапировского цикла

Во-вторых, удалось построить семейство циклов длины 9, количество которых растёт как $\Theta(2^{n/2})$. Типичный элемент этого семейства приведён на рис. 19 (здесь выражения соответствуют числу триад в выделенных зигзагах и двухуровневых гребёнках). Дерево, изображённое слева, устроено следующим образом: его основу составляет правый зигзаг первого рода (количество триад не меньше 5); L — множество триад зигзага, имеющих левых сыновей. К некоторым триадам из L справа присоединены двухуровневые гребёнки, при этом хотя бы одна гребёнка должна быть добавлена.

Полученному дереву, состоящему из $2m + 1$ триады, можно поставить в соответствие композицию числа m. Количество слагаемых в композиции равно числу триад множества L, если гребёнка состоит из $2k$ триад, то соответствующее ей слагаемое равно $k + 1$ (если гребёнка отсутствует, то слагаемое равно 1). Так, дереву на рис. 19 слева соответствует разложение числа 15 в упорядоченную сумму $4+1+4+1+1+3+1$. Оценка количества рассматриваемых орбит следует из того, что количество композиций числа m растёт как $\Theta(2^m)$.

Приведённые семейства являются типичными примерами того, что естественно назвать *ручными компонентами* преобразования Донахью — семействами деревьев, для которых орбита (или, более общо, её часть) могут быть явно вычислены. Важность ручных компонент связана с тем, что неконструктивных способов доказательства существования семейств орбит в настоящее время, по-видимому, нет.

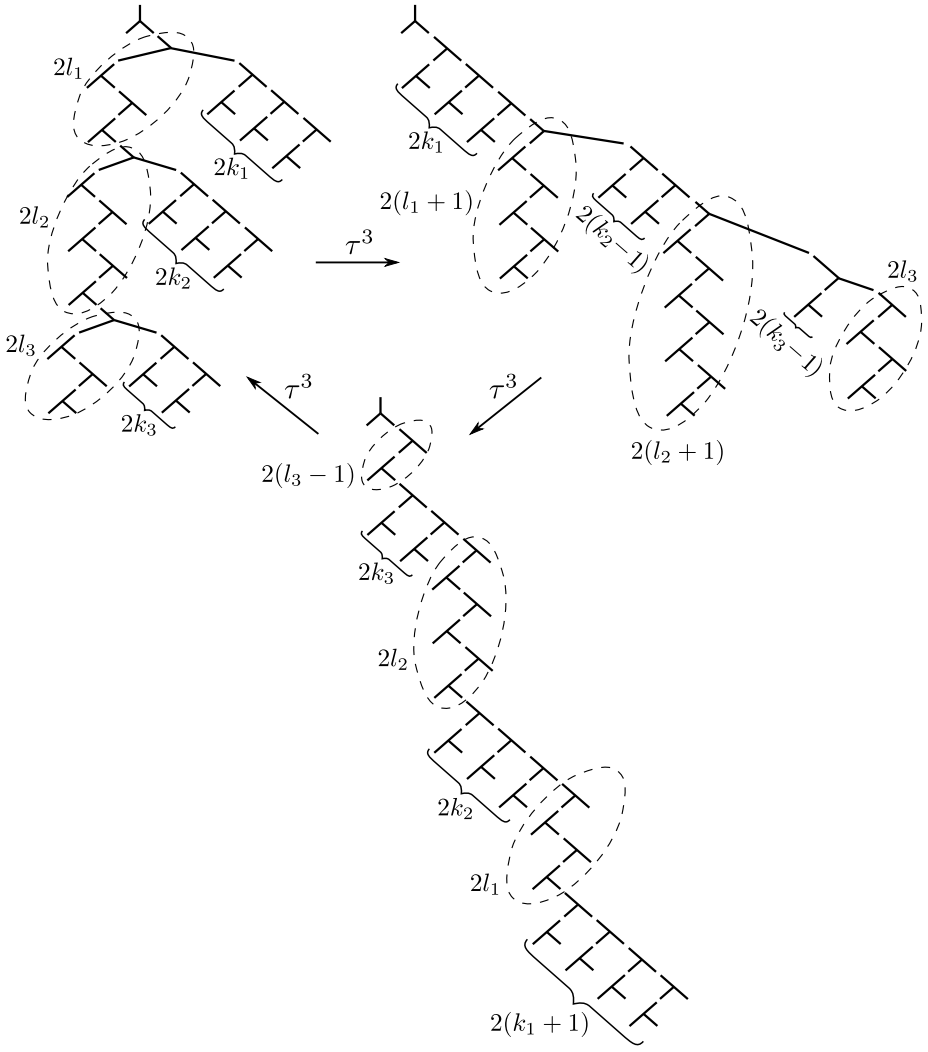

Рис. 19. Экспоненциальное семейство циклов длины 9

6. Ручные компоненты: длинные орбиты

Двойственным к вопросу о количестве орбит является вопрос о том, насколько длинными могут быть орбиты преобразования Донахью. Этот вопрос является значительно более трудным. Эмпирические данные демонстрируют наличие очень длинных циклов, однако очень длинные циклы по природе своей не могут быть ручными компонентами и в рамках обсуждаемого в данной работе элементарного подхода не могут быть построены. Соответственно результаты делятся на слабые (ручные семейства орбит, длины которых растут быстрее, чем линейная функция, но всё-таки как полиномы небольшой степени) и фрагментарные (ручные семейства незамкнутых фрагментов орбит). Результаты второго типа — единственная (не реализованная на данный момент) надежда построить из этих фрагментов достаточно длинные орбиты в рамках элементарного подхода.

В [3] приводится теорема об окаймлении и показано, как из неё следует существование экспоненциального семейства орбит, длины которых растут как $\Theta(n^{\log_2 3})$ от числа листьев. *Окаймлением дерева* называется процедура расцепления всех его листьев на два листа следующего уровня (рис. 20).

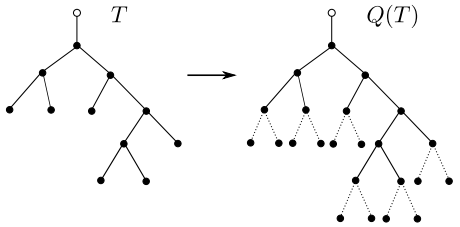

Рис. 20. Окаймление дерева

Символом $\Delta(T)$ обозначим длину орбиты дерева T. Имеет место

Теорема 8 (об окаймлении [3]).

1) Если $\Delta(T)$ нечётна, то $\Delta(Q(T)) = 3\Delta(T)$.
2) Если $\Delta(T)$ чётна, то $\Delta(Q(T)) = \frac{3}{2}\Delta(T)$.

Из теоремы 8 тривиально следует, что каждому дереву можно сопоставить семейство его последовательных окаймлений, длины орбит которых, начиная с некоторого места, начинают расти как $\Theta(3^k)$, где k — количество окаймлений. Остаётся заметить, что при окаймлении дерева количество его вершин удваивается.

С другой стороны, построено семейство деревьев с n триадами, длины орбит которых растут как $\Theta(n^2)$. Представитель этого семейства изображён на рис. 21 слева (выражения соответствуют числу триад в выделенных фрагментах). Данное дерево состоит из двух зигзагов, присоединённых к корневой триаде, в одном из зигзагов реализован внутренний росток (выделен на рисунке полужирным). Если бы этот росток не был реализован, то длина орбиты дерева была бы $72k + 24$, в этом можно убедиться путём непосредственного моделирования. При реализации ростка каждые $72k + 24$ преобразования происходит перемещение ростка по циклу, изображённому на рис. 21 справа. Таким образом, получаем квадратичный рост длины орбиты дерева.

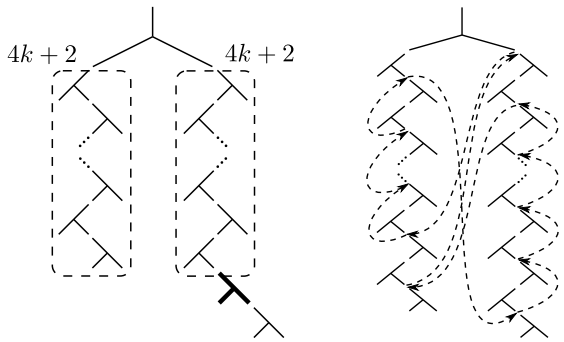

Рис. 21. Деревья, длины орбит которых растут как $\Theta(n^2)$

Заключение

Преобразование Донахью кажется достаточно сложным, чтобы на его примере можно было бы сформулировать подходы, полезные в комбинаторной динамике вообще. В этом смысле его можно сравнить с игрой «Жизнь» или назвать комбинаторным аналогом турбулентности. Оборотной стороной этой привлекательной сложности является то, что большинство исследователей (начиная с Р. Донахью и Л. Шапиро), обнаружив данное преобразование, ограничиваются констатацией только нескольких простых его свойств и переходят к рассмотрению других объектов. Авторы данной работы, напротив, уверены, что изучение этого преобразования возможно. Приведённые

результаты выглядят несколько разрозненными, однако они являются начальной стадией формирования техники, которая может позволить получить существенные продвижения в изучении двух ключевых вопросов: оценке количества орбит и построению очень длинных орбит.

ЛИТЕРАТУРА

1. *Стенли Р.* Перечислительная комбинаторика. Деревья, производящие функции и симметрические функции. М.: Мир, 2005. 768 с.
2. *Татт У.* Теория графов. М.: Мир, 1988. 424 с.
3. *Пушкарев И. А., Бызов В. А.* Преобразование Донахью: элементарный подход // Записки научных семинаров ПОМИ. 2013. Т. 411. С. 148–177.
4. *Donaghey R.* Automorphisms on Catalan trees and bracketings // J. Combinatorial Theory. Ser. B. 1980. V. 29. No. 1. P. 75–90.
5. *Кнут Д.* Искусство программирования. Т. 4А. Комбинаторные алгоритмы. Ч. 1. М.: Вильямс, 2016. 960 с.
6. *Callan D.* A bijection on Dyck paths and its cycle structure // Electronic J. Combinatorics. 2007. V. 14. No. 1. P. R28.
7. `www.oeis.org/A080070` — The On-Line Encyclopedia of Integer Sequences. 2003.
8. *Пушкарев И. А.* Об одном преобразовании плоских деревьев // Математический вестник педвузов и университетов Волго-Вятского региона. 2006. № 8. С. 92–99.
9. *Bergeron F., Labelle G., and Leroux P.* Combinatorial Species and Tree-like Structures. N.Y.: Cambridge University Press, 1997. 457 p.
10. *Shapiro L. W.* The cycle of six // The Fibonacci Quarterly. 1979. V. 17. No. 3. P. 253–259.
11. *Грэхем Р., Кнут Д., Паташник О.* Конкретная математика. Основание информатики. М.: Мир, 1998. 703 с.

REFERENCES

1. *Stanley R. P.* Enumerative Combinatorics: Vol. 2. N.Y., Cambridge University Press, 1999. 585 p.
2. *Tutte W. T.* Graph Theory. California, Addison-Wesley Publishing Company, 1984. 333 p.
3. *Pushkarev I. A. and Byzov V. A.* Donaghey's transformation: an elementary approach. J. Math. Sci., 2014, vol. 196, no. 5. pp. 199–215.
4. *Donaghey R.* Automorphisms on Catalan trees and bracketings. J. Combinatorial Theory, Ser. B, 1980, vol. 29, no. 1, pp. 75–90.
5. *Knuth D. E.* The Art of Computer Programming, Vol. 4A: Combinatorial Algorithms, P. 1. New Jersey, Addison-Wesley Professional, 2011. 883 p.
6. *Callan D.* A bijection on Dyck paths and its cycle structure. Electronic J. of Combinatorics, 2007, vol. 14, no. 1, pp. R28.
7. `www.oeis.org/A080070` — The On-Line Encyclopedia of Integer Sequences. 2003.
8. *Pushkarev I. A.* Ob odnom preobrazovanii ploskikh derev'yev [On a transformation of plane trees]. Matematicheskiy Vestnik Pedvuzov i Universitetov Volgo-Vyatskogo Regiona, 2006, no. 8, pp. 92–99. (in Russian)
9. *Bergeron F., Labelle G., and Leroux P.* Combinatorial Species and Tree-like Structures. N.Y., Cambridge University Press, 1997. 457 p.
10. *Shapiro L. W.* The cycle of six. Fibonacci Quarterly, 1979, vol. 17, no 3. pp. 253–259.
11. *Graham R. L., Knuth D. E., and Patashnik O.* Concrete Mathematics: A Foundation for Computer Science. New Jersey, Addison-Wesley Professional, 1994. 672 p.

МАТЕМАТИЧЕСКИЕ МЕТОДЫ КРИПТОГРАФИИ

UDC 519.7 DOI 10.17223/20710410/44/3

CRYPTANALYTIC CONCEPT OF FINITE AUTOMATON INVERTIBILITY WITH FINITE DELAY[1]

G. P. Agibalov

National Research Tomsk State University, Tomsk, Russia

E-mail: agibalov@mail.tsu.ru

The automaton invertibility with a finite delay plays a very important role in the analysis and synthesis of finite automata cryptographic systems. The automaton cryptanalitic invertibility with a finite delay τ is studied in the paper. From the cryptanalyst's point of view, this notion means the theoretical possibility for recovering, under some conditions, a prefix α of a length n in an unknown input sequence $\alpha\delta$ of an automaton from its output sequence γ of the length $n+\tau$ and perhaps an additional information such as parameters τ and n, initial (q), intermediate (θ) or final (t) state of the automaton or the suffix δ of the length τ in the input sequence. The conditions imposed on the recovering algorithm require for prefix α to be arbitrary and may require for the initial state q and suffix δ to be arbitrary or existent, that is, the variable α is always bound by the universal quantifier and each of variables q and δ may be bound by any of quantifiers — universal $(\forall)$ or existential $(\exists)$ one. The variety of information, which can be known to a cryptanalyst, provides many different types of the automaton invertibility and, respectively, many different classes of invertible automata. Thus, in the paper, an invertibility with a finite delay τ of a finite automaton A is the ability of this automaton to resist recovering or, on the contrary, to allow precise determining any input word α of a length n for the output word γ being the result of transforming by the automaton A in its initial state q the input word $\alpha\delta$ with the δ of length τ and with the known n, τ, A, γ and $\upsilon \subseteq \{\delta, q, \theta, t\}$ where q and δ may be arbitrary or some elements in their sets and θ and t are respectively intermediate and final states of A into which A comes from q under acting of input words α and $\alpha\delta$ respectively. According to this, the automaton A is called invertible with a delay τ if there exists a function $f(\gamma, \upsilon)$ and a triplet of quantifiers $\varkappa \in \{Q_1x_1Q_2x_2Q_3x_3 : Q_ix_i \in \{\forall q, \exists q, \forall\alpha, \forall\delta, \exists\delta\}, i \neq j \Rightarrow x_i \neq x_j\}$ such that $\varkappa[f(\gamma, \upsilon) = \alpha]$; in this case f is called a recovering function, $(\varkappa, \upsilon)$ — an invertibility type, $\varkappa$ — an invertibility degree, υ — an invertibility order of the automaton A and $\exists f\varkappa[f(\gamma, \upsilon) = \alpha]$ — an invertibility condition of type $(\varkappa, \upsilon)$ for the automaton A. So, 208 different types of the automaton A invertibility are defined at all. The well known types of (strong) invertibility and weak invertibility described for finite automata earlier by scientists (D. A. Huffman, A. Gill, Sh. Even, A. A. Kurmit, Z. D. Dai, D. F. Ye, K. Y. Lam, R. Tao and many others) in our theory belong to types $(\forall q\forall\alpha\forall\delta, \varnothing)$ and $(\forall q\forall\alpha\forall\delta, \{q\})$ respectively. For every invertibility type, we have defined a class of automata with this type of invertibility and described the inclusion relation on the set of all these classes. It has turned out that the graph of this relation is the union of twenty nine lattices with thirteen of them each containing sixteen

[1]The author is supported by the RFBR-grant no. 17-01-00354.

classes and sixteen lattices each containing thirteen classes. To solve the scientific problems (invertability tests, synthesis of inverse automata and so on) related to the different and concrete invertibility classes, we hope to continue these investigations.

Keywords: *finite automata, information-lossless automata, automata invertibility, cryptanalytic invertibility.*

Introduction

In the theory of analysis and synthesis of finite automaton cryptosystems, the invertibility property of finite automata takes the most important place. From cryptanalytic point of view, it means the theoretical possibility to recover a nonempty part of input word of an automaton using its output word and, possibly, some additional information about the automaton — about its transition and output functions, about its states — initial, intermediate or final, about its class, etc and about the rest of input word playing an auxiliary (often — official) role — about its length, value and location in the input word. A variety of this information kinds generates the different types of the automaton invertibility and, respectively, the different classes of the invertible automata. In this paper, we assume that the transition and output functions of the automata under consideration are completely known, a nonempty prefix of an input word need be recovered so that the length of the next part of the word following after the prefix and called the recovering or invertibility delay are also known.

So under the invertibility with a finite delay τ of a finite automaton A, we understand the property of A which allows uniquely compute its any input word α using an output word γ produced by the automaton A in an initial state q as its reaction to an input word $\alpha\delta$ with δ of the length τ, with the known τ, A, γ, and with the unknown, possibly, some or all the values from the list $\upsilon \subseteq \{\delta, q, \theta, t\}$, where q and δ can be arbitrary or some elements of their sets, θ and t are, respectively, intermediate and final states of the automaton A, into which it comes from q under the influence of input words α and $\alpha\delta$ respectively.

According to this, the automaton A is called *invertible* with the delay τ if there exist a function $f(\gamma, \upsilon)$ and a triplet of quantifiers $\varkappa \in \{Q_1x_1Q_2x_2Q_3x_3 : Q_ix_i \in \in \{\forall q, \exists q, \forall\alpha, \forall\delta, \exists\delta\}, i \neq j \Rightarrow x_i \neq x_j\}$ such that $\varkappa(f(\gamma, \upsilon) = \alpha)$; in this case, f is called recovering function, $(\varkappa, \upsilon)$ — invertibility type, $\varkappa$ — invertibility degree, υ — invertibility order of the automaton A and $\exists f\varkappa(f(\gamma, \upsilon) = \alpha)$ — invertibility condition for type $(\varkappa, \upsilon)$.

In the automata theory, a notion of information lossless automaton (ILA) are often used as a synonym to a notion of an invertible automaton. For the first time, ILAs were investigated by D. A. Huffman [1, 2] (his results can be also found in the monograph by A. Gill [3]), later — by Sh. Even [4] and also by A. A. Kurmit, who has described his own results on ILA in the detailed monograph [5] where ILA with a finite delay is considered with the known initial or final state and is called there ILA of a finite order, respectively, of I or of II type. In 1959, A. D. Zakrevsky [6] has proposed a symmetric cipher on the base of a strongly connected ILA with zero delay (with an output function being bijective for every state). For the sake of fairness, we need to say that first the similar automata were used by the Japanese during World War Two in their ciphering machine known as Purple [7]. Recently, the automaton invertibility became a research subject for Chinese scientists headed by professor R. Tao. They have produced FAPKC — Finite Automaton Public Key Cryptosystems based on memory finite automata which are invertible with finite delay and with known initial state [7 – 9].

The main results of the works enumerated above and related to the automaton invertibility with a finite delay are in reality the definitions and constructive tests of two types of invertibility — strong and weak (our types $(\forall q \forall \alpha \forall \delta, \varnothing)$ and $(\forall q \forall \alpha \forall \delta, \{q\})$ respectively) and algorithms for synthesis of inverse automata for them. These types of invertibility are really defined in the mentioned works through the automaton properties (classes) and afterwards it is proved that if an automaton belongs to a certain class, then the recovering its input word prefixes is possible. This looks like "a cart ahead of horse".

In our paper, a general definition for an arbitrary type of finite automaton invertibility is introduced. Every particular type of invertibility is obtained from this definition by setting particular values of the definition parameters which are the degree $\varkappa$ and the order υ of the invertibility. So, formally, 208 types of finite automaton invertibility with a delay are introduced in all, including types of strong and weak invertibility mentioned above from [3, 5, 9]. For every type of automaton invertibility with a fixed delay, we define the class of all finite automata invertible of this type and show that the set of all these classes partially ordered by the inclusion relation is the union of 29 lattices. The definition of the arbitrary type as well as of each particular type of an automaton invertibility is given in a completely clear and natural way, namely through the existence of a function recovering the unknown prefix of an automaton input word by using another known information. As for constructive tests for automaton invertibility of each type, they are supposed to be formulated and proved in terms of the automaton itself properties. A consequence of this fact is that the definitions of strong and weak invertibilities in monographs [3, 5, 9] are theorems in our theory. Besides, we have succeeded in defining and researching many such automaton invertibility types and classes which are not studied by other scientists. Of course, we don't exclude that not all these classes are of high importance from science point of view, but the only existence of them induces people to thorough studying them for the purpose of solving some theoretical and applied problems, including establishment of necessary and sufficient conditions for automaton invertibility of each type; building up a constructive test for belonging a finite automaton to an invertibility class; algorithmic synthesis of the automata in a given invertibility class; characterization of the invertible automata, to which inverse automata exist, and algorithmic synthesis of the last; development of the effective algorithms for recovering word prefix on the input of an invertible automaton in a particular invertibility class; creation of private and public key cryptosystems on the basis of invertible automata of different invertibility classes; algorithmic cryptanalysis of these cryptosystems.

The solutions of these problems and their research in computer experiments are supposed to perform by the author and his colleagues for several future years with regular publications of the results in the journal "Prikladnaya Diskretnaya Matematika" and their presentation on the International Conference "Computer Security and Cryptography" — SIBECRYPT.

1. Agreements

An arbitrary finite automaton is presented as $A = (X, Q, Y, \psi, \varphi)$, where X, Q and Y are its input alphabet, the set of states and the output alphabet respectively; ψ and φ — its functions, respectively, of transitions and outputs, $\psi : X \times Q \to Q$ and $\varphi : X \times Q \to Y$. The functions, being defined for pairs $xq \in X \times Q$, we extend to pairs $\alpha q \in X^* \times Q$ by induction on the length $|\alpha|$ of the word $\alpha \in X^*$, namely the functions $\psi : X^* \times Q \to Q$ and $\bar{\varphi} : X^* \times Q \to Y^*$ are defined as $\psi(\Lambda, q) = q$, $\psi(\alpha\beta, q) = \psi(\beta, \psi(\alpha, q))$, $\bar{\varphi}(\Lambda, q) = \Lambda$, $\bar{\varphi}(x, q) = \varphi(x, q)$ and $\bar{\varphi}(\alpha\beta, q) = \bar{\varphi}(\alpha, q)\bar{\varphi}(\beta, \psi(\alpha, q))$. Here and everywhere further, the symbol Λ denotes the empty word in any alphabet.

Thus, $\psi(\alpha, q)$ is a state, into which the automaton A comes from a state q under the influence of input word α, and $\bar{\varphi}(\alpha, q)$ is an output word which the automaton produces this time. The function $\varphi_q : X \to Y$, defined as $\varphi_q(x) = \varphi(x, q)$ for all $x \in X$, is called the output function of the automaton A in the state $q \in Q$.

We don't exclude partially defined automata from the consideration. For presentation of the information in them, we use the symbol ϖ, regarding it as any word of any length and over any alphabet. So the record $\varpi \in X^n$, for example, means that ϖ is a word of a length n in the alphabet X and the record $f(a) = \varpi$ — that a function value $f(x)$ is not defined for $x = a$. In comparison between two words in the same alphabet, we consider that the word ϖ equals a word α iff $|\varpi| = |\alpha|$. In particular, two words ϖ coincide iff their lengths are equal.

Further, we adopt the convention for any logical formula

$$F = Q_1 x_1 Q_2 x_2 \dots Q_n x_n P(x_1, x_2, \dots, x_n),$$

where $Q_1, Q_2, \dots, Q_n$ are the symbols of quantifiers $\forall, \exists$ and the formula P doesn't contain quantifiers, to say that a n-tuple $c_1 c_2 \dots c_n$ of values of variables $x_1, x_2, \dots, x_n$ satisfies F if, for each $i = 1, 2, \dots, n$, the value c_i is chosen in the range of the variable x_i in the following way: in case $Q_i = \forall$ — anyhow, in case $Q_i = \exists$ — (in dependence on already chosen c_j for $j < i$) so that $P(c_1, c_2, \dots, c_n) = \texttt{true}$. By the definition of the truth of F, such a tuple exists if and only if $F = \texttt{true}$.

Finally, everywhere further, by the symbol τ we denote a non-negative integer called a delay and, in the absence of additional remarks, it is supposed that $a \in X$, $b \in X$, $\alpha \in X^*$, $\beta \in X^*$, $\delta \in X^\tau$, $\varepsilon \in X^\tau$, $q \in Q$, $s \in Q$.

2. Definition of invertibility with finite delay

Consider a finite automaton $A = (X, Q, Y, \psi, \varphi)$. Let q, α, δ be variables with values in Q, X^*, X^τ denoting, respectively, an initial state, a prefix (beginning) and suffix (ending) of an input word $\alpha\delta$ of the automaton A and $K = \{\forall q, \forall\alpha, \forall\delta, \exists q, \exists\delta\}$ be the set of universal and existential quantifiers which bind these variables. In reality, the quantifiers in K are $\forall q \in Q$, $\forall\alpha \in X^*$, $\forall\delta \in X^\tau$, $\exists q \in Q$, $\exists\delta \in X^\tau$ without previously fixed symbols indicating ranges of variables in question and omitted in K for conciseness of record. Besides, notice that K doesn't contain the quantifier $\exists\alpha$. This is because, for a cryptanalyst, the input word α can be any one.

Also let $\theta = \psi(\alpha, q)$, $t = \psi(\alpha\delta, q)$ and $V = \{\Lambda, q, \theta, t, \delta, q\theta, qt, q\delta, \theta t, \theta\delta, t\delta, q\theta t, q\theta\delta, qt\delta, \theta t\delta, q\theta t\delta\}$. It is seen that symbols θ and t denote an intermediate and final states, into which the automaton A comes from the state q after having received on its input the words α and $\alpha\delta$ respectively. The members of the set V are meant for describing what we call here an invertibility order of the automaton A. In fact, they are some functions in q, α, δ.

We say that the automaton A is *invertible with the delay* τ if there exist quantifiers K_1, K_2, K_3 in K with different variables from $\{q, \alpha, \delta\}$ as well as a function $f : Y^* \times V \to X^*$ and a tuple $\upsilon(q, \alpha, \delta) \in V$ such that the following formula is true

$$\Phi = K_1 K_2 K_3 (f(\bar{\varphi}(\alpha\delta, q), \upsilon(q, \alpha, \delta)) = \alpha);$$

in this case, $(K_1 K_2 K_3, \upsilon)$ is called *invertibility type* of the automaton A, $K_1 K_2 K_3$ — *invertibility degree*, υ — *invertibility order*, f — *recovering function* (for input prefix), τ — *recovery delay*, or *invertibility delay* and $\exists f[\Phi]$ — *invertibility condition* of this type for the automaton A.

Taking into account the commutativity of the same type quantifiers, in the table for the automaton A, we present invertibility conditions of all possible invertibility types with the delay τ. From this table, for an invertibility of a type $(K_1K_2K_3, \upsilon)$, the invertibility condition is obtained by attaching the quantifier prefix $\exists f K_1K_2K_3$ from the left column to the (so called) underlying expression $f(\bar{\varphi}(\alpha\delta, q), \upsilon(q, \alpha, \delta)) = \alpha$ from the right column with the proper invertibility order υ. Further, the invertibility condition with the quantifier prefix of a number i and its underlying expression of a number j in the table is denoted by $U_{i,j}$ or (if you need to know τ) $U_{i,j}[\tau]$. For example, $U_{1,1}[\tau] = \exists f \forall q \forall \alpha \forall \delta (f(\bar{\varphi}(\alpha\delta, q)) = \alpha)$, $U_{1,2}[\tau] = = \exists f \forall q \forall \alpha \forall \delta (f(\bar{\varphi}(\alpha\delta, q), q) = \alpha)$, $U_{5,10}[\tau] = \exists f \exists q \forall \alpha \exists \delta (f(\bar{\varphi}(\alpha\delta, q), \psi(\alpha, q), \delta) = \alpha)$ and so on. Thus, for any finite automaton, we have formally defined $208 (= 13 \cdot 16)$ invertibility types with any finite delay. But later, we will see that, for some of these types with different invertibility orders, the invertibility conditions can be equivalent and define the same type of invertible automata.

Conditions for different types of invertibility with a delay τ of the automaton A

No	Quantifier prefix $\exists f Q_1x_1Q_2x_2Q_3x_3$	No	Underlying expression $f(\bar{\varphi}(\alpha\delta, q), \upsilon(q, \alpha, \delta)) = \alpha$
		1	$f(\bar{\varphi}(\alpha\delta, q)) = \alpha$
		2	$f(\bar{\varphi}(\alpha\delta, q), q) = \alpha$
		3	$f(\bar{\varphi}(\alpha\delta, q), \psi(\alpha, q)) = \alpha$
1	$\exists f \, \forall q \, \forall \alpha \, \forall \delta$	4	$f(\bar{\varphi}(\alpha\delta, q), \psi(\alpha\delta, q)) = \alpha$
2	$\exists f \, \forall q \, \forall \alpha \, \exists \delta$	5	$f(\bar{\varphi}(\alpha\delta, q), \delta) = \alpha$
3	$\exists f \, \forall q \, \exists \delta \, \forall \alpha$	6	$f(\bar{\varphi}(\alpha\delta, q), q, \psi(\alpha, q)) = \alpha$
4	$\exists f \, \exists q \, \forall \alpha \, \forall \delta$	7	$f(\bar{\varphi}(\alpha\delta, q), q, \psi(\alpha\delta, q)) = \alpha$
5	$\exists f \, \exists q \, \forall \alpha \, \exists \delta$	8	$f(\bar{\varphi}(\alpha\delta, q), q, \delta) = \alpha$
6	$\exists f \, \exists q \, \exists \delta \, \forall \alpha$	9	$f(\bar{\varphi}(\alpha\delta, q), \psi(\alpha, q), \psi(\alpha\delta, q)) = \alpha$
7	$\exists f \, \forall \alpha \, \exists q \, \forall \delta$	10	$f(\bar{\varphi}(\alpha\delta, q), \psi(\alpha, q), \delta) = \alpha$
8	$\exists f \, \forall \alpha \, \exists q \, \exists \delta$	11	$f(\bar{\varphi}(\alpha\delta, q), \psi(\alpha\delta, q), \delta) = \alpha$
9	$\exists f \, \forall \alpha \, \forall \delta \, \exists q$	12	$f(\bar{\varphi}(\alpha\delta, q), q, \psi(\alpha, q), \psi(\alpha\delta, q)) = \alpha$
10	$\exists f \, \forall \alpha \, \exists \delta \, \forall q$	13	$f(\bar{\varphi}(\alpha\delta, q), q, \psi(\alpha, q), \delta) = \alpha$
11	$\exists f \, \forall \delta \, \exists q \, \forall \alpha$	14	$f(\bar{\varphi}(\alpha\delta, q), q, \psi(\alpha\delta, q), \delta) = \alpha$
12	$\exists f \, \exists \delta \, \forall q \, \forall \alpha$	15	$f(\bar{\varphi}(\alpha\delta, q), \psi(\alpha, q), \psi(\alpha\delta, q), \delta) = \alpha$
13	$\exists f \, \exists \delta \, \forall \alpha \, \exists q$	16	$f(\bar{\varphi}(\alpha\delta, q), q, \psi(\alpha, q), \psi(\alpha\delta, q), \delta) = \alpha$

Having a function $f : Y^* \times V \to X^*$, we can define a function $f' : Y^* \times V \to X^*$ so that $f'(\gamma y, \upsilon) = f(\gamma, \upsilon)$ for all $\gamma \in Y^*$, $y \in Y$ and $\upsilon \in V$. Since $\bar{\varphi}(\alpha\delta x, q) = = \bar{\varphi}(\alpha\delta, q)\varphi(x, \psi(\alpha\delta, q))$, the equality $f(\bar{\varphi}(\alpha\delta, q), \upsilon) = \alpha$ implies the equality $f'(\bar{\varphi}(\alpha\delta x, q), \upsilon) = = \alpha$. By the principle of mathematical induction, this implication proves that if, for a type of invertibility, a finite automaton is invertible with a finite delay, then, for the same type of invertibility, the automaton is invertible with any greater integer delay.

In this work, by the invertibility, we only understand an invertibility of a finite automaton, of a certain type, of a certain order, and of a finite delay and usually don't mention these its attributes without a particular need.

3. Invertibility classes

For any $i \in \{1, 2, \ldots, 13\}$ and $j \in \{1, 2, \ldots, 16\}$, we say that an automaton $A = = (X, Q, Y, \psi, \varphi)$ belongs to an (*invertibility*) *class* $C_{i,j}[\tau]$ if the condition $U_{i,j}[\tau]$ is true; in this case, the condition $U_{i,j}[\tau]$ is called the *invertibility condition in the class* $C_{i,j}[\tau]$ of the automaton A. The purpose of this paragraph is the description of the inclusion relation on the set of all invertibility classes with a particular delay, following from the property: if $U_{i,j}[\tau] \Rightarrow U_{k,l}[\tau]$, then $C_{i,j}[\tau] \subseteq C_{k,l}[\tau]$. There are two cases when the premise

$U_{i,j}[\tau] \Rightarrow U_{k,l}[\tau]$ in this property takes place and this fact is recognized immediately by the invertibility types $(K_1K_2K_3, \upsilon)$ in $U_{i,j}[\tau]$ and $(K_1'K_2'K_3', \upsilon')$ in $U_{k,l}[\tau]$:

1) $i = k$, $j \neq l$ and all the elements in υ are contained in υ';
2) $i \neq k$, $j = l$ and $K_1K_2K_3P(q, \alpha, \delta) \Rightarrow K_1'K_2'K_3'P(q, \alpha, \delta)$ for any predicate P in three variables.

For instance, in the first case, $U_{i,5} \Rightarrow U_{i,13}$ and therefore, $C_{i,5} \subseteq C_{i,13}$ for all i and, in the second case, $U_{7,j} \Rightarrow U_{9,j}$ and therefore, $C_{7,j} \subseteq C_{9,j}$ for all j.

In the case 2, truth (or falsehood) of pointed out implication is easy established with the help of identically true formulas of predicate logic such as $\forall x S(x) \Rightarrow \exists x S(x)$, $\exists x \forall y R(x, y) \Rightarrow \forall y \exists x R(x, y)$ and the like.

The implication $U_{i,j}[\tau] \Rightarrow U_{k,l}[\tau]$, connecting the invertibility conditions for two automata, induces the inclusion relation $C_{i,j}[\tau] \subseteq C_{k,l}[\tau]$ between the invertibility classes of these automata. On every of sets $\{C_{i,j}[\tau] : j = 1, 2, \ldots, 16\}$, $i = 1, \ldots, 13$, and $\{C_{i,j}[\tau] : i = 1, 2, \ldots, 13\}$, $j = 1, \ldots, 16$, this relation defines a lattice — a partially ordered set, in which, for every pair of elements, there exist the least upper and the greatest lower bounds. These lattices are shown in the Figs. 1 and 2.

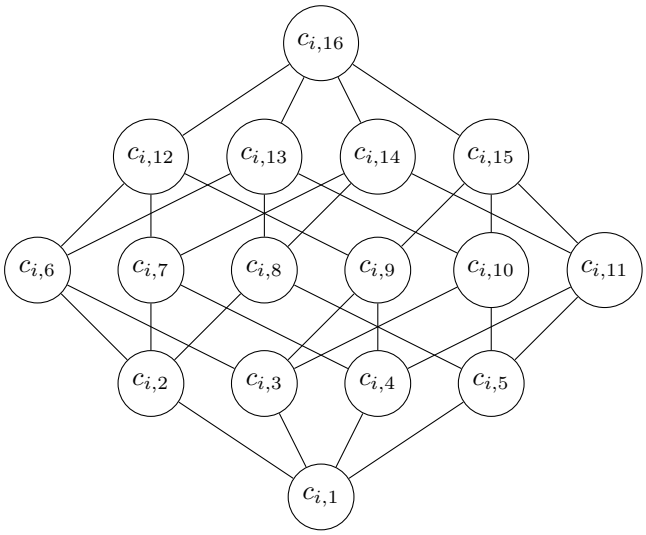

Fig. 1

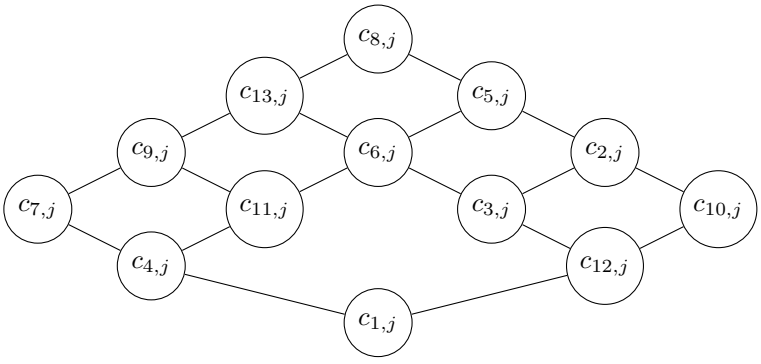

Fig. 2

In the case $\tau = 0$, the sequence δ in quantifier prefixes and underlying expressions in table is the empty word and, as a consequence, all these prefixes and expressions break

up the blocks of equal entities, namely: the first — up the blocks $\bar{1} = \{1, 2, 3, 10, 12\}$, $\bar{4} = \{4, 5, 6, 11\}$ и $\bar{7} = \{7, 8, 9, 13\}$; the second — up the blocks $1' = \{1, 5\}$, $2' = \{2, 8\}$, $3' = \{3, 4, 9, 10, 11, 15\}$ and $6' = \{6, 7, 12, 13, 14, 16\}$. Hence $C_{i,j}[0] = C_{k,j}[0]$ for all $i, k \in I$, $I = \bar{1}, \bar{4}, \bar{7}$ and $j = 1, 2, \ldots, 16$, as well $C_{i,j}[0] = C_{i,l}[0]$ for all $j, l \in J$, $J = 1', 2', 3', 6'$ and $i = 1, 2, \ldots, 13$.

4. Automata invertibility problems

The automaton invertibility conditions which are contained in the definition of this notion for its different types are given in a non-constructive form and it is difficult to apply it in practice. Formulation and correct proof of constructive tests for invertibility of every type is the first in the row of problems related to the cryptanalytic notion of finite automaton invertibility.

From cryptographic point of view, in this row the problem of generating invertible automata of all possible types takes an important place. In different settings of this problem, many different requirements to automata under generation can present — with an equal probability in a certain class, with limited complexity, with great or, on the contrary, little invertibility delay and the like. Its solution seems to be impossible without a proper solution of the first problem.

The notion of finite automaton invertibility under consideration doesn't imply the obligatory existence of an inverse automaton to an invertible automaton. Moreover, it is possible that the function recovering an input prefix can not be finite-automated one for some types of automaton invertibility. In this case evidently the problem appears: given an invertible (of a certain type) automaton, find out whether it has an inverse automaton and if it has, then construct the inverse to it. The solution of this problem in turn implies the definition of inverse to any automaton of every invertibility class. In the absence of inverse automata to the automata of an invertibility class, we have the problem of constructing for them functions recovering prefixes of input sequences under known output sequences.

In subsequent investigations by the author and his colleagues, some of these problems are meant to be solved for some of invertibility classes defined.

5. Invertibility conditions

For investigating the properties of the automaton invertibility, the invertibility condition in its definition need to be re-formulated in a more constructive way and first of all to get out of request for explicit testing the existence of a recovering function. In this section, we present a test (Proposition 1) for automaton invertibility of any type $(\forall q \forall \alpha \forall \delta, \upsilon(q, \alpha, \delta))$ and give some necessary conditions (Proposition 2) for an automaton to be invertible of any type $(Q_1 q Q_2 \alpha Q_3 \delta, \upsilon(q, \alpha, \delta))$ both (test and necessary conditions) without explicit performance of a procedure of testing the existence of a recovering function. The propositions follow from the corresponding auxiliary lemmas about logical formulas. To formulate lemmas, we first introduce some needed symbols.

Let n be a positive integer; $Q_1, \ldots, Q_n$ be symbols of quantifiers, $Q_k \in \{\forall, \exists\}$, $k \in \in \{1, \ldots, n\}$; $x_1, \ldots, x_n, y_1, \ldots, y_n$ be different subject variables and D_i be the range of x_i and y_i for $i \in \{1, \ldots, n\}$. Also let $g(x_1, \ldots, x_n)$ be a function in variables $x_1, \ldots, x_n$ with a range D_g, $k_0 \in \{1, \ldots, n\}$, and $Q_{k_0} = \forall$. Finally, let $f : D_g \to D_{k_0}$ denotes an arbitrary function with the domain D_g and the range D_{k_0}. Consider a logical formula

$$Q_1 x_1 Q_2 x_2 \ldots Q_n x_n (f(g(x_1, x_2, \ldots, x_n)) = x_{k_0}) \tag{1}$$

in the normal form, that is, with the quantifier prefix $Q_1x_1Q_2x_2\ldots Q_nx_n$ and a underlying equality $f(g(x_1,x_2,\ldots,x_n))=x_{k_0}$ without quantifiers.

Lemma 1. In the case $Q_1=\ldots=Q_n=\forall$ the function f with the property (1) exists if and only if

$$\forall x_1\ldots\forall x_n\forall y_1\ldots\forall y_n(x_{k_0}\neq y_{k_0}\Rightarrow g(x_1,\ldots,x_n)\neq g(y_1,\ldots,y_n)). \tag{2}$$

Proof. Necessity. Take any $x_1,\ldots,x_n,y_1,\ldots,y_n$, where $x_{k_0}\neq y_{k_0}$. By the condition (1), $f(g(x_1,x_2,\ldots,x_n))\neq f(g(y_1,y_2,\ldots,y_n))$. Therefore, in view of functionality of f, we obtain $g(x_1,x_2,\ldots,x_n)\neq g(y_1,y_2,\ldots,y_n)$.

Sufficiency. For any $x_1,\ldots,x_n$, let $f(g(x_1,x_2,\ldots,x_n))=x_{k_0}$. This definition of f is correct since, by the condition (2), if for some $x_1,\ldots,x_n,y_1,\ldots,y_n$ the equality $g(x_1,x_2,\ldots,x_n)=g(y_1,y_2,\ldots,y_n)$ holds, then $x_{k_0}=y_{k_0}$. ■

Taking in lemma 1 $n=3$, $x_1=q$, $x_2=\alpha$, $x_3=\delta$, $y_1=s$, $y_2=\beta$, $y_3=\varepsilon$, $g(x_1,\ldots,x_n)=$ $=(\bar\varphi(\alpha\delta,q),\upsilon(q,\alpha,\delta))$, $g(y_1,\ldots,y_n)=(\bar\varphi(\beta\varepsilon,s),\upsilon(s,\beta,\varepsilon))$, and $k_0=2$, $x_{k_0}=\alpha$, $y_{k_0}=\beta$, we get

Proposition 1. The automaton A is invertible of the type $(\forall q\forall\alpha\forall\delta,\upsilon(q,\alpha,\delta))$, that is,

$$\exists f\forall q\forall\alpha\forall\delta(f(\bar\varphi(\alpha\delta,q),\upsilon(q,\alpha,\delta))=\alpha)$$

if and only if

$$\forall q\forall\alpha\forall\delta\forall s\forall\beta\forall\varepsilon(\alpha\neq\beta\Rightarrow(\bar\varphi(\alpha\delta,q),\upsilon(q,\alpha,\delta))\neq(\bar\varphi(\beta\varepsilon,s),\upsilon(s,\beta,\varepsilon)).$$

Lemma 2. For any true quantifier logic formulas in a normal form

$$Q_1z_1\ldots Q_mz_mA(z_1,\ldots,z_m)\ \text{ and }\ R_1z_1\ldots R_mz_mB(z_1,\ldots,z_m),$$

where $Q_i,R_i\in\{\forall,\exists\}$ and $Q_iR_i\neq\exists\exists$ for every $i\in\{1,\ldots,m\}$, there exist some values $c_1,\ldots,c_m$ of variables $z_1,\ldots,z_m$ respectively such that $A(c_1,\ldots,c_m)=B(c_1,\ldots,c_m)=$ $=\texttt{true}$.

Proof. Applying the induction scheme by integer $t\geqslant 1$, we will show that for any such an integer $t\leqslant m$ the equalities $Q_{t+1}z_{t+1}\ldots Q_mz_mA(c_1,\ldots,c_t,z_{t+1},\ldots,z_m)=$ $=R_{t+1}z_{t+1}\ldots R_mz_mB(c_1,\ldots,c_t,z_{t+1},\ldots,z_m)=\texttt{true}$ take place and under $t=m$ we will obtain the state of the lemma.

For $t=0$, the equality under proof is true by the condition. Assuming that it is true for any $t\leqslant j$ where j is an integer and $0\leqslant j<m$, and taking as c_{j+1} any value of the variable z_{j+1} in the case $Q_{j+1}=R_{j+1}=\forall$ and a value of z_{j+1} under which $Q_{j+2}z_{j+2}\ldots Q_mz_mA(c_1,\ldots,c_{j+1},z_{j+2},\ldots,z_m)=\texttt{true}$ or $R_{j+2}z_{j+2}\ldots R_mz_mB(c_1,\ldots,c_{j+1},z_{j+2},\ldots,z_m)=\texttt{true}$ in the case $Q_{j+1}=\exists$ or $R_{j+1}=\exists$ respectively, we obtain that it is also true for $t=j+1$. ■

Lemma 3. For any function g, if there exists a function f with the property (1), then

$$Q_1x_1\ldots Q_nx_nQ_1y_1\ldots Q_ny_n(x_{k_0}\neq y_{k_0}\Rightarrow g(x_1,\ldots,x_n)\neq g(y_1,\ldots,y_n)). \tag{3}$$

Proof. Formulas

$$Q_1x_1\ldots Q_nx_n(f(g(x_1,\ldots,x_n))=x_{k_0}),\quad Q_1y_1\ldots Q_ny_n(f(g(y_1,\ldots,y_n))=y_{k_0})$$

are equivalent. Therefore, by the condition (1)

$$Q_1x_1\dots Q_nx_n(f(g(x_1,\dots,x_n))=x_{k_0})\ \&\ Q_1y_1\dots Q_ny_n(f(g(y_1,\dots,y_n))=y_{k_0}).$$

Hence,

$$Q_1x_1\dots Q_nx_nQ_1y_1\dots Q_ny_n(f(g(x_1,\dots,x_n))=x_{k_0}\ \&\ f(g(y_1,\dots,y_n))=y_{k_0}). \qquad (4)$$

Suppose, the state (3) is false and its negation, that is, the following state is true:

$$Q'_1x_1\dots Q'_nx_nQ'_1y_1\dots Q'_ny_n(x_{k_0}\neq y_{k_0}\ \&\ g(x_1,\dots,x_n)=g(y_1,\dots,y_n)), \qquad (5)$$

where, for any $j\in\{1,\dots,n\}$, the symbol Q'_j is a dual quantifier, namely $\forall'=\exists$ and $\exists'=\forall$. By the lemma 2 related to the formulas (4) and (5), there exist values $a_1,\dots,a_n$ of the variables $x_1,\dots,x_n$ and values $b_1,\dots,b_n$ of variables $y_1,\dots,y_n$ respectively such that $f(g(a_1,\dots,a_n))=a_{k_0}$, $f(g(b_1,\dots,b_n))=b_{k_0}$ and $a_{k_0}\neq b_{k_0}$, $g(a_1,\dots,a_n)=g(b_1,\dots,b_n)$. A contradiction is obtained, namely: from one side, $a_{k_0}\neq b_{k_0}$, from another one, $a_{k_0}=$ $=f(g(a_1,\dots,a_n))=f(g(b_1,\dots,b_n))=b_{k_0}$. ■

Let x_1,x_2,x_3 and y_1,y_2,y_3 be the different variables from the sets $\{q,\alpha,\delta\}$ and $\{s,\beta,\varepsilon\}$ respectively such that if x_i is q, α or δ, then y_i is s, β or ε respectively, $Q_i\in\{\forall,\exists\}$, and if $x_i=\alpha$, then $Q_i=\forall$, $i=1,2,3$.

Proposition 2. If an automaton A is invertible of any type $(Q_1x_1Q_2x_2Q_3x_3,\upsilon(q,\alpha,\delta))$, that is, $\exists fQ_1x_1Q_2x_2Q_3x_3(f(\bar\varphi(\alpha\delta,q),\upsilon(q,\alpha,\delta))=\alpha)$, then

$$Q_1x_1Q_2x_2Q_3x_3Q_1y_1Q_2y_2Q_3y_3(\alpha\neq\beta\Rightarrow(\bar\varphi(\alpha\delta,q),\upsilon(q,\alpha,\delta))\neq(\bar\varphi(\beta\varepsilon,s),\upsilon(s,\beta,\varepsilon)).$$

Proof. The Proposition 2 follows from the Lemma 3 in the same way as the Proposition 1 follows from the Lemma 1. ■

REFERENCES

1. *Huffman D. A.* Canonical forms for information-lossless finite-state logical machines. IRE Trans. Circuit Theory, 1959, vol. 6, Spec. Suppl., pp. 41–59.
2. *Huffman D. A.* Notes on information-lossless finite-state automata. Nuovo Cimento, 1959, vol. 13, Suppl. 2, pp. 397–405.
3. *Gill A.* Introduction to the Theory of Finite-State Machines. N.Y., McGraw-Hill Book Company, 1962. 300 p.
4. *Even Sh.* On information-lossless automata of finite order. IEEE Trans. Electron. Comput., 1965, vol. 14, no. 4, pp. 561–569.
5. *Kurmit A. A.* Information Lossless Automata of Finite Order. N.Y., John Wiley Publ., 1974.
6. *Zakrevskiy A. D.* Metod avtomaticheskoy shifratsii soobshcheniy [The method for messages automatic encryption]. Prikladnaya Diskretnaya Matematika, 2009, no. 2(4), pp. 127–137. (in Russian)
7. *Agibalov G. P.* Konechnye avtomati v kriptografii [Finite automata in cryptography]. Prikladnaya Diskretnaya Matematika. Prilojenie, 2009, no. 2, pp. 43–73. (in Russian)
8. *Dai Z. D., Ye D. F., and Lam K. Y.* Weak invertibility of finite automata and cryptanalysis on FAPKC. LNCS, 1998, vol. 1514, pp. 227–241.
9. *Tao R.* Finite Automata and Application to Cryptography. N.Y., Springer, 2009. 406 p.

МАТЕМАТИЧЕСКИЕ ОСНОВЫ КОМПЬЮТЕРНОЙ БЕЗОПАСНОСТИ

УДК 004.94.056.53

АНАЛИЗ МЕТОДОВ АТРИБУТНОГО РАЗГРАНИЧЕНИЯ ДОСТУПА

М. Н. Калимолдаев, Р. Г. Бияшев, О. А. Рог

Институт информационных и вычислительных технологий, г. Алматы, Республика Казахстан

Приведён аналитический обзор основных моделей и методов разграничения доступа, начиная от традиционных (DAC, MAC, RBAC) и до последних разработок — многочисленных моделей, реализующих атрибутное разграничение доступа (ABAC). Описана разрабатываемая в настоящее время модель типизированного атрибутного разграничения доступа (ТАРД). Сформулированы требования к методам разграничения доступа, обеспечивающие безопасное совместное использование информационных ресурсов как в локальных, так и в глобальных вычислительных средах. Проанализированы достоинства и недостатки существующих моделей ABAC. Показано, что модели ТАРД отвечают поставленным требованиям универсальности, гибкости, удобства администрирования, способствующим обеспечению безопасности разграничения доступа вне зависимости от типа операционной среды.

Ключевые слова: *атрибутное разграничение доступа (ABAC), типизированное атрибутное разграничение доступа (ТАРД), DAC, MAC, RBAC, политика разграничения доступа, язык спецификации, синтаксис, семантика, моделирование.*

DOI 10.17223/20710410/44/4

ANALYSIS OF THE METHODS FOR ATTRIBUTE-BASED ACCESS CONTROL

M. N. Kalimoldayev, R. G. Biyashev, O. A. Rog

Institute of Information and Computational Technologies, Almaty, Republic of Kazakhstan

E-mail: olga@ipic.kz

The paper contains an analytical overview of the basic models and methods for access control from the traditional ones (DAC, MAC, RBAC) to the latest developments — numerous models implementing attribute based access control (ABAC). The model of typed attribute based access control (TAAC) being developed currently is described. The following disadvantages of traditional models are pointed out: identification of entities with unique names; access rights redundancy ("coarse-grained access control"); difficult managing large number of users; operating in closed environments; the inability to use integrated security policies; lack of built-in administration tools. It is found out that to ensure the safe sharing of information resources in both local and global

computing environments, access control models must meet the requirements of universality, flexibility and ease of administration while performing the following tasks: identification of entities by several features for fine-grained access control; design and use of multiple access control policies to implement the "multiple policy" paradigm and adapt the system to work in various environments; administration as a means for dynamic policy modeling and convenient privilege managing a large number of users. The advantages and disadvantages of different types of ABAC models are considered. The advantages are: identification of entities by sets of attributes; "fine-grained access control"; flexibility and expressiveness of model specification languages; the possibility of creating new and modeling traditional methods of access control; relative ease of administration; managing privileges of groups of users. The main disadvantage of ABAC is the complexity of calculating attribute values. It is shown that the TAAC models meet the above requirements and provide the following: "fine-grained access control" by identifying entities with the sets of typed attributes; decrease in complexity and increase in speed of calculations; management privileges of hierarchical groups of subjects and objects; dynamic policy construction; multi-criteria access control.

Keywords: *attribute-based access control (ABAC), typed attribute-based access control (TAAC), DAC, MAC, RBAC, access control policy, specification language, syntax, semantics, modeling.*

Введение

В результате развития новых технологий вычислительные среды эволюционировали от централизованных систем, основанных на механизмах, обеспечивающих их производительность за счёт различных способов параллельного исполнения команд, до распределённых гетерогенных систем, осуществляющих децентрализованное хранение и обработку данных с помощью эффективно взаимодействующих служб и алгоритмов.

На сегодняшний день распределённые вычислительные среды отходят от понятий высокопроизводительных распределённых вычислений в сторону развития виртуального сотрудничества людей и/или организаций, объединённых общими правилами коллективного доступа к определённым вычислительным ресурсам. Методы предоставления доступа к информации становятся сервисно-ориентированными, что позволяет эксплуатировать одни и те же ресурсы в режиме совместного использования и требует обеспечения их защиты путём организации разграничения доступа.

При этом субъекты получают доступ к объектам системы в соответствии с политикой авторизации, определяющей допуск к ресурсам определённых видов согласно предоставленным полномочиям с одновременным запретом различных видов несанкционированного доступа. Актуальность проблемы защиты растёт по мере увеличения объёмов хранимых данных и роста сложности программного обеспечения для их обработки.

Подавляющее большинство приложений снабжаются средствами контроля доступа в той или иной форме. Системы разграничения доступа, будучи важнейшими компонентами систем защиты, наиболее подвержены рискам из-за возможных ошибок в конфигурации политик разграничения доступа.

1. Недостатки традиционных моделей разграничения доступа и пути их преодоления

В процессе реализации защищённого доступа к данным система выполняет задачи идентификации, аутентификации и авторизации. Идентификация заключается в при-

своении субъекту или объекту идентификатора, ассоциируемого с перечнем разрешённых ему действий. Аутентификация является средством для доказательства права использовать идентификатор, выполнять роль или владеть определёнными атрибутами. На основании результатов операций идентификации и аутентификации осуществляется авторизация как способ выполнения политики посредством предоставления или запрета доступа субъекта к объекту [1].

С начала 1970-х годов было разработано много моделей, основными из которых являются модели дискреционного разграничения доступа DAC (Discretional Access Control), мандатного разграничения доступа MAC (Mandatory Access Control) и ролевого разграничения доступа RBAC (Role Based Access Control) [2 – 4].

Задачи идентификации сущностей в этих моделях выполняются путём присвоения субъектам и объектам уникальных имён, ввиду чего модели называют идентификаторными. Доступ субъекта к объекту осуществляется на основе проверки имён или приписанных им ролей согласно заранее определённым системным политикам.

Идентификаторные модели эффективно работают в замкнутых и относительно неизменных системах, с определённым кругом заранее известных пользователей, имеющих доступ к известным сервисам.

Основным недостатком данных моделей является то, что они не учитывают дополнительных параметров разграничения доступа, таких, как сведения о ресурсах, отношения пользователей к ресурсам, динамической информации — времени суток, IP-адресов и т. д., что ведёт к «грубому» разграничению доступа («coarse-grained access control») и служит причиной появления «избыточности прав доступа» у пользователей. Идентификаторные модели не содержат средств администрирования полномочий, оставляя это «третьей стороне» — системному администратору. В широкомасштабных системах управление правами большого числа пользователей и машин становится слишком сложным и подверженным ошибкам. В работе [5], написанной в 1993 г., проанализированы недостатки существующих моделей и сформулированы проблемы, требующие решения при организации разграничения доступа.

Прежде всего необходимо решить вопросы идентификации с целью избавления от избыточных прав доступа путём обеспечения точного наименования сущностей. Следующим недостатком является недостаток гибкости. Единственной политики безопасности, обеспечиваемой традиционными моделями DAC, MAC и RBAC в автономных приложениях, недостаточно для защиты данных в сложных системах, требующих интегрированных политик разграничения доступа для одновременного выполнения различных критериев защиты с целью обеспечения целостности, конфиденциальности и доступности данных. Одновременное выполнение этих условий требуется при защите сложных приложений в области медицины, финансов, резервирования билетов, научных исследований, цифровых библиотек и др.

Отмечается необходимость введения новой парадигмы — «множественной политики», означающей, что в системе должна быть предусмотрена возможность применения разных политик авторизации в зависимости от требований безопасности среды, в которой функционирует система. Для этого необходимо наличие средств конструирования различных политик без реконфигурации самой системы. Как правило, конкретные политики конструируются на основе единого объекта — политики высокого уровня, или метаполитики, реализуемой в виде фреймворка. Необходимо уточнить аспекты понятия множественной политики, касающиеся порядка применения создаваемых на основе метаполитики механизмов разграничения доступа различных видов.

Одним из путей является поочерёдное применение полученных политик, в результате которого система последовательно меняет применяемый критерий разграничения доступа. Возможность одновременного применения ряда созданных политик для разграничения доступа по различным признакам делает защиту многокритериальной.

В последнее время наблюдается значительный рост числа крупномасштабных распределённых открытых систем, являющихся виртуальными организациями, составленными из множества независимых автономных доменов. Ввиду того, что ресурсы и пользователи зачастую располагаются в разных доменах, связи между субъектами и объектами в таких системах становятся более сложными и динамичными. При этом возникает необходимость идентификации сущностей наборами характеристик, а не предопределёнными именами, так как решения о предоставлении доступа должны приниматься с учётом оценки наборов атрибутов субъектов и объектов, делая традиционные идентификаторные модели разграничения доступа неэффективными.

Для гетерогенных сервисов распределённых виртуальных сред необходимы механизмы авторизации, основанные на идентификации наборами признаков, которые обеспечивают необходимую точность, являются адекватными и надёжно защищают от атак [6].

2. Методы атрибутного разграничения доступа

Для решения этих проблем был предложен атрибутный метод разграничения доступа (Attribute Based Access Control — ABAC). Его основу составляет безыдентификаторный подход, который заключается в обозначении субъектов и объектов совокупностями атрибутов и позволяет принимать решение по управлению доступом без предварительного знания субъектов или их отношения к поставщику услуг.

Наиболее общим определением ABAC является следующее. Атрибутное разграничение доступа — это метод, посредством которого запросы субъекта на выполнение определённых операций над объектом удовлетворяются или отвергаются на основе приписанных им атрибутов, условий среды выполнения и набора политик, сформулированных с учётом этих условий и атрибутов.

Атрибутное разграничение доступа является многообещающей альтернативой традиционным моделям. Оно привлекает внимание как академических исследователей, так и создателей промышленных приложений [7 – 9].

Преимущество ABAC состоит в том, что оно позволяет создавать политики доступа на основе атрибутов пользователей и объектов, а не назначать роли, права собственности или метки безопасности вручную системным администратором. Это упрощает администрирование в сложных системах с большим числом пользователей, устраняя необходимость ручного вмешательства при авторизации пользователей для определённых ролей или уровней безопасности, а также создавая возможность автоматизации решения по управлению доступом для удалённых пользователей из других доменов.

Система именования сущностей атрибутами обеспечивает точность идентификации и, следовательно, «точное» разграничение доступа (fine-grained access control) в процессе организации защищённого использования ресурсов, не допуская избыточных прав доступа у пользователей. Языки спецификации моделей ABAC дают гибкость и выразительную мощность описаниям политик безопасности. При этом многие из них разрешают моделировать традиционные методы разграничения доступа — DAC, MAC, RBAC.

Модели атрибутного разграничения доступа находят применение в самых разных областях современных вычислений для защиты приложений, баз данных, файловых серверов, в облачных средах и больших данных.

За последнее время разработано множество ABAC-моделей, как базовых, так и специализированных [10 – 12]. Их объединяет то, что они могут рассматриваться в качестве основополагающих моделей нового направления защиты, способных решать задачи разграничения доступа, поставленные в [5].

Типичная ABAC-модель содержит следующие компоненты:

— атрибуты пользователей;
— атрибуты объектов;
— атрибуты контекста или вычислительной среды;
— политики авторизации, основанные на этих атрибутах.

Атрибуты обычно определяются в виде функций, аргументами которых служат субъекты или объекты, а результатом — значения их атрибутов. Политика авторизации предоставляет группам пользователей определённые виды доступа (такие, например, как чтение и запись) к заданным объектам на основе оценки значений их атрибутов.

В [13, 14] описываются две техники спецификации политик авторизации. Наиболее распространённой из них считается спецификация, определяющая политики с помощью формул логики, содержащих атрибуты в качестве своих переменных (LAP — Logical-formula Authorization Policy). LAP задаётся с помощью булевых выражений, состоящих из подвыражений, соединённых логическими операторами и операторами отношений. LAP предоставляет доступ субъекта к объекту, если в результате вычисления логическое выражение принимает значение «истина». Примерами моделей LAP-ABAC служат [10, 11, 15, 16]. Гибкость этих моделей продемонстрирована их способностью моделировать традиционные DAC, MAC и RBAC. Альтернативной техникой представления атрибутных политик разграничения доступа является перечисление. Примерами этой категории служат Policy Machine (PM) [17] и 2-sorted-RBAC [18].

Неформально перечислимая политика авторизации (Enumerated Authorization Policy, EAP) определяется как множество кортежей (uai, OP, oai), где uai и oai — значения атрибутов пользователя и объекта соответственно; OP — множество операций, доступных пользователю. В EAP кортежи различны и независимы, а значения атрибутов могут быть как атомарными, так и в виде множеств.

Полезность перечислимых политик авторизации продемонстрирована на многих примерах. Так, в Policy Machine [17] определены перечислимые атрибутные политики с использованием одного атрибута пользователя, одного атрибута объекта и набора возможных действий. Простая структура политики перечисления не делает её менее выразительной. Показано, что посредством PM могут быть сконфигурированы политики MAC и RBAC.

Преимущество логического подхода для представления атрибутных политик разграничения доступа в виде формул логики заключается в его простоте и лёгкости использования. Создание новых правил авторизации не представляет трудностей, так как не включает дополнительных расходов, требуемых, например, для инжиниринга ролей в RBAC. Эти правила способны гибко и в сжатой форме описывать даже сложные политики. Не существует ограничений на количество используемых в них атрибутов и сложность языка описания правил [7].

С другой стороны, создание выразительных вычислительных языков для спецификации атрибутных правил разграничения доступа делает задачи вычисления зна-

чений разнородных атрибутов в процессе конструирования и выполнения политик NP-полными или даже неразрешимыми, что служит, вместе с отсутствием формальных определений моделей и сложностью администрирования, главным препятствием широкому применению метода ABAC.

Относительно политик перечисления, разработка которых начата сравнительно недавно, сделано предположение о полиномиальном времени, требуемом для оценки атрибутов.

3. Виды моделей ABAC

В [6, 9, 13] приводятся описания многочисленных моделей атрибутного разграничения доступа, разработанных за последнее время. Их можно разделить на две категории — модели общего назначения и специализированные, предназначенные для применения в определённых вычислительных средах, таких, как облачные вычисления, веб-сервисы и т. д.

Примерами моделей общего назначения служат:

— логический фреймворк для ABAC (A Logic-Based Framework for Attribute-Based Access Control) — модель в форме фреймворка, основанного на логическом программировании, в которой политики определяются в виде логических программ, а атрибуты и операции — как объекты теории перечислимых множеств;
— атрибутная матрица доступа (Attribute-Based Access Matrix Model, ABAM). В ней строки представляют собой пары, состоящие из субъектов и множеств их атрибутов, а столбы содержат пары объектов и их атрибутов. Клетка матрицы содержит множество прав доступа субъекта к объекту в соответствии с принятой политикой безопасности;
— модель Rubio-Medrano, отображающая атрибуты сущностей в маркеры безопасности, соотнесённые с привилегиями, путём обхода графа маркеров, определяемого администратором. В результате обхода осуществляется принятие/отказ решения о доступе;
— ABAC-alpha — ещё одна недавно созданная модель (2012), предназначенная специально для моделирования DAC, MAC и RBAC. Для этого даётся формальное определение основных элементов ABAC (пользователей, объектов, политик и т. д.), их отношений и ограничений, позволяющее эмулировать традиционные модели;
— модель Policy Machine предлагает инновационный подход к разграничению доступа. Она предоставляет архитектуру и фреймворк для спецификации и реализации обобщённых атрибутных политик разграничения доступа, составляющие унифицированный механизм для реализации широкого круга различных политик;
— модель атрибутного разграничения доступа на основе иерархических групп (Hierarchical Group and Attribute-Based Access Control — HGABAC) создает обобщённую модель ABAC с иерархическим представлением групп атрибутов субъектов и объектов.

К числу специализированных моделей ABAC относятся следующие: модель CA-ABAC, предназначенная для организации разграничения доступа в облаках; T-ABAC, применяемая в системах реального времени; MPABAC — в объединённых рабочих и образовательных средах. Специальные модели разработаны для сред мобильных приложений, грид-вычислений, веб-сервисов, цифровых библиотек и т. д.

В [9] приводится анализ этих моделей и делается вывод о том, что большинство из них ориентировано на специфические условия использования и не отвечают требо-

ваниям, обеспечивающим универсальность, удобство администрирования и гибкость управления безопасностью.

Для выполнения данных требований модели должны обладать следующими характеристиками:

— иметь формальное определение;
— содержать средства администрирования;
— обеспечивать возможность применения множественных политик и включать способы их конструирования;
— язык описания модели должен быть высокоуровневым, т. е. не зависеть от операционной среды.

4. Задача спецификации моделей ABAC

Типичный механизм разграничения доступа содержит данные, описывающие политики разграничения доступа и атрибуты, а также набор функций для приёма и обработки запросов на доступ согласно этим политикам.

Реализация разграничения доступа осуществляется по-разному в различных операционных средах. Особенности распределённых гетерогенных сред ставят дополнительные задачи, требующие учитывать круг контролируемых объектов (пользователей и ресурсов), различные типы операций (такие, например, как чтение, пересылка, одобрение, выбор), а также типы данных (записи, файлы, сообщения, рабочие заметки). Администраторы вынуждены принимать во внимание наличие разных доменов безопасности, управляемых различными политиками на основании характерных атрибутов.

Необходимость осуществления глобального контроля безопасности ставит задачу разработки высокоуровневых средств, позволяющих конструировать различные политики на основе метаполитики как единого высокоуровневого объекта, с помощью соответствующих средств администрирования, которые не зависели бы от операционной среды.

В настоящее время разработан ряд языков спецификации политик авторизации и основанных на них систем разграничения доступа. К их числу относятся XACML, NGAC, Ponder, Akenti, dRBAC и др. [19, 20].

Многие из рассмотренных моделей ABAC используют собственные встроенные языки описания политик или вовсе не имеют языков.

В работах [21 – 30] приводится описание различных аспектов, а также формальное представление модели разрабатываемого метода типизированного атрибутного разграничения доступа (ТАРД). В данной работе кратко рассмотрены общие положения ТАРД, даны определения его модели на нескольких уровнях, описан порядок функционирования основанных на ней систем.

Модель ТАРД принадлежит классу моделей ABAC, но, в отличие от ABAC, атрибутам безопасности сущностей ТАРД приписаны определённые типы. Решение о возможности доступа принимается на основе обработки однотипных атрибутов пары субъект — объект.

Данная модель может быть отнесена к разряду моделей разграничения доступа общего назначения, основанной на логических формулах (LAP-ABAC). В её состав входят:

— средства определения возможности доступа субъектов к объектам в соответствии с их полномочиями;

— средства двухступенчатого администрирования — для конструирования политик авторизации и для управления идентификацией сущностей в процессе разграничения доступа.

В соответствии с определением типа атрибутов T модель ТАРД имеет многоуровневое представление, которое делает её инструментом конструирования механизмов разграничения доступа в виде типов, моделирующих различные политики безопасности, включая DAC, MAC и RBAC. Создаваемые политики могут применяться как по очереди, так и одновременно, что обеспечивает выполнение всех аспектов парадигмы «множественной политики».

Идентификация сущности ТАРД осуществляется путём присвоения ей метки безопасности в виде значения определённого типа или множественной метки безопасности в виде ряда значений различных типов.

Тип T определяется как математический объект, содержащий домен типа — конечное множество всевозможных значений атрибутов в виде полного частично упорядоченного множества, структурированного отношением предшествования, и определённых на нем операций типизации Type, представленных непрерывными монотонными функциями, а также операцией доступа Acc [21, 23, 27, 28]. Операция типизации присваивает сущности e метку безопасности T:

$$T(e) = \mathrm{Type}(e).$$

Множественная метка безопасности в виде кортежа $T_1(e), \ldots, T_K(e)$ присваивается в результате применения нескольких операций типизации типов $T_1, \ldots, T_K$.

Предикат Acc осуществляет сравнение меток безопасности типа T субъекта и объекта, разрешая/отвергая возможность доступа. Истинностное значение означает разрешение на доступ субъекту s к объекту o:

$$\mathrm{Acc}(T(s), T(o)) = \texttt{true/false}.$$

Набор операций типа T образует механизм разграничения доступа, а также служит средством реализации политики типизированного атрибутного разграничения доступа $P(T)$, задаваемой типом T [23].

Спецификация политики безопасности $P(T)$ задаётся конкретным видом структуры и значениями узлов домена типа T, а также соответствующими этой структуре операциями. Тип T служит ограничением на значения атрибутов и круг операций с атрибутами данного типа. Данное обстоятельство лежит в основе принципа безопасности моделей ТАРД [26].

Метод ТАРД имеет многоуровневое определение в виде метауровня META, объектного уровня OBJ и уровня матрицы доступа AM. В соответствии с этим даётся многоуровневое определение типа T.

На уровне META тип представлен метатипом T_{META}, являющимся обобщённым представлением политики $P_{\mathrm{META}}(T)$. Он служит для порождения политик типизированного атрибутного разграничения доступа объектного уровня $P_{\mathrm{OBJ}}(T)$.

Типы объектного уровня T_{OBJ} представляют собой ряд конкретных политик разграничения доступа $P_{\mathrm{OBJ}}(T)$, получаемых из метаполитики $P_{\mathrm{META}}(T)$. При этом T_{OBJ} является интенсиональным представлением типа T.

Тип T_{AM} уровня AM — это реализация политики типизированного атрибутного разграничения доступа $P_{\mathrm{OBJ}}(T)$ в виде множества типизированных переменных, образующих матрицу доступа и являющихся экстенсиональным представлением типа T.

Построена модель $M(T)$, которая является формальным представлением типа T. Она предназначена для конструирования и представления предметной области ТАРД на этапах работы системы разграничения доступа. Модель, так же как и тип, имеет многоуровневое определение. Переход на следующий уровень производится путём моделирования семантики предыдущего уровня в процессе функционирования системы разграничения доступа.

Уровень метамодели $M_{\text{META}}(T)$ представляет метатип T_{META} и играет роль фреймворка, или инструмента для создания конкретных моделей ТАРД. Объектная модель $M_{\text{OBJ}}(T)$ — это объектный тип T_{OBJ}, который является средством реализации конкретной политики безопасности типа T, полученной из метатипа T_{META}.

Матрица доступа $M_{\text{AM}}(T)$ — сложноструктурированная область типизированных значений атрибутов в виде меток безопасности сущностей, содержит результаты выполнения политики безопасности $P(T)$ в процессе функционирования модели $M_{\text{OBJ}}(T)$.

В процессе работы на разных уровнях модель выполняет различные функции.

На уровне META осуществляется конструирование семантических значений типа T_{OBJ} специальной операцией интерпретации I, которое заключается в определении вида структуры домена типа T_{OBJ} как подструктуры домена типа T_{META} и присвоении значений его элементам. В результате создаются различные виды моделей $M^i_{\text{OBJ}}(T)$, предназначенные для выполнения соответствующих политик разграничения доступа $P^i_{\text{OBJ}}(T)$:

$$M_{\text{META}}(T, I) \to M^i_{\text{OBJ}}(T).$$

Формирование матрицы доступа осуществляется операциями Type сконструированных моделей $M^i_{\text{OBJ}}(T)$. При этом производится присвоение полномочий сущностям в виде их меток безопасности:

$$M^i_{\text{OBJ}}(T, \text{Type}(e)) \to M^i_{\text{AM}}(T).$$

На уровне AM производится обработка создаваемой матрицы доступа путём выдачи разрешения на доступ согласно критерию, задаваемому типом T_{OBJ}, в результате выполнения операции доступа Acc модели $M^i_{\text{OBJ}}(T)$:

$$M^i_{\text{OBJ}}(T, \text{Acc}(T(s), T(o)) \to \{\texttt{true}, \texttt{false}\}.$$

Модели $M^i_{\text{OBJ}}(T)$ могут выполняться как последовательно, так и параллельно, реализуя все аспекты множественной политики.

Обозначим $M^i_{\text{OBJ}}(T) = M_{\text{OBJ}}(T_i)$ и рассмотрим множество различных моделей $\{M_{\text{OBJ}}(T_i) : i = 1, \ldots, K\}$, где типы T_i семантически независимы. Одновременное использование этих моделей в рамках одной системы обеспечивает разграничение доступа по ряду критериев, определяемых политиками $T_1, \ldots, T_K$.

Предикат $MM_{\text{AM}}(T_1, \ldots, T_K)$ предоставляет доступ субъекту s к объекту o при условии одновременного выполнения критериев $T_1, \ldots, T_K$:

$$MM_{\text{AM}}(T_1, \ldots, T_K) = M_{\text{AM}}(T_1, \text{Acc}_1) \wedge \ldots \wedge M_{\text{AM}}(T_K, \text{Acc}_K). \quad (1)$$

Построено формальное представление модели ТАРД, определяемое на уровнях META, OBJ и AM в виде языковых спецификаций T^{AS} и T^{LS}, взаимосвязь которых обусловливает её функционирование [27, 28].

В работе [27] приводится представление типа атрибутов в виде алгебраической системы, в [28] содержится обзор возможностей применения логики для разграничения

доступа, используемых в настоящее время, а также описание языка T^{LS} для представления типа атрибутов ТАРД в виде логики.

Модель ТАРД представима в виде пары

$$M(T) = (T^{AS}, T^{LS}),$$

где $T^{AS} = (D, \sigma)$ — многоуровневое определение многосортной алгебраической системы, состоящей из домена D и сигнатуры σ операций типа T, служащей для представления состояния каждого уровня модели:

$$T^{AS} = (T^{AS}_{\text{МЕТА}}, T^{AS}_{\text{ОБЪ}}, T^{AS}_{\text{АМ}});$$

T^{LS} — многосортная логическая система, включающая язык L с алфавитом, представленным доменом D и правилами грамматики — операциями типа T, а также аксиомами Ax и правилами вывода Inf:

$$T^{LS} = (L, Ax, Inf).$$

Функционирование многоуровневой модели ТАРД заключается в моделировании семантики следующего уровня путём интерпретации модели, представляющей предыдущий уровень. При этом T^{LS}, представленная в виде тройки $(T^{LS}_{\text{МЕТА}}, T^{LS}_{\text{ОБЪ}}, T^{LS}_{\text{АМ}})$, образует систему семантического моделирования.

Ввиду того, что определение T^{LS} включает функции интерпретации, схему взаимодействия спецификаций модели ТАРД можно представить следующим образом:

Метауровень МЕТА: $T^{LS}_{\text{МЕТА}}(T^{AS}_{\text{МЕТА}}) \to T^{AS}_{\text{ОБЪ}}$.

Объектный уровень ОБЪ: $T^{LS}_{\text{ОБЪ}}(T^{AS}_{\text{ОБЪ}}) \to T^{AS}_{\text{АМ}}$.

Уровень матрицы доступа АМ: $T^{LS}_{\text{АМ}}(T^{AS}_{\text{АМ}}) \to \{\texttt{true}, \texttt{false}\}$.

В целях реализации модели ТАРД для каждого её уровня даётся формальное определение системы типизированного атрибутного разграничения доступа как программы (или программной системы) $S(T)$, построенной на основе модели $M(T)$. Она обеспечивает разграничение доступа в соответствии с политикой, представленной типом T [29].

Программа S является общим представлением функции семантического моделирования состояний предметной области ТАРД. Многокритериальное разграничение доступа осуществляется программой S, построенной на основе моделей $M(T_1), \ldots, M(T_K)$ согласно формуле (1).

Архитектура системы ТАРД состоит из модуля настройки и модуля выполнения, соответствующих мета- и объектным уровням определения модели ТАРД.

Модуль настройки (Администратор 1) осуществляет конфигурирование системы путём задания одного типа T или нескольких типов $T_1, \ldots, T_K$ атрибутов (которое заключается в задании их структуры и определении значений элементов доменов), служащих критериями разграничения доступа. Критерий или набор критериев, обеспечиваемых сконструированными типами, должен отвечать требованиям защиты данной информационной системы.

Модуль выполнения осуществляет две функции — функцию администрирования (Администратор 2), которая присваивает сущностям (субъектам и объектам) значение типа $T(e)$ или кортежи значений типов $(T_1(e), \ldots, T_K(e))$ в качестве их меток безопасности, и функцию разграничения, которая обеспечивает доступ субъектов к объектам на основе сравнения меток безопасности одинаковых типов.

Необходимо отметить, что выразительность метода ТАРД несколько ограничена по сравнению с общепринятыми ABAC-методами.

Описанная модель ТАРД обладает следующими характеристиками:

— принцип обработки однотипных атрибутов является предпосылкой обеспечения скорости вычислений их значений;
— модель имеет возможность формально доказывать правильность решений о предоставлении доступа, используя дедуктивный аппарат логической спецификации;
— обеспечивает наглядность и контроль процесса администрирования;
— способна динамически конструировать новые модели разграничения доступа вместе с возможностью моделировать традиционные DAC, MAC, RBAC и применять их в системе как одну за другой, так и одновременно;
— модель непосредственно реализуема на языках функционального и логического программирования с использованием аппарата программирования в ограничениях.

Перечисленные особенности позволяют использовать системы типизированного атрибутного разграничения доступа в качестве центров управления политиками безопасности, создаваемых в локальных и глобальных вычислительных средах. На основе этих центров, рассматриваемых в качестве виртуальных организаций, могут создаваться домены администрирования, поддающиеся локализации и обеспечивающие в силу этого полную защиту информации на контролируемых ими участках распределённых гетерогенных вычислительных сред [30].

Заключение

Рассмотрены основные методы разграничения доступа в их развитии — от традиционных моделей DAC, MAC и RBAC до последних разработок — моделей, основанных на методе атрибутного разграничения доступа ABAC.

Приведён перечень недостатков традиционных моделей и пути их преодоления. Сформулированы требования, обеспечивающие гибкость управления безопасностью, универсальность и удобство администрирования, которым должны отвечать модели разграничения доступа, функционирующие как в локальных, так и в распределённых гетерогенных средах.

Перечислены основные разработанные к настоящему времени модели атрибутного разграничения доступа, отмечены их достоинства и недостатки. Описана модель типизированного атрибутного разграничения доступа, показано, что она отвечает основным требованиям построения моделей, обеспечивающим безопасное пользование разделяемыми ресурсами.

ЛИТЕРАТУРА

1. *Karp A., Haury H., and Davis M.* From ABAC to ZBAC: The evolution of access control models // ISSA J. 2010. No. 8. P. 22–30.
2. *Sandhu R. S. and Samarati P.* Access control: principle and practice // IEEE Commun. Mag. 1994. No. 32(9). P. 40–48.
3. *Девянин П. Н.* Модели безопасности компьютерных систем: учеб. пособие для вузов. М.: Академия, 2005. 144 с.
4. *Гайдамакин Н. А.* Разграничение доступа к информации в компьютерных системах. Екатеринбург: Изд-во Урал. ун-та, 2003. 328 с.
5. *Hosmer H.* The multipolicy paradigm for trusted systems // Proc. NSPW '92-93. ACM, N.Y.: ACM, 1993. P. 19–32.
6. *Lang B. et al.* A flexible attribute based access control method for grid computing // J. Grid Comput. 2009. No. 7(2). P. 169–180.

7. *Hu V. C., Ferraiolo D., Kuhn R., et al.* Guide to Attribute Based Access Control (ABAC) Definition and Considerations. NIST Special Publication, 800:162, 2014. http://dx.doi.org/10.6028/NIST.SP.800-162
8. https://nccoe.nist.gov/sites/default/files/nccoe/NIST_SP1800-3c_ABAC_0.pdf — NCCOE. Attribute Based Access Control How-to Guides for Security Engineers. Accessed November 25, 2015.
9. *Servos D. and Osborn S.* Current research and open problems in attribute-based access control // ACM Computing Surveys. 2017. V. 49. Iss. 4. Art. 65.
10. *Jin X., Krishnan R., and Sandhu R. S.* A unified attribute-based access control model covering DAC, MAC and RBAC // LNCS. 2012. V. 7371. P. 41–55.
11. *Servos D. and Osborn S.* HGABAC: Towards a formal model of hierarchical attribute-based access control // Foundations and Practice of Security. Springer, 2014. P. 187–204.
12. *Yuan E. and Tong J.* Attributed based access control (ABAC) for web services // Proc. ICWS'2005. Washington, 2005. P. 561–569.
13. *Biswas R., Sandhu R., and Krishnan R.* Label-based access control: An ABAC model with enumerated authorization policy // Proc. ABAC'16. N.Y.: ACM, 2016. P. 1–12.
14. *Biswas P., Sandhu R., and Krishnan R. A.* A comparison of logical-formula and enumerated authorization policy ABAC models // LNCS. 2016. V. 9766. P. 122–129.
15. *Shen H. and Hong F.* An attribute-based access control model for web services // Proc. PDCAT'06. IEEE, 2006. P. 74–79.
16. *Wang L., Wijesekera D., and Jajodia S.* A logic-based framework for attribute based access control // Proc. FMSE'04. ACM, 2004. P. 45–55.
17. *Ferrailo D., Atluri V., and Gavrila S.* The Policy Machine: A novel architecture and framework for access control policy specification and enforcement // J. Systems Architecture. 2011. V. 57(4). P. 412–424.
18. *Kuijper W. and Ermolaev V.* Sorting out role based access control // Proc. 19th ACM SACMAT. ACM, 2014. P. 63–74.
19. *Ferraiolo D., Chandramouli R., Hu V. C., and Kuhn R. A.* A Comparison of Attribute Based Access Control (ABAC) Standards for Data Service Applications, Extensible Access Control Markup Language (XACML) and Next Generation Access Control (NGAC). Natl. Inst. Stand. Technol. Spec. Publ. 800-178, 2016. 68 p.
20. *Wakefield R.* Policy Management in a Distributed Computing Environment. http://www.cs.colostate.edu/~waker/papers/CS556_Policy_Management_in_Distributed_Computing.pdf. 2008.
21. *Калимолдаев М. Н., Бияшев Р. Г., Рог О. А.* Формальное представление функциональной модели многокритериальной системы разграничения и контроля доступа к информационным ресурсам // Проблемы информатики. 2014. № 1(22). С. 43–55.
22. *Rog O. A.* Polymorphic typing of entities in the multi-criteria system of access control and a task of constructing types // Inform. Technologies, Management and Society. 12th Intern. Scientific Conf. Riga, April 16–17, 2014. P. 66.
23. *Бияшев Р. Г., Калимолдаев М. Н., Рог О. А.* Полиморфная типизация сущностей и задача конструирования механизма многокритериального разграничения доступа // Изв. НАН РК. Сер. физ.-мат. 2014. № 5. С. 33–41.
24. *Бияшев Р. Г., Калимолдаев М. Н., Рог О. А.* Конструирование систем многокритериального атрибутного разграничения доступа в облачных структурах // 11 Междунар. Азиатская школа-семинар «Проблемы оптимизации сложных систем». Чолпон-Ата, 27 июля–7 августа 2015. С. 148–152.

25. *Бияшев Р. Г., Калимолдаев М. Н., Рог О. А.* Логический подход к организации многокритериального атрибутного разграничения доступа // Int. Conf. «Computational and Informational Technologies in Science, Engineering and Education» (September 24–27, 2015). Almaty: Казак университеті, 2015. P. 86.
26. *Бияшев Р. Г., Калимолдаев М. Н., Рог О. А.* Представление ограничений моделей атрибутного разграничения доступа // Изв. НАН РК. Сер. физ.-мат. 2016. № 1. С. 58–65.
27. *Бияшев Р. Г., Калимолдаев М. Н., Рог О. А.* Моделирование семантики типизированного атрибутного разграничения доступа // Проблемы информатики. 2017. № 1. С. 25–37.
28. *Калимолдаев М. Н., Бияшев Р. Г., Рог О. А.* Применение логики для построения моделей разграничения доступа к информации // Докл. НАН РК. 2017. № 3. С. 48–54.
29. *Калимолдаев М. Н., Бияшев Р. Г., Рог О. А.* Основы архитектуры программных систем для осуществления типизированного атрибутного разграничения доступа // Современные проблемы информатики и вычислительных технологий: Материалы науч. конф. (29–30 июня 2017). Алматы, 2017. С. 88–95.
30. *Калимолдаев М. Н., Бияшев Р. Г., Рог О. А.* О применении типизированного атрибутного разграничения доступа в глобальных вычислительных средах // Изв. науч.-технич. общества «КАХАК». Алматы, 2017. № 3(58). С. 30–36.

REFERENCES

1. *Karp A., Haury H., and Davis M.* From ABAC to ZBAC: The evolution of access control models. ISSA J., 2010, no. 8, pp. 22–30.
2. *Sandhu R. S. and Samarati P.* Access control: principle and practice. IEEE Commun. Mag., 1994, no. 32(9), pp. 40–48.
3. *Devyanin P. N.* Modeli bezopasnosti kompyuternyh sistem. Ucheb. posobie dlja stud. vyssh. ucheb. zavedenij) [Models of security of computer systems: Textbook for students of higher educational institutions]. Moscow, Akademija Publ., 2005. 144 p. (in Russian)
4. *Gaydamakin N. A.* Razgranichenie dostupa k informatsii v komp'yuternykh sistemakh) [Differentiation of Access to Information in Computer Systems]. Ekaterinburg, USU Publ., 2003. 328 p. (in Russian)
5. *Hosmer H.* The multipolicy paradigm for trusted systems. Proc. NSPW '92-93, N.Y., ACM, 1993, pp. 19–32.
6. *Lang B. et al.* A flexible attribute based access control method for grid computing. J. Grid Comput., 2009, no. 7(2), pp. 169–180.
7. *Hu V. C., Ferraiolo D., Kuhn R., et al.* Guide to Attribute Based Access Control (ABAC) Definition and Considerations. NIST Special Publ., 800:162, 2014. `http://dx.doi.org/10.6028/NIST.SP.800-162`
8. `https://nccoe.nist.gov/sites/default/files/nccoe/NIST_SP1800-3c_ABAC_0.pdf` — NCCOE. Attribute Based Access Control How-to Guides for Security Engineers. Accessed November 25, 2015.
9. *Servos D. and Osborn S.* Current research and open problems in attribute-based access control. ACM Computing Surveys, 2017, vol. 49, iss. 4, art. 65.
10. *Jin X., Krishnan R., and Sandhu R. S.* A unified attribute-based access control model covering DAC, MAC and RBAC. LNCS, 2012, vol. 7371, pp. 41–55.
11. *Servos D. and Osborn S.* HGABAC: Towards a formal model of hierarchical attribute-based access control. Foundations and Practice of Security, Springer, 2014, pp. 187–204.
12. *Yuan E. and Tong J.* Attributed based access control (ABAC) for web services. Proc. ICWS'2005, Washington, 2005, pp. 561–569.

13. *Biswas R., Sandhu R., and Krishnan R.* Label-based access control: An ABAC model with enumerated authorization policy. Proc. ABAC'16, N.Y., ACM, 2016, pp. 1–12.
14. *Biswas P., Sandhu R., and Krishnan R. A.* A comparison of logical-formula and enumerated authorization policy ABAC models. LNCS, 2016, vol. 9766, pp. 122–129.
15. *Shen H. and Hong F.* An attribute-based access control model for web services. Proc. PDCAT'06, IEEE, 2006, pp. 74–79.
16. *Wang L., Wijesekera D., and Jajodia S.* A logic-based framework for attribute based access control. Proc. FMSE'04, ACM, 2004, pp. 45–55.
17. *Ferrailo D., Atluri V., and Gavrila S.* The Policy Machine: A novel architecture and framework for access control policy specification and enforcement. J. Systems Architecture, 2011, vol. 57(4), pp. 412–424.
18. *Kuijper W. and Ermolaev V.* Sorting out role based access control. Proc. 19th ACM SACMAT, ACM, 2014, pp. 63–74.
19. *Ferraiolo D., Chandramouli R., Hu V. C., and Kuhn R. A.* A Comparison of Attribute Based Access Control (ABAC) Standards for Data Service Applications, Extensible Access Control Markup Language (XACML) and Next Generation Access Control (NGAC). Natl. Inst. Stand. Technol. Spec. Publ. 800-178, 2016. 68 p.
20. *Wakefield R.* Policy Management in a Distributed Computing Environment. `http://www.cs.colostate.edu/~waker/papers/CS556_Policy_Management_in_Distributed_Computing.pdf`. 2008.
21. *Kalimoldayev M. N., Biyashev R. G., and Rog O. A.* Formal'noe predstavlenie funkcional'noj modeli mnogokriterial'noj sistemy razgranichenija i kontrolja dostupa k informacionnym resursam [Formal representation of the functional model of a multi-criteria system of the access control to information resources]. Problemy Informatiki, 2014, no. 1(22), pp. 43–55. (in Russian)
22. *Rog O. A.* Polymorphic typing of entities in the multi-criteria system of access control and a task of constructing types. Inform. Technologies, Management and Society, 12th Intern. Scientific Conf., Riga, April 16–17, 2014, p. 66.
23. *Biyashev R. G., Kalimoldayev M. N., and Rog O. A.* Polimorfnaja tipizacija sushhnostej i zadacha konstruirovanija mehanizma mnogokriterial'nogo razgranichenija dostupa [Polymorphic typing of entities and a task of constructing a mechanism for multi-criteria access control]. Izv. NAN RK, Ser. fiziko-matematicheskaja, 2014, no. 5, pp. 33–41. (in Russian)
24. *Biyashev R. G., Kalimoldayev M. N., and Rog O. A.* Konstruirovanie sistem mnogokriterial'nogo atributnogo razgranichenija dostupa v oblachnyh strukturah [Designing of systems of multi-criteria attribute-based access control in cloud structures]. 11 Mezhdunar. Aziatskaja Shkola-seminar "Problemy optimizacii slozhnyh sistem", Cholpon-Ata, 27 July – 7 August 2015, pp. 148–152. (in Russian)
25. *Biyashev R. G., Kalimoldayev M. N., and Rog O. A.* Logicheskij podhod k organizacii mnogokriterial'nogo atributnogo razgranichenija dostupa [Logical approach to the organization of multi-criteria attribute-based access control]. Intern. Conf. "Computational and Informational Technologies in Science, Engineering and Education" (September 24–27, 2015), Almaty, Kazak University, 2015, p. 86. (in Russian)
26. *Biyashev R. G., Kalimoldayev M. N., and Rog O. A.* Predstavlenie ogranichenij modelej atributnogo razgranichenija dostupa [Representation of the constraints of models of attribute-based access control]. Izv. NAN RK, Ser. fiziko-matematicheskaja, 2016, no. 1, pp. 58–65. (in Russian)
27. *Biyashev R. G., Kalimoldayev M. N., and Rog O. A.* Modelirovanie semantiki tipizirovannogo atributnogo razgranichenija dostupa [Modeling of semantics of typed attribute-based access control]. Problemy Informatiki, 2017, no. 1, pp. 25–37. (in Russian)

28. *Kalimoldayev M. N., Biyashev R. G., and Rog O. A.* Primenenie logiki dlja postroenija modelej razgranichenija dostupa k informacii [The use of logic for constructing models for the control of access to information]. Dokl. NAN RK, 2017, no. 3, pp. 48–54. (in Russian)
29. *Kalimoldayev M. N., Biyashev R. G., and Rog O. A.* Osnovy arhitektury programmnyh sistem dlja osushhestvlenija tipizirovannogo atributnogo razgranichenija dostupa [Fundamentals of the architecture of software systems for the implementation of typed attribute-based access control]. Proc. "Sovremennye Problemy Informatiki i Vychislitel'nyh Tehnologij" (29–30 Juny 2017), Almaty, 2017, pp. 88–95. (in Russian)
30. *Kalimoldayev M. N., Biyashev R. G., and Rog O. A.* O primenenii tipizirovannogo atributnogo razgranichenija dostupa v global'nyh vychislitel'nyh sredah [On the application of typed attribute-based access control in global computing environments]. Izv. Nauchno-Tehnicheskogo Obshhestva "KAHAK", Almaty, 2017, no. 3(58), pp. 30–36. (in Russian)

МАТЕМАТИЧЕСКИЕ ОСНОВЫ НАДЁЖНОСТИ ВЫЧИСЛИТЕЛЬНЫХ И УПРАВЛЯЮЩИХ СИСТЕМ

УДК 519.718

О НАДЁЖНОСТИ СХЕМ В БАЗИСЕ, СОСТОЯЩЕМ ИЗ ФУНКЦИИ ВЕББА, В P_k ПРИ НЕИСПРАВНОСТЯХ ТИПА 0 И ТИПА $k-1$ НА ВЫХОДАХ ЭЛЕМЕНТОВ[1]

М. А. Алехина*, О. Ю. Барсукова**

*Пензенский государственный технологический университет, г. Пенза, Россия
**Пензенский государственный университет, г. Пенза, Россия

Рассматривается реализация функций k-значной логики ($k \geqslant 3$) схемами из ненадёжных функциональных элементов в полном базисе, состоящем из функции Вебба. Предполагается, что элементы схемы переходят в неисправные состояния независимо друг от друга, подвержены однотипным константным неисправностям типа 0 или типа $k-1$ на выходах. Конструктивно доказано, что при неисправностях типа 0 почти любую функцию k-значной логики можно реализовать асимптотически оптимальной по надёжности схемой, функционирующей с ненадёжностью, асимптотически равной ненадёжности одного базисного элемента; при неисправностях типа $k-1$ любую функцию k-значной логики можно реализовать надёжной схемой, которая функционирует с ненадежностью, асимптотически не большей, чем в 3 раза, ненадежности одного базисного элемента. Полученные результаты справедливы в двойственном (относительно перестановки, порождаемой функцией Лукашевича) базисе при однотипных константных неисправностях типа $k-1$ и типа 0 соответственно.

Ключевые слова: *функции k-значной логики, ненадёжные функциональные элементы, надёжность и ненадёжность схемы, синтез схем из ненадёжных элементов, неисправности на выходах элементов.*

DOI 10.17223/20710410/44/5

ABOUT RELIABILITY OF CIRCUITS IN THE BASIS CONSISTING OF THE WEBB FUNCTION IN P_k UNDER FAILURES OF 0 TYPE AND $k-1$ TYPE AT THE OUTPUTS OF ELEMENTS

M. A. Alekhina*, O. Yu. Barsukova**

*Penza State Technological University, Penza, Russia
**Penza State University, Penza, Russia

E-mail: alekhina.marina19@yandex.ru, oksana.barsukova.71@gmail.com

We consider the realization of k-valued logics ($k > 3$) functions by circuits of unreliable functional elements in the complete basis consisting of the Webb function. We assume

[1]Работа поддержана грантом РФФИ № 17-01-00451а.

that elements of the circuit pass to fault states independently of each other, and they are exposed to single constant faults of type 0 or $k-1$ at their outputs. It is constructively proved that, under faults of type 0, almost any function of k-valued logics can be implemented by an asymptotically optimal in reliability circuit functioning with the unreliability which is asymptotically equal to unreliability of one basis element; under faults of type $k-1$, any function of k-valued logics can be implemented by an reliable circuit which functions with unreliability asymptotically no more than three times of the unreliability of one basic element. The obtained results are valid in a dual (with respect to the permutation generating by the Lukashevich function) basis for single-type constant faults of type $k-1$ and type 0 respectively.

Keywords: *k-valued logics functions, unreliable functional elements, reliability and unreliability of circuit, synthesis of circuits from unreliable elements, faults at outputs of elements.*

Введение

Настоящая работа продолжает исследования по надёжности схем, реализующих булевы функции [1–3]. В отличие от названных работ, рассматривается реализация функций k-значной логики ($k \geqslant 3$) схемами из ненадёжных элементов в полном базисе B, состоящем из функции Вебба $V_k(x_1, x_2) = (\max\{x_1, x_2\} + 1) \bmod k$, а также в двойственном (относительно перестановки, порождаемой функцией $N(x) = k-1-x$, которую называют отрицанием Лукашевича) базисе $B^* = \{(\min\{x_1, x_2\}+k-1) \bmod k\}$.

Задача синтеза надёжных схем в этих базисах решена в [4] при инверсных неисправностях на выходах базисных элементов (когда на каждом входном наборе любого из базисных элементов вероятность появления неверного значения на выходе элемента одинакова), причём были применены два различных метода синтеза и найдены условия, при которых один метод даёт лучшую оценку надёжности схем, чем другой. В отличие от [4], будем исследовать однотипные константные неисправности на выходах элементов двух видов: типа 0 и типа $k-1$. Предполагается, что элементы схемы переходят в неисправные состояния с вероятностью $\varepsilon \in (0, 1/2)$ независимо друг от друга.

Пусть $k, n \in \mathbb{N}$, $k \geqslant 3$, $E_k = \{0, 1, 2, \ldots, k-1\}$, P_k — множество всех функций k-значной логики, т. е. функций $f(\tilde{x}^n) : (E_k)^n \to E_k$, где $\tilde{x}^n = (x_1, \ldots, x_n)$. Используемые понятия и определения можно найти в [4, 5].

Далее докажем, что: 1) при неисправностях типа 0 на выходах элементов почти любую функцию k-значной логики можно реализовать асимптотически оптимальной по надёжности схемой, функционирующей с ненадёжностью, асимптотически ($\varepsilon \to 0$) равной ε (где ε — вероятность появления неисправности типа 0); 2) при неисправностях типа $k-1$ на выходах элементов любую функцию k-значной логики можно реализовать надёжной схемой, функционирующей с ненадёжностью асимптотически не больше 3ε (где ε — вероятность появления неисправности типа $k-1$). Для доказательства будем использовать первый из двух методов синтеза надёжных схем, изложенных в [4]. Сложностью $L(S)$ схемы S будем называть число элементов в ней.

1. Оценки ненадёжности схем при неисправностях типа 0

Будем считать, что элементы базиса $B = \{V_k\}$ с вероятностью ε подвержены однотипным константным неисправностям типа 0 на выходах элементов.

Теорема 1 [5]. Пусть f — произвольная функция из P_k; S — любая схема, реализующая f; $P(S)$ — ненадёжность схемы S. Тогда можно построить схему $\psi(S)$, которая реализует функцию f с ненадёжностью

$$P(\psi(S)) \leqslant \varepsilon + (2^{2^k-1} - 2^k)\varepsilon^2 + 2^{2k}\varepsilon P(S) + (2^{2^k} - 2^k - 1)P^2(S). \tag{1}$$

В теореме 1 установлено рекуррентное соотношение между ненадёжностью схемы $\psi(S)$ и ненадёжностью исходной схемы S. В соотношении (1) отбросим отрицательные слагаемые в правой части, получим неравенство

$$P(\psi(S)) \leqslant \varepsilon + 2^{2^k-1}\varepsilon^2 + 2^{2k}\varepsilon P(S) + 2^{2^k}P^2(S). \tag{2}$$

Замечание 1. Справедливы следующие утверждения:

1) при $k \geqslant 3$ верно:
   а) $0{,}5 + 1{,}25k^4 + 25k^8 < 27k^8$;
   б) $5k^4 + 2 < 5{,}1k^4$;
   в) $0{,}5 + 1{,}275k^4 + 26{,}01k^8 < 27k^8$;
   г) $5k^4 + 4 < 5{,}1k^4$;
2) функция $g(x) = 2^{2x-2^x}$ убывает на $[3, +\infty)$, поэтому наибольшее значение на $[3, +\infty)$, равное $1/4$, она принимает в точке $x = 3$.

Преобразуем правую часть (2), учитывая п. 2 замечания 1:

$$2^{2^k-1}\varepsilon^2 + 2^{2k}\varepsilon P(S) + 2^{2^k}P^2(S) = 2^{2^k}(1/2\varepsilon^2 + 2^{2k-2^k}\varepsilon P(S) + P^2(S)) \leqslant$$
$$\leqslant 2^{2^k}(1/2\varepsilon^2 + 1/4\varepsilon P(S) + P^2(S)).$$

Тогда из соотношения (2) следует неравенство

$$P(\psi(S)) \leqslant \varepsilon + 2^{2^k}(1/2\varepsilon^2 + 1/4\varepsilon P(S) + P^2(S)). \tag{3}$$

Чтобы получить верхнюю оценку ненадёжности схем, применим леммы 1–4, доказанные в [4].

Лемма 1 [4]. m-Местный максимум ($m \geqslant 2$) можно реализовать схемой из $(m-1)k$ элементов.

Лемма 2 [4]. Константы $0, 1, \ldots, k-1$ можно реализовать схемой, сложность которой не более $k^2 + k - 2$.

Для $i \in E_k$ определим следующие функции:

$$J_i(x) = \begin{cases} k-1, & \text{если } x = i, \\ 0, & \text{если } x \neq i. \end{cases}$$

Лемма 3 [4]. Функции $J_0(x), J_1(x), \ldots, J_{k-1}(x)$ можно реализовать схемой T, содержащей не более $k(k-1)^2$ элементов, т. е. $L(T) \leqslant k(k-1)^2$.

Лемма 4 [4]. Функцию $\min\{x_1, x_2\}$ можно реализовать такой схемой $S_\&$, что $L(S_\&) < 3k^3 + 4{,}5k^2 - 6{,}5k$.

Теорема 2. Любую функцию $f \in P_k$ можно реализовать схемой D', ненадёжность которой $P(D') < 2\varepsilon$ при всех $\varepsilon \in (0, \varepsilon_0]$, где $\varepsilon_0 = 2^{-2^k}/(27k^8)$.

Доказательство. Так же, как в [4, теорема 2], применим индукцию по числу переменных функции $f(\tilde{x}^n)$.

1. Б а з а и н д у к ц и и: $n = 1$. Представим функцию $f(x)$ в виде (аналог совершенной ДНФ) [6]

$$f(x) = J_0(x)\&f(0) \vee J_1(x)\&f(1) \vee \ldots \vee J_{k-1}(x)\&f(k-1).$$

Промоделируем эту формулу схемой S', используя схемы из лемм 2 и 3, и найдём число элементов в ней. По лемме 2 для реализации всех констант достаточно k^2+k-2 элементов. По лемме 3 для реализации функций $J_0(x), J_1(x), \ldots, J_{k-1}(x)$ достаточно $k(k-1)^2$ элементов. По лемме 4 для реализации k двухместных минимумов достаточно $k(3k^3+4{,}5k^2-6{,}5k)$ элементов. Наконец, для реализации одного k-местного максимума достаточно $k(k-1)$ элементов. Поэтому схема S' содержит не более

$$k^2+k-2+k(k-1)^2+k(3k^3+4{,}5k^2-6{,}5k)+k(k-1) = \\ = 3k^4+5{,}5k^3-6{,}5k^2+k-2 < 3k^4+5{,}5k^3$$

элементов. Поскольку $k \geqslant 3$, верно неравенство $5{,}5 < 2k$. Поэтому $L(S') < 5k^4$.

Известно [7, с. 47], что ненадёжность любой схемы не больше суммы ненадёжностей всех её элементов, поэтому $P(S') \leqslant 5k^4\varepsilon$.

По схеме S' построим схему $\psi(S')$ (см. теорему 1). Оценим ненадёжность $P(\psi(S'))$ схемы $\psi(S')$ по формуле (3), учитывая условие $\varepsilon \leqslant \varepsilon_0$:

$$P(\psi(S')) < \varepsilon + 2^{2^k}(1/2\varepsilon^2 + 5/4k^4\varepsilon^2 + (5k^4\varepsilon)^2) \leqslant \varepsilon + \varepsilon^2 2^{2^k}(1/2 + 25k^8 + 1{,}25k^4) \leqslant \\ \leqslant \varepsilon + 2^{2^k}\varepsilon^2 \cdot 27k^8 \leqslant 2\varepsilon.$$

Здесь предпоследнее неравенство получено с учётом замечания 1, п. 1, а.

2. И н д у к т и в н ы й п е р е х о д. Допустим, что утверждение верно для функций $f(\tilde{x}^{n-1})$, $n \geqslant 2$. Докажем его для функций $f(\tilde{x}^n)$. Разложим функцию $f(\tilde{x}^n)$ по переменной x_n:

$$f(x_1, \ldots, x_{n-1}, x_n) = J_0(x_n)\&f(x_1, \ldots, x_{n-1}, 0) \vee J_1(x_n)\&f(x_1, \ldots, x_{n-1}, 1) \vee \ldots \\ \vee J_{k-1}(x_n)\&f(x_1, \ldots, x_{n-1}, k-1).$$

Реализуем функцию $f(\tilde{x}^n)$ схемой C (рис. 1).

На рис. 1 схема S_i, $i \in E_k$, реализует функцию $f_i = f(x_1, \ldots, x_{n-1}, i)$. Функции $f_0, f_1, \ldots, f_{k-1}$, согласно индуктивному предположению, можно реализовать схемами с ненадёжностью меньше 2ε, поэтому будем считать, что $P(S_i) < 2\varepsilon$, $i \in E_k$. Схему T возьмём из леммы 3. Она содержит не более $k(k-1)^2$ элементов и реализует все функции $J_i(x_n)$. В качестве схемы $S_\vee$ возьмём схему из k элементов (лемма 1), реализующую двухместный максимум, а в качестве схемы $S_\&$, реализующей двухместный минимум, возьмём схему, которая содержит менее $3k^3+4{,}5k^2-6{,}5k$ элементов (лемма 4).

В схеме C выделим подсхему A, выход которой является выходом схемы C, а на входы подаются $f_0, f_1, \ldots, f_{k-1}, x_n$. Схема A содержит не более

$$k(k-1)^2+k(3k^3+4{,}5k^2-6{,}5k)+k(k-1) = 3k^4+5{,}5k^3-7{,}5k^2 < 3k^4+5{,}5k^3 < 5k^4$$

элементов, поэтому получаем неравенство $P(A) \leqslant 5k^4\varepsilon$.

Если схема A исправна, то для реализации функции f она использует значение одной из схем $S_0, S_1, \ldots, S_{k-1}$, реализующих соответственно функции $f_0, f_1, \ldots, f_{k-1}$. Поэтому $P(C) < P(A) + 2\varepsilon < (5k^4+2)\varepsilon \leqslant 5{,}1k^4\varepsilon$ (см. замечание 1, п. 1, б).

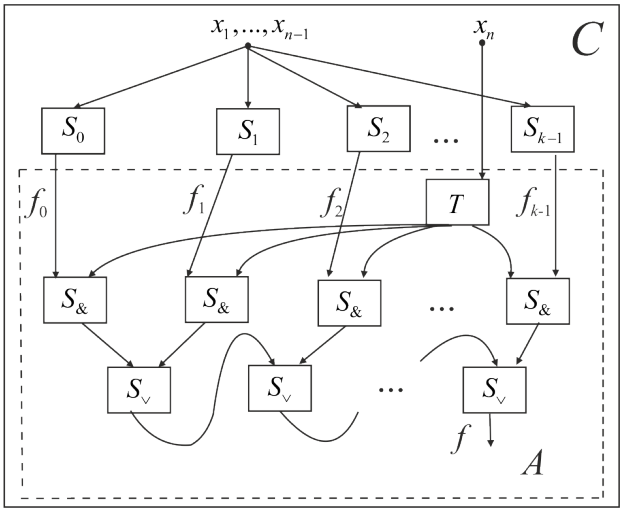

Рис. 1. Схема C

По схеме C построим схему $\psi(C)$ (см. теорему 1). Оценим ненадёжность $P(\psi(C))$ схемы $\psi(C)$ по формуле (3):

$$P(\psi(C)) < \varepsilon + 2^{2^k}(1/2 + (5{,}1k^4)^2 + 2^{-2} \cdot 5{,}1k^4)\varepsilon^2 < \varepsilon + 2^{2^k}(1/2 + 26{,}01k^8 + 1{,}275k^4)\varepsilon^2 < \\ < \varepsilon + 2^{2^k} \cdot 27k^8\varepsilon^2 \leqslant 2\varepsilon.$$

Здесь предпоследнее неравенство получено с учётом замечания 1, п. 1, в.

Следовательно, схема $P(\psi(C))$ – искомая схема D'. ■

Теорема 3. Любую функцию $f \in P_k$ можно реализовать такой схемой D, что $P(D) < \varepsilon + c_1(k)\varepsilon^2$ при всех $\varepsilon \in (0, \varepsilon_0]$, где $\varepsilon_0 = 2^{-2^k}/(27k^8)$, $c_1(k) = 5 \cdot 2^{2^k}$.

Доказательство. По теореме 2 любую функцию $f \in P_k$ можно реализовать схемой D' с ненадёжностью $P(D') < 2\varepsilon$. По схеме D' построим схему $\psi(D')$ и оценим её ненадёжность $P(\psi(D'))$ по формуле (3):

$$P(\psi(D')) < \varepsilon + \varepsilon^2 2^{2^k}(1/2 + 2 \cdot 1/4 + 4) = \varepsilon + 5 \cdot 2^{2^k}\varepsilon^2.$$

Схема $\psi(D')$ — искомая схема D. ■

Из теоремы 3 следует, что любую функцию из P_k можно реализовать схемой, ненадёжность которой асимптотически (при $\varepsilon \to 0$) не больше ε.

Пусть $K(n)$ — множество функций k-значной логики, каждая из которых зависит от переменных $x_1, \ldots, x_n$ $(n \geqslant 1)$ и отлична от функций $0, k-1, x_1, \ldots, x_n$. Обозначим $K = \bigcup\limits_{n=1}^{\infty} K(n)$. Очевидно, что $|K(n)| = k^{k^n} - n - 2$, а значит, класс $K(n)$ содержит почти все функции из множества $P_k(n)$, поскольку $\lim\limits_{n\to\infty} \dfrac{k^{k^n} - n - 2}{k^{k^n}} = 1$.

Справедлива теорема о нижней оценке ненадёжности схем, реализующих функции из класса K.

Теорема 4. Пусть функция $f \in K$. Тогда для любой схемы S, реализующей f, верно неравенство $P(S) \geqslant \varepsilon$ при всех $\varepsilon \in (0, 1/2)$.

Доказательство. Для доказательства достаточно выделить подсхему из одного элемента, выход которого является выходом схемы, и оценить вероятность появления 0 на любом ненулевом входном наборе схемы.

Действительно, пусть $f \in K(n)$, S — любая схема, реализующая f. Заметим, что схема S содержит хотя бы один элемент. Пусть E — функциональный элемент схемы S, выход которого является выходом схемы S.

Поскольку $f \not\equiv 0$, найдутся такие значение $i \in E_k \setminus \{0\}$ и входной набор $\tilde{a}^n$, что $f(\tilde{a}^n) = i$. Вычислим вероятность $P_0(S, \tilde{a}^n)$ появления значения 0 на выходе схемы S на наборе $\tilde{a}^n$, обозначив через p_0 вероятность появления нулевого набора на входах элемента E:

$$P_0(S, \tilde{a}^n) = p_0 + (1 - p_0)\varepsilon = \varepsilon + p_0(1 - \varepsilon) \geqslant \varepsilon.$$

Теорема доказана. ■

Следовательно, любая схема, реализующая функцию $f \in K$, функционирует с ненадёжностью, которая не меньше ε. Это означает, что схема, реализующая функцию $f \in K$ и удовлетворяющая условиям теоремы 3, является асимптотически оптимальной по надёжности и функционирует с ненадёжностью, асимптотически равной ε при $\varepsilon \to 0$.

Поскольку ненадёжности двойственных (относительно перестановки, порождаемой функцией $N(x) = k - 1 - x$, которую называют отрицанием Лукашевича) схем равны [8], утверждение, доказанное в базисе B для ненадёжности схемы, реализующей функцию f, при неисправностях типа 0 на выходах элементов, верно в базисе B^* для ненадёжности двойственной схемы, реализующей функцию f^*, при неисправностях типа $k - 1$ на выходах элементов. Следовательно, в базисе B^* при неисправностях типа $k - 1$ на выходах элементов почти любую функцию k-значной логики можно реализовать асимптотически оптимальной по надёжности схемой, которая функционирует с ненадёжностью, асимптотически равной ε при $\varepsilon \to 0$.

Заметим, что: 1) функции $x_1, \ldots, x_n$ можно реализовать абсолютно надёжно, не используя функциональные элементы; 2) $(x_i)^* = x_i$ $(i \in \{1, ..., n\})$; 3) $0^* = k - 1$, $(k - 1)^* = 0$.

2. Верхняя оценка ненадёжности схем при неисправностях типа $k - 1$

Будем считать, что элементы базиса $B = \{V_k\}$ с вероятностью ε подвержены однотипным константным неисправностям типа $k - 1$ на выходах элементов.

Теорема 5 [5]. Пусть f — произвольная функция из P_k, S — любая схема, реализующая f, $P(S)$ — ненадёжность схемы S. Тогда можно построить схему $\psi(S)$, которая реализует функцию f с ненадёжностью

$$P(\psi(S)) \leqslant 3\varepsilon + (2^{2^k - 1} - 2^k)\varepsilon^2 + 2^{2k}\varepsilon P(S) + (2^{2^k} - 2^k - 1)P^2(S). \tag{4}$$

В теореме 5 установлено рекуррентное соотношение между ненадёжностью схемы $\psi(S)$ и ненадёжностью исходной схемы S. В соотношении (4) отбросим отрицательные слагаемые в правой части и учтём замечание 1, п. 2, получим неравенство

$$P(\psi(S)) \leqslant 3\varepsilon + 2^{2^k}(1/2\varepsilon^2 + 1/4\varepsilon P(S) + P^2(S)). \tag{5}$$

Теорема 6. Любую функцию $f \in P_k$ можно реализовать схемой D', ненадёжность которой $P(D') < 4\varepsilon$ при всех $\varepsilon \in (0, \varepsilon_0]$, где $\varepsilon_0 = 2^{-2^k}/(27k^8)$.

Доказательство. Доказательство проведем методом индукции по числу переменных функции $f(\tilde{x}^n)$.

1. Б а з а и н д у к ц и и: $n = 1$. Представим функцию $f(x)$ в виде (аналог совершенной ДНФ) [6]

$$f(x) = J_0(x)\&f(0) \vee J_1(x)\&f(1) \vee \ldots \vee J_{k-1}(x)\&f(k - 1).$$

Промоделируем эту формулу схемой S', используя схемы из лемм 2–4, и получим $L(S') < 5k^4$, $P(S') \leqslant 5k^4\varepsilon$.

По схеме S' построим схему $\psi(S')$ (см. теорему 5). Оценим ненадёжность $P(\psi(S'))$ схемы $\psi(S')$ по формуле (5), учитывая условие $\varepsilon \leqslant \varepsilon_0$:

$$P(\psi(S')) < 3\varepsilon + 2^{2^k}(1/2\varepsilon^2 + 1/4 \cdot 5k^4\varepsilon^2 + (5k^4\varepsilon)^2) = 3\varepsilon + \varepsilon^2 2^{2^k}(1/2 + 25k^8 + 1,25k^4) \leqslant$$
$$\leqslant 3\varepsilon + 2^{2^k} \cdot 27k^8\varepsilon^2 \leqslant 4\varepsilon.$$

Здесь предпоследнее неравенство получено с учётом замечания 1, п. 1, а.

2. И н д у к т и в н ы й п е р е х о д. Допустим, что утверждение теоремы верно для функций $f(\tilde{x}^{n-1})$ $(n \geqslant 2)$. Докажем его для функций $f(\tilde{x}^n)$. Разложим $f(\tilde{x}^n)$ по переменной x_n:

$$f(x_1, \ldots, x_{n-1}, x_n) = J_0(x_n)\& f(x_1, \ldots, x_{n-1}, 0) \vee J_1(x_n)\& f(x_1, \ldots, x_{n-1}, 1) \vee \ldots$$
$$\vee J_{k-1}(x_n)\& f(x_1, \ldots, x_{n-1}, k-1).$$

Реализуем функцию $f(\tilde{x}^n)$ схемой C (см. рис. 1).

Функции $f_0, f_1, \ldots, f_{k-1}$, согласно индуктивному предположению, можно реализовать схемами с ненадёжностью меньше 4ε, поэтому будем считать, что $P(S_i) < 4\varepsilon$, $i \in E_k$. Аналогично доказательству теоремы 2 получаем, что схема A содержит не более $5k^4$ элементов, поэтому верно неравенство $P(A) \leqslant 5k^4\varepsilon$.

Если схема A исправна, то для реализации функции f она использует значение одной из схем $S_0, S_1, \ldots, S_{k-1}$, реализующих соответственно функции $f_0, f_1, \ldots, f_{k-1}$. Поэтому $P(C) < P(A) + 4\varepsilon < (5k^4 + 4)\varepsilon < 5{,}1k^4\varepsilon$ (см. замечание 1, п. 1, г).

По схеме C построим схему $\psi(C)$ (см. теорему 5). Оценим ненадёжность $P(\psi(C))$ схемы $\psi(C)$ по формуле (5):

$$P(\psi(C)) < 3\varepsilon + 2^{2^k}(1/2 + (5{,}1k^4)^2 + 1/4 \cdot 5{,}1k^4)\varepsilon^2 < 3\varepsilon + 2^{2^k} \cdot 27k^8\varepsilon^2 \leqslant 3\varepsilon + \varepsilon = 4\varepsilon.$$

Здесь предпоследнее неравенство получено ввиду замечания 1, п. 1, в.

Следовательно, схема $P(\psi(C))$ — искомая схема D'. ■

Теорема 7. Любую функцию $f \in P_k$ можно реализовать такой схемой D, что $P(D) < 3\varepsilon + c_2(k)\varepsilon^2$ при всех $\varepsilon \in (0, \varepsilon_0]$, где $\varepsilon_0 = 2^{-2^k}/(27k^8)$, $c_2(k) = 17{,}5 \cdot 2^{2^k}$.

Доказательство. По теореме 6 любую функцию $f \in P_k$ можно реализовать схемой D' с ненадёжностью $P(D') < 4\varepsilon$. По схеме D' построим схему $\psi(D')$ и оценим её ненадёжность $P(\psi(D'))$ по формуле (5):

$$P(\psi(D')) < 3\varepsilon + \varepsilon^2 2^{2^k}(1/2 + 1/4 \cdot 4 + 16) = 3\varepsilon + 17{,}5 \cdot 2^{2^k}\varepsilon^2.$$

Схема $\psi(D')$ — искомая схема D. ■

Из теоремы 7 следует, что при однотипных константных неисправностях типа $k-1$ любую функцию из P_k можно реализовать схемой, ненадёжность которой асимптотически (при $\varepsilon \to 0$) не больше 3ε.

Справедлива теорема о нижней оценке ненадёжности схем.

Теорема 8. Пусть функция $f \in K$. Тогда для любой схемы S, реализующей f, верно неравенство $P(S) \geqslant \varepsilon$ при всех $\varepsilon \in (0, 1/2)$.

Доказательство. Достаточно выделить подсхему из одного элемента, выход которого является выходом схемы, и оценить вероятность появления $k-1$ на любом входном наборе схемы $\tilde{a}^n$, $f(\tilde{a}^n) \neq k-1$ (см. доказательство теоремы 4). ■

Следовательно, любая схема, реализующая функцию $f \in K$, функционирует с ненадёжностью, которая не меньше ε. Это означает, что схема, реализующая функцию $f \in K$ и удовлетворяющая условиям теоремы 7, функционирует с ненадёжностью, асимптотически не больше 3ε и не меньше ε при $\varepsilon \to 0$.

Поскольку ненадёжности двойственных схем равны [8], утверждение, доказанное в базисе B для ненадёжности схемы, реализующей функцию f, при неисправностях типа $k-1$ на выходах элементов, верно в базисе B^* для ненадёжности двойственной схемы, реализующей функцию f^*, при неисправностях типа 0 на выходах элементов. Следовательно, в базисе B^* при неисправностях типа 0 на выходах элементов почти любую функцию k-значной логики можно реализовать схемой, которая функционирует с ненадёжностью, асимптотически не больше 3ε и не меньше ε при $\varepsilon \to 0$.

Выводы

В полном базисе, состоящем из функции Вебба, при неисправностях типа 0 на выходах элементов и в двойственном ему базисе при неисправностях типа $k-1$ на выходах элементов почти любую функцию можно реализовать асимптотически оптимальной по надёжности схемой, функционирующей с ненадёжностью, асимптотически (при $\varepsilon \to 0$) равной ε.

В полном базисе, состоящем из функции Вебба, при неисправностях типа $k-1$ на выходах элементов и в двойственном ему базисе при неисправностях типа 0 на выходах элементов почти любую функцию можно реализовать асимптотически оптимальной по надёжности схемой, функционирующей с ненадёжностью, асимптотически (при $\varepsilon \to 0$) не больше 3ε и не меньше ε.

ЛИТЕРАТУРА

1. *Pippenger N.* On networks of noisy gates // 26 Symp. Foundation on Computer Science. 21–23.10.1985, Portland. P. 30–38.
2. *Ортюков С. И.* Об избыточности реализации булевых функций схемами из ненадежных элементов // Труды семинара по дискретной математике и ее приложениям (Москва, 27–29 января 1987 г.). М.: Изд-во МГУ, 1989. С. 166–168.
3. *Uhlig D.* Reliable networks from unreliable gates with almost minimal complexity // LNCS. 1987. V. 278. P. 462–469.
4. *Алехина М. А., Барсукова О. Ю.* Синтез надежных схем в базисе, состоящем из функции Вебба, в P_k // Изв. вузов. Математика. 2019. № 7. С. 15–28.
5. *Алехина М. А.* Рекуррентные соотношения для ненадежностей схем при однотипных константных неисправностях типов 0 и $k-1$ в базисе, состоящем из функции Вебба, в P_k // Изв. вузов. Поволжский регион. Физико-математические науки. 2018. № 4. С. 25–30.
6. *Яблонский С. В.* Введение в дискретную математику. М.: Наука, 2001. 385 с.
7. *Редькин Н. П.* Надежность и диагностика схем. М.: Изд-во МГУ, 1992. 192 с.
8. *Алехина М. А.* Надежность двойственных схем в P_k // Изв. вузов. Поволжский регион. Физико-математические науки. 2017. № 1. С. 3–13.

REFERENCES

1. *Pippenger N.* On networks of noisy gates. 26 Symp. Foundation on Computer Science, 21–23.10.1985, Portland, pp. 30–38.

2. *Ortyukov S. I.* Ob izbytochnosti realizatsii bulevykh funktsiy skhemami iz nenadezhnykh ehlementov [On the redundancy of the implementation of Boolean functions by circuits from unreliable elements]. Proc. Workshop Discr. Math. and its Appl., Moscow, MSU Publ., 1989. pp. 166–168. (in Russian)
3. *Uhlig D.* Reliable networks from unreliable gates with almost minimal complexity. LNCS, 1987, vol. 278, pp. 462–469.
4. *Alekhina M. A. and Barsukova O. Yu.* Sintez nadezhnyh skhem v bazise, sostoyashchem iz funkcii Vebba, v P_k [The synthesis of reliable circuit in the basis consisting of the Webb function, in P_k]. Russian Mathematics, 2019, no. 7, pp. 15–28. (in Russian)
5. *Alekhina M. A.* Rekurrentnye sootnosheniya dlya nenadezhnostej skhem pri odnotipnyh konstantnyh neispravnostyah tipov 0 i $k-1$ v bazise, sostoyashchem iz funkcii Vebba, v P_k [Recurrent relations for unreliability of circuits under single-type constant faults of types 0 and $k-1$ in the basis consisting of the Webb function in P_k]. University Proc. Volga Region. Physical and Mathematical Sciences, 2018, vol. 4, pp. 25–30. (in Russian)
6. *Yablonskii S. V.* Vvedenie v diskretnuyu matematiku [Introduction into Discrete Mathematics]. Moscow, Nauka Publ., 2001. 385 p. (in Russian)
7. *Redkin N. P.* Nadezhnost' i diagnostika skhem [Reliability and Diagnostics of Circuits]. Moscow, MSU Publ., 1992. 192 p. (in Russian)
8. *Alekhina M. A.* Nadezhnost' dvojstvennyh skhem v P_k [Reliability of dual circuit in P_k]. University Proc. Volga Region. Physical and Mathematical Sciences, 2017, vol. 1, pp. 3–13. (in Russian)

ПРИКЛАДНАЯ ТЕОРИЯ КОДИРОВАНИЯ

УДК 004.056.5

О ПРИМЕНИМОСТИ АЛГЕБРОГЕОМЕТРИЧЕСКИХ КОДОВ L-КОНСТРУКЦИИ КАК КОДОВ ЗАЩИТЫ ОТ КОПИРОВАНИЯ

Д. В. Загуменнов*, В. В. Мкртичян**

**Южный федеральный университет, г. Ростов-на-Дону, Россия*
***ФГАНУ НИИ «Спецвузавтоматика», г. Ростов-на-Дону, Россия*

Схемы широковещательного шифрования используются для защиты легально тиражируемой цифровой продукции от несанкционированного копирования. В схемах распространитель тиражирует данные свободно в зашифрованном виде, а для расшифрования выдаёт каждому легальному пользователю уникальный набор ключей, по которому можно однозначно определить пользователя, его получившего, и далее определить источник несанкционированного распространения. Для организации схем широковещательного шифрования используются методы, основанные на использовании помехоустойчивых кодов с большим кодовым расстоянием и быстрых алгоритмов списочного декодирования. В работе рассматривается возможность использования в этих схемах алгеброгеометрических кодов (АГ-кодов) L-конструкции и списочных декодеров Судана — Гурусвами. Рассмотрена проблема построения таких кодов. В ходе исследования получены достаточные условия применимости АГ-кодов. Представлен алгоритм построения одноточечного АГ-кода, применимого в схемах широковещательного шифрования, обоснована его корректность, приведён пример работы.

Ключевые слова: *помехоустойчивое кодирование, схемы специального широковещательного шифрования, алгеброгеометрические коды.*

DOI 10.17223/20710410/44/6

ON APPLICATION OF ALGEBRAIC GEOMETRY CODES OF L-CONSTRUCTION IN COPY PROTECTION

D. V. Zagumennov*, V. V. Mkrtichyan**

**Southern Federal University, Rostov-on-Don, Russia*
***FGANU NII «Specvuzavtomatika», Rostov-on-Don, Russia*

E-mail: zagumionnov.denis@yandex.ru

Traceability schemes which are applied to the broadcast encryption can prevent unauthorized parties from accessing the distributed data. In a traceability scheme, a distributor encrypts the data and gives each authorized user a unique key suit to decrypt the data. This suit uniquely identifies the recipient and therefore allows the tracing of the source of an unauthorized redistribution. A widely used approach to the constructing good traceability scheme is the use of error-correcting codes with a suitable minimum distance and efficient decoding algorithms. The paper deals with the usage of algebraic geometry codes (AG-codes) of L-construction and Sudan — Guruswami

list decoding algorithms in these schemes. We suggest the problem of constructing traceability AG-codes and obtain sufficient conditions for applying them. Let $C \subset \mathbb{F}_q^n$ be the algebraic geometry code constructed using curve of genus g and divisor of degree α. Firstly, if $c < \sqrt{n/\alpha}$, then C is a traceability code when the number of attackers is a maximum of c. Secondly, if $\alpha \geqslant \log_q N + g - 1$, then C can be used to build traceability schemes maintaining N users. Finally, we obtain several cumbersome bounds on the number of intruders within which it is possible to use Sudan — Guruswami hard- and soft- decision list decoding algorithms for tracing the unauthorized redistribution source. Based on these derived conditions and some other lemmas, the algorithm for suitable one-point AG-code construction is presented. The algorithm can be polynomially reduced to the Riemann — Roch space basis construction problem. Much attention is given to the algorithm validity and its sample execution. Besides, the paper gives a brief description of AG-codes and Sudan — Guruswami hard- and soft- decision list decoding algorithms.

Keywords: *error-correcting codes, traceability schemes, algebraic geometry codes.*

Введение

Рассматривается перспективный способ защиты легально тиражируемой цифровой продукции от несанкционированного копирования, называемый схемой специального широковещательного шифрования (ССШШ) [1]. В ССШШ данные тиражируются свободно в зашифрованном виде, а каждому легальному пользователю выдаётся уникальный набор ключей. В случае обнаружения нелегального использования этого набора его владелец может быть идентифицирован контроллером. В ССШШ допускаются атаки следующего вида: легальные пользователи объединяются в коалиции с целью комбинирования ключей из своих наборов и конструирования новых с целью последующего несанкционированного расшифрования данных. Для борьбы с подобными атаками в [1 – 5] предложен метод обнаружения членов коалиций, основанный на использовании некоторых классов линейных кодов и являющийся эффективным при применении быстрых алгоритмов списочного декодирования. Под эффективностью понимается существование алгоритма, позволяющего определить пользователей из коалиции злоумышленников за полиномиальное (от длины кода) время работы.

Интересной представляется задача определения, являются ли алгеброгеометрические коды (АГ-коды) L-конструкции [6] и списочные методы их декодирования [7, 8] пригодными для эффективного использования в ССШШ. Отметим актуальность рассмотрения АГ-кодов. Во-первых, АГ-коды обобщают многие другие классы линейных помехоустойчивых кодов [9, п. 4.3.1–4.3.3]. Во-вторых, АГ-коды обладают большим минимальным кодовым расстоянием [9, теорема 4.1.1], что может положительно сказаться на выполнении достаточного условия возможности использования линейных кодов в ССШШ (см. п. 1.2). В-третьих, длина АГ-кодов в общем случае не ограничена мощностью поля, в отличие от кодов Рида — Соломона и Рида — Маллера, использование которых исследовано в [4, 5] и [8, Tracing Algorithm, B. Example], а ограничена лишь мощностью кривой, на которой строится код и которая может быть существенно больше, чем мощность поля [9, теорема 3.1.25]. Это может позволить строить более эффективные относительно затрат памяти ССШШ (см. п. 2.4).

Основное внимание в работе уделено формализации построения АГ-кодов, которые можно использовать в ССШШ. Построенные согласно полученным результатам и составленным рекомендациям коды далее можно использовать в различных вариа-

циях схем, например в схеме из [8], основанной на использовании мягкого списочного декодера Судана — Гурусвами.

В работе представлено доказательство утверждения о верхней границе для максимальной мощности коалиции злоумышленников, в пределах которой АГ-коды возможно применять в ССШШ. Получены утверждения о таких границах, в пределах которых возможно применение в ССШШ алгоритмов списочного декодирования [7, 8]. Исследована связь между этими утверждениями. На основе этих границ построен концептуальный алгоритм построения АГ-кода L-конструкции, применимого в ССШШ.

1. Предварительные сведения

Пусть $\mathbb{F}_q$ — конечное поле мощности q, $C \subseteq \mathbb{F}_q^n$ — линейный код, n — длина, k — размерность, d — минимальное кодовое расстояние кода C. Коды с параметрами n, k, d, определённые над полем $\mathbb{F}_q$, будем называть $[n, k, d]_q$-кодами. Для $x, y \in \mathbb{F}_q^n$ будем обозначать $\mathrm{d}(x, y) = |\{i \in \{1, \ldots, n\} : x_i \neq y_i\}|$.

1.1. Схемы специального широковещательного шифрования

В [10] представлены способы защиты легально тиражируемой цифровой продукции от несанкционированного копирования, называемые схемами широковещательного шифрования. В [11] эти подходы описаны и проанализированы более подробно.

ССШШ [1] — это усовершенствованная схема 4.1 из [11], основанная на использовании некоторых классов линейных кодов. Схемы из [11] будем называть классическими схемами широковещательного шифрования (КСШШ). В настоящей работе рассматривается именно ССШШ.

В ССШШ в качестве открытого ключа используется матрица $M = (E_{k_{ij}}(s_i))_{j=1,\ldots,q}^{i=1,\ldots,n}$, где $q = p^r$; p простое; $r, n \in \mathbb{N}$; s_i — разделённые части некоторого секрета s. Каждому пользователю схемы выдаётся уникальный набор ключей $(k_{1,i_1}, \ldots, k_{n,i_n})$ и вектор-номер $(i_1, \ldots, i_n)$, $i_k \in \{1, \ldots, q\}$, $k = 1, \ldots, n$, необходимый для того, чтобы пользователь знал, какие именно части матрицы M он может расшифровать на своих ключах. Проведя расшифрование и получив все части s_i секрета s, пользователь может получить доступ к распространяемой цифровой продукции.

В схемах широковещательного шифрования допускаются атаки следующего вида: легальные пользователи объединяются в коалиции, комбинируют свои вектор-номера и соответственно наборы ключей, получая таким образом новый «пиратский» набор, который может быть использован с целью несанкционированного распространения и последующего расшифрования данных.

Для этого в [1] в качестве вектор-номеров предложено использовать векторы некоторого кода C над полем $\mathbb{F}_q$. В [2, 3] описаны классы кодов, используя которые, возможно бороться с подобными коалиционными атаками, в частности класс c-ТА-кодов.

Далее будем использовать следующие обозначения: N — число легальных пользователей; c — максимальная мощность коалиции злоумышленников; n — длина используемого в ССШШ линейного кода; q — мощность поля, над которым построен используемый код.

Определение 1. ССШШ с N легальными пользователями, корректно функционирующие при атаках коалиции злоумышленников мощности не более c, будем обозначать (N, c)-ССШШ.

В [11] также представлены КСШШ, в которых существует вероятность объявить невиновного пользователя участником коалиции злоумышленников (схемы с секретом, п. 4.3 и 4.4). Далее будем сравнивать ССШШ только с теми КСШШ, в которых,

как и в ССШШ, гарантировано выявление хотя бы одного пользователя из коалиции злоумышленников (открытые схемы, п. 4.1 и 4.2)

В таблице приведено сравнение (теоремы 1 и 2 из [11]) ССШШ с этими КСШШ, а также с тривиальной схемой, когда секрет шифруется на отдельных ключах и рассылается в таком виде всем пользователям. Под длиной ключа понимается длина всей ключевой информации, хранящейся у пользователя, то есть суммарная длина вектор-номера и набора ключей. Под затратами памяти понимается размер матрицы M, под сложностью расшифрования — минимальное число операций расшифрования, которые предстоит выполнить пользователю для получения распространяемой информации в расшифрованном виде.

Через R обозначим отношение числа бит, которые необходимо использовать для представления в памяти элемента поля $\mathbb{F}_q$, к числу бит, которые необходимо использовать для представления в памяти ключа выбранной шифросистемы. Будем полагать, что части s_i разделённого секрета s имеют ту же длину, что и s. Такую схему разделения секрета легко организовать, если секрет s выбран из некоторой аддитивной группы G. В этом случае s можно разделить следующим образом: $s = \bigoplus_{i=1}^{n} s_i$.

Сравнение ССШШ с КСШШ и тривиальной схемой

Параметр	Длина ключевой информации	Затраты памяти	Сложность расшифрования
Единица измерения	Длина ключа в битах	Длина зашифованного секрета в битах	Число операций расшифрования
Тривиальная схема	1	N	1
Открытая одноуровневая схема	$4c^2 \log N$	$8c^4 \log N$	$4c^2 \log N$
Открытая двухуровневая схема	$8/3 \cdot c^2 \log^2 c \log(\mathrm{e}N/c))$	$32/3 \cdot \mathrm{e}c^3 \log^4 c \log(\mathrm{e}N/c)$	$8/3 \cdot c^2 \log^2 c \log(\mathrm{e}N/c))$
ССШШ	$n(1+R)$	nq	n

Единицей измерения в первом столбце выбрана длина ключа шифрсистемы. Для ССШШ пользователю потребуется набор из n таких ключей, а также вектор-номер, состоящий из n элементов поля $\mathbb{F}_q$. Легко проверить, что эта информация займёт $n(1+R)$ длин ключей.

Единицей измерения во втором столбце выбрана длина зашифрованного секрета, так как используемая в качестве открытого ключа матрица M состоит из зашифрованных на различных ключах частей разделённого секрета. Так как части разделённого секрета имеют ту же длину, что и сам секрет, матрица M занимает в памяти nq длин зашифрованного секрета.

Единицей измерения в третьем столбце является число операций расшифрования. Для получения секрета s необходимо выполнить n расшифрований. После i-го расшифрования получим одну из частей s_i разделённого секрета, а после всех расшифрований восстановим s.

Из таблицы видно, что параметры КСШШ и тривиальной схемы, в том числе объём затрат памяти, зависят от величин N и c, в то время как параметры ССШШ зависят от параметров кода: длины кода n и мощности поля q. Таким образом, сразу сделать вывод о большей эффективности той или иной схемы не удастся. Однако из таблицы можно сформулировать следующее утверждение. Оно понадобится при дальнейшем построении ССШШ, более эффективных, чем тривиальная схема и КСШШ.

Утверждение 1. Пусть $C - [n, k, d]_q$-код, используемый для построения (N, c)-ССШШ. ССШШ является более эффективной относительно затрат по памяти, чем КСШШ и тривиальная схема, если $nq < \min\{N, 8c^4 \log N, 32/3 \cdot \mathrm{e}c^3 \log^4 c \log(\mathrm{e}N/c)\}$.

Доказательство. Прямо следует из таблицы. ■

Определение 2. Нижним порогом затраты памяти для ССШШ будем называть величину $B(N, c) = \min\{N, 8c^4 \log N, 32/3 \cdot \mathrm{e}c^3 \log^4 c \log(\mathrm{e}N/c)\}$.

Будем сравнивать тривиальную схему и КСШШ с ССШШ только относительно затрат памяти, то есть относительно размера матрицы M, и не будем сравнивать схемы относительно длины ключевой информации и сложности расшифрования. Ясно, что с точки зрения двух последних параметров наиболее эффективной является тривиальная схема, где пользователю понадобится только один ключ и одна операция расшифрования. При этом, однако, тривиальная схема показывает большую неэффективность с точки зрения затрат памяти. Большой размер открытого ключа создаёт большую нагрузку на канал распространения данных, что существенно осложняет техническую реализацию этой схемы. Поэтому сравнение схем относительно затрат памяти представляется наиболее интересным.

Построение ССШШ, более эффективных по всем рассмотренным в таблице параметрам, представляется темой отдельного исследования. Для этого нужно сформулировать утверждения об остальных параметрах, аналогичные утверждению 1, и следить за их выполнением при организации схемы.

Утверждение 2. Пусть $C - [n, k, d]_q$-код. Код C возможно использовать для организации (N, c)-ССШШ, если выполнено условие $q^k \geqslant N$.

Доказательство. Для организации (N, c)-ССШШ каждому из N пользователей необходимо выдать вектор-номер, являющийся кодовым словом кода C. Значит, количество кодовых слов не должно быть меньше числа пользователей. У кода C всего q^k кодовых слов. Получаем неравенство $q^k \geqslant N$. ■

1.2. Коды защиты от копирования

Пусть $c \in \mathbb{N} \setminus \{1\}$. Коалицией кода C назовём множество $C_0 = \{u^{(1)}, u^{(2)}, \ldots, u^{(c)}\}$, где $u^{(i)} \in C$. Число c будем называть мощностью коалиции, а множество коалиций кода мощности не больше c будем обозначать $\mathrm{coal}_c(C)$. Множеством потомков коалиции C_0 назовём множество

$$\mathrm{desc}(\mathrm{C}_0) = \{(y_1, y_2, , \ldots, y_n) \in \mathbb{F}_q^n : y_i = u_i^{(j)},\ j \in \{1, \ldots, c\}\}.$$

Определение 3 [2, определение 1.1]. Будем называть линейный код c-ТА-кодом, если выполняется следующее условие:

$$\forall\ C_0 \in \mathrm{coal}_c(C)\ \ \forall\ v \in (C \setminus C_0)\ \ \forall\ y \in \mathrm{desc}(\mathrm{C}_0)\ \ \exists\ \omega \in C_0\ \mathrm{d}(w, y) < \mathrm{d}(v, y).$$

Утверждение 3 [3, теорема 5]. Пусть C — код длины n с минимальным кодовым расстоянием d, $c \in \mathbb{N} \setminus \{1\}$. Если код C удовлетворяет условию $d > n - n/c^2$, то C является c-ТА-кодом, причём если $C_0 \in \mathrm{coal}_c(C)$ и $w \in \mathrm{desc}(C_0)$, то

$$1)\ \exists\ y \in C_0\ (\mathrm{d}(w, y) \leqslant n - n/c);\quad 2)\ \forall\ v \in C \setminus C_0\ (\mathrm{d}(w, v) > n - n/c).$$

Под радиусом работы списочного декодера будем понимать максимальное число исправляемых им ошибок.

Следствие 1. Пусть C — код длины n с минимальным кодовым расстоянием d, $c \in \mathbb{N} \setminus \{1\}$. Рассмотрим списочный декодер этого кода с радиусом работы r. Тогда достаточные условия применимости кода и декодера в ССШШ выглядят следующим образом: $d > n - n/c^2$, $r \geqslant n - n/c$.

Условие c-ТА, наличие подходящего в смысле следствия 1 списочного декодера, а также условия из утверждений 1 и 2 являются достаточными условиями того, что код может быть использован для построения (N, c)-ССШШ. В работе исследована возможность использования АГ-кодов L-конструкции [6, 9, 12] и методов списочного декодирования [7, 8] для построения (N, c)-ССШШ.

1.3. Алгеброгеометрические коды

Рассмотрим концептуальный способ построения АГ-кода L-конструкции над $\mathbb{F}_q$. Пусть $m \in \mathbb{N} \setminus \{1\}$, $\mathbb{F}_q[x_1, \ldots, x_m]$ — кольцо многочленов от m переменных с коэффициентами из поля $\mathbb{F}_q$. Неприводимость многочлена $f \in \mathbb{F}_q[x_1, \ldots, x_m]$ далее рассматривается над любыми алгебраическими расширениями поля $\mathbb{F}_q$ вплоть до его алгебраического замыкания, если это не оговаривается особо.

Определение 4. Пусть $f \in \mathbb{F}_q[x_1, x_2]$ — неприводимый многочлен. Нули этого многочлена являются точками двумерного аффинного пространства $\mathbb{A}^2(\mathbb{F}_q)$ [9, разд. 2.1.1, с. 106]. Их объединение будем называть плоской аффинной кривой [9, разд. 2.1.1; 12, определение 2.2].

Обозначим $\mathbb{F}_q^{\hom}[X_1, X_2, X_3]$ множество однородных многочленов из $\mathbb{F}_q[X_1, X_2, X_3]$.

Определение 5. Пусть $F \in \mathbb{F}_q^{\hom}[X_1, X_2, X_3]$ — неприводимый многочлен. Нули этого многочлена являются точками двумерного проективного пространства $\mathbb{P}^2(\mathbb{F}_q)$ [9, разд. 2.1.1, с. 106]. Их объединение будем называть плоской проективной кривой и обозначать как X_{pr}:

$$X_{\text{pr}} = \{P = (X_1 : X_2 : X_3) \in \mathbb{P}^2(\mathbb{F}_q) : F(X_1, X_2, X_3) = 0\}.$$

Количество точек на кривой X будем обозначать $|X|$. Далее везде будем считать, что кривая X_{pr} задана нулями многочлена F.

Между неприводимыми многочленами f из $\mathbb{F}_q[x_1, x_2]$ и F из $\mathbb{F}_q[X_1, X_2, X_3]$ существует взаимно-однозначное соответствие [12, с. 7–8]. Таким образом, существует также и взаимно-однозначное соответствие между аффинными и проективными кривыми. Процесс построения проективной кривой по соответствующей аффинной называется гомогенизацией.

Определение 6. Пусть $F \in \mathbb{F}_q^{\hom}[X_1, X_2, X_3]$, нули которого задают проективную кривую X_{pr}; (F) — главный идеал в $\mathbb{F}_q[X_1, X_2, X_3]$, порождённый F. Можно проверить, что

$$R = \left\{ \frac{P}{Q} : P, Q \in \mathbb{F}_q^{\hom}[X_1, X_2, X_3], \deg(P) = \deg(Q), Q \notin (F) \right\}$$

с естественными операциями умножения и сложения дробей является кольцом, а

$$I = \left\{ \frac{P}{Q} : P, Q \in \mathbb{F}_q^{\hom}[X_1, X_2, X_3], \deg(P) = \deg(Q), Q \notin (F), P \in (F) \right\}$$

— максимальным идеалом в R [12, с. 7]. Фактор-кольцо R/I в силу максимальности идеала I является полем [13, с. 80], оно называется полем рациональных функций на кривой X_{pr} и обозначается как $\mathbb{F}_q(X_{\text{pr}})$.

Очевидно, что $F \in \mathbb{F}_q[X_1, X_2, X_3]$ можно представить как

$$F = \sum_{i+j+k=\deg(F)} \alpha_{i,j,k} X_1^i X_2^j X_3^k, \quad \alpha_{i,j,k} \in \mathbb{F}_q.$$

Частной производной по переменной X_1 называется многочлен

$$F_{X_1} = \sum_{i+j+k=\deg(F)} i\alpha_{i,j,k} X_1^{i-1} X_2^j X_3^k, \quad \alpha_{i,j,k} \in \mathbb{F}_q.$$

Здесь элемент $i\alpha_{i,j,k}$ нужно понимать как $\alpha_{i,j,k}$, сложенный с самим собой i раз. Аналогично определяются частные производные F_{X_2} и F_{X_3}.

Определение 7. Плоская проективная кривая X_{pr} называется гладкой в точке $M \in X_{\mathrm{pr}}$, если хотя бы одно из чисел $F_{X_1}(M), F_{X_2}(M), F_{X_3}(M)$ не равно нулю.

Далее будем рассматривать только гладкие плоские кривые.

Кривые имеют параметр $g \in \mathbb{N} \cup \{0\}$, называемый родом. В случае плоских гладких кривых род вычисляется по известной формуле [9, следствие 2.2.8]. Пусть F — многочлен, нули которого задают кривую. Тогда

$$g = (\deg(F) - 1)(\deg(F) - 2)/2. \tag{1}$$

Из формулы (1) видно, что гладкие плоские кривые могут иметь в качестве рода число g, только если существуют решения уравнения $(x-1)(x-2) = 2g$.

Утверждение 4 [9, теорема 3.1.25]. Пусть X — аффинная или проективная кривая рода g над полем $\mathbb{F}_q$, тогда $|X| \leqslant q + 1 + g\lceil 2\sqrt{q} \rceil$.

Утверждение 5 [12, определение 2.15 и теорема 2.16]. Пусть X_{pr} — проективная кривая, $\mathbb{F}_q(X_{\mathrm{pr}})$ — поле рациональных функций на ней, $M \in X_{\mathrm{pr}}$. Тогда

$$\exists T \in \mathbb{F}_q(X_{\mathrm{pr}})\ \forall H \in \mathbb{F}_q(X_{\mathrm{pr}})\ \exists U \in \mathbb{F}_q(X_{\mathrm{pr}})(T(M) = 0\ \&\ U(M) \neq 0\ \&\ H = T^m U),$$

где $m \in \mathbb{Z}$, причём значение m не зависит от выбора элемента T.

Определение 8. Порядком $H = T^m U \in \mathbb{F}_q(X_{\mathrm{pr}})$ в точке $M \in X_{\mathrm{pr}}$ назовём значение m и будем обозначать это $\mathrm{ord}_M(H) = m$. Договоримся, что $\mathrm{ord}_M(0) = \infty$.

Замечание 1. Заметим, что в силу определения 8 порядок (в фиксированной точке) произведения функций равен сумме порядков функций-сомножителей, а порядок функции, обратной данной, равен по модулю порядку данной функции, но имеет противоположный знак.

Определение 9. Пусть M — точка на кривой X_{pr}; $L \in \mathbb{F}_q^{\mathrm{hom}}[X, Y, Z]$ — однородный многочлен степени 1, такой, что $L(M) \neq 0$; $G \in \mathbb{F}_q^{\mathrm{hom}}[X, Y, Z]$, $\deg(G) = r$. Назовём кратностью пересечения G и X_{pr} в точке M число $\mathrm{ord}_M(G/L^r)$ и обозначим её как $I(M; X_{\mathrm{pr}}; G)$.

Понятие кратности пересечения является аналогом понятия порядка для однородных многочленов.

Утверждение 6 [12, теорема 2.23]. Пусть $G \in \mathbb{F}_q^{\mathrm{hom}}[X, Y, Z]$ и кривая X_{pr} задана многочленом F. Предположим, что F не делит G. Тогда выполняется равенство

$$\sum_{M \in X_{\mathrm{pr}}} I(M; X_{\mathrm{pr}}; G) = \deg(G)\deg(F).$$

Определение 10. Дивизором D на проективной кривой X_{pr} называется формальная сумма следующего вида: $D = \sum\limits_{M \in X_{\text{pr}}} a_M M$, $a_M \in \mathbb{Z}$.

Так как множество точек на проективной кривой над конечным полем конечно, то эти суммы всегда конечны.

Определение 11. Степенью дивизора D называют число $\deg(D) = \sum a_M$, а носителем дивизора — множество $\operatorname{supp}(D) = \{M \in X_{\text{pr}} : a_M \neq 0\}$.

Определение 12. Говорят, что дивизор $D = \sum a_M M$, $a_M \in \mathbb{Z}$, $M \in X_{\text{pr}}$, эффективен, если все $a_M \geqslant 0$. Этот факт обозначается следующим образом: $D \geqslant 0$.

Дивизоры образуют абелеву группу относительно поточечного сложения.

Определение 13. Пусть $G \in \mathbb{F}_q^{\text{hom}}[X, Y, Z]$, $\deg(G) = r$. Назовём дивизором пересечения G и X_{pr} дивизор вида

$$X_{\text{pr}} \cdot G = \sum_{M \in X_{\text{pr}}} I(M; X_{\text{pr}}; G) M.$$

Определение 14. Пусть $H = P/Q \in \mathbb{F}_q(X_{\text{pr}})$, где $P, Q \in \mathbb{F}_q^{\text{hom}}[X, Y, Z]$. Дивизором H на проективной кривой X_{pr} называется дивизор

$$(H) = \sum_{M \in X_{\text{pr}}} \operatorname{ord}_M(H) M.$$

Замечание 2. Учитывая определения 8 и 9, можно проверить, что

$$(H) = X_{\text{pr}} \cdot P - X_{\text{pr}} \cdot Q.$$

Действительно, используя замечание 1, получаем

$$\operatorname{ord}_M(H) = \operatorname{ord}_M\left(\frac{P}{Q}\right) = \operatorname{ord}_M\left(\frac{P}{L^r} \cdot \frac{L^r}{Q}\right) = \operatorname{ord}_M\left(\frac{P}{L^r}\right) + \operatorname{ord}_M\left(\frac{L^r}{Q}\right) =$$
$$= \operatorname{ord}_M\left(\frac{P}{L^r}\right) - \operatorname{ord}_M\left(\frac{Q}{L^r}\right) = I(M; X_{\text{pr}}; P) - I(M; X_{\text{pr}}; Q),$$

где $L \in \mathbb{F}_q^{\text{hom}}[X, Y, Z]$; $\deg(L) = 1$; $L(M) \neq 0$; $r = \deg(P) = \deg(Q)$.

Определение 15. Зафиксируем дивизор $D = \sum a_M M$ и рассмотрим множество

$$L(D) = \{H \in \mathbb{F}_q(X_{\text{pr}}) : (H) + D \geqslant 0\}.$$

Множество $L(D)$ называется пространством функций Римана — Роха, ассоциированным с дивизором D. Пространство Римана — Роха является конечномерным векторным пространством [12, определение 2.36, теорема 2.37].

Пусть $X_{\text{pr}} \neq \varnothing$ — гладкая плоская проективная кривая, $Points = \{P_1, \ldots, P_n\} \subset X_{\text{pr}}$, $D = \sum\limits_{M \in X_{\text{pr}}} a_M M$, $a_M \in \mathbb{Z}$, — дивизор на X_{pr}, такой, что $\operatorname{supp}(D) \cap Points = \varnothing$. Построим отображение

$$Ev_{Points} : L(D) \to \mathbb{F}_q^n, \quad Ev_{Points}(h) = (H(P_1), H(P_2), \ldots, H(P_n)). \tag{2}$$

Образ этого отображения $C = \operatorname{Im}(Ev_{Points}(L(D)))$ называется АГ-кодом L-конструкции. Как видно, АГ-код зависит от выбора кривой X_{pr}, множества $Points$ и дивизора D. Дивизор D будем называть дивизором кода C.

Заметим, что при вычислении функции $H \in L(D)$ на точках из $Points$ невозможно деление на ноль. Действительно, если существует $P_i \in Points$, такая, что при вычислении H в P_i проявляется деление на ноль, это означает, что $\mathrm{ord}_{P_i}(H) < 0$. Но $P_i \notin \mathrm{supp}(D)$, так как $\mathrm{supp}(D) \cap Points = \varnothing$. Значит, в дивизоре $(H) + D$ есть слагаемое, соответствующее P_i, при котором коэффициент отрицателен. Но это невозможно, так как $(H) + D \geqslant 0$. Таким образом, предположение неверно, и $\mathrm{ord}_{P_i}(H) \geqslant 0$ для любой точки $P_i \in Points$, то есть деление на ноль невозможно.

Утверждение 7 [9, теорема 4.1.1]. Пусть X_{pr} — кривая рода g и D — дивизор, такой, что $0 < \deg(D) = \alpha < n$. Тогда АГ-код $C = \mathrm{Im}(Ev_{Points}(L(D)))$ является $[n, k, d]_q$-кодом, где $k \geqslant \alpha - g + 1$ и $d \geqslant n - \alpha$. Если $\alpha > 2g - 2$, то $k = \alpha - g + 1$.

Замечание 3. Величина $d^* = n - \alpha$ называется конструктивным расстоянием алгеброгеометрического кода C. Если $\alpha > 2g - 2$, то $d^* = n - k - g + 1$. Отсюда видно, что C является кодом с максимально допустимым расстоянием тогда и только тогда, когда $g = 0$.

Замечание 4. Естественно встаёт вопрос о вычислении базиса пространства Римана — Роха $L(D)$, являющегося, как видно по построению АГ-кода, пространством кодируемых сообщений.

При малых параметрах m, q, g базис можно попытаться вычислить вручную. Сначала необходимо найти все точки кривой. Далее, так как пространство $L(D)$ является пространством кодируемых сообщений, размерность этого пространства равна размерности k соответствующего АГ-кода. Предположим, что $\deg(D) = \alpha > 2g - 2$, тогда $k = \alpha - g + 1$. Таким образом, нужно найти k линейно независимых функций ϕ_i, таких, что $(\phi_i) + D \geqslant 0$. Для проверки этих условий нужно знать величины $\mathrm{ord}_M(\phi_i)$, где $M \in X_{\mathrm{pr}}$. Для этого можно использовать замечание 2, в частности то, что порядок функции в точке равен разности кратностей пересечений числителя и знаменателя. Кратность пересечения $I(M; X_{\mathrm{pr}}; G)$ многочлена G в точке M с кривой X_{pr} равна нулю, если $G(M) \neq 0$ (определение 9). Для определения кратности пересечения в случае $G(M) = 0$ необходимо использовать различные свойства. Например, если известны кратности пересечений для всех точек, помимо M, кратность пересечения можно найти из теоремы 6. В силу замечания 1, если известна некоторая факторизация $G = G_1 \cdot G_2 \cdot \ldots \cdot G_s$, то $I(M; X_{\mathrm{pr}}; G) = I(M; X_{\mathrm{pr}}; G_1) + I(M; X_{\mathrm{pr}}; G_2) + \ldots + I(M; X_{\mathrm{pr}}; G_s)$. Кроме того, если $\deg(G) = 1$ и $G(M) = 0$, то если G не является касательной в точке M к X_{pr}, можно утверждать, что $I(M; X_{\mathrm{pr}}; G) = 1$. Это часто используется в [12], примеры 2.33 и 2.38, 2.34 и 2.76. Уравнение касательной к кривой X_{pr}, заданной многочленом F, в точке M в проективном пространстве выглядит следующим образом [12, определение 2.11]:

$$F_{X_1}(M)X_1 + F_{X_2}(M)X_2 + F_{X_3}(M)X_3 = 0.$$

Для обработки больших параметров могут понадобиться такие алгоритмы вычисления базиса, как, например, алгоритм Хесса [14]. Алгоритм Хесса используется, например, в пакете компьютерной алгебры Magma [15].

Далее будем считать, что $0 < \deg(D) = \alpha < n$.

Замечание 5. Если дивизор D имеет вид $D = \alpha Q$, то построенный по такому дивизору код называется одноточечным АГ-кодом. В таком случае в качестве точки Q обычно берут точку вида $(M_1 : M_2 : 0), M_i \in \mathbb{F}_q$ [9, пример 4.1.5; 12, примеры 2.67 и 2.75].

1.4. Классический декодер Судана — Гурусвами

Утверждение 8 [7, теорема 27]. Пусть C — АГ-код длины n, D — дивизор кода C, $\deg(D) = \alpha$, конструктивное расстояние кода $d^* = n - k - g + 1$. Тогда существует алгоритм декодирования (классический списочный декодер Судана — Гурусвами — классический СДСГ) этого кода со сложностью, полиномиальной по n, исправляющий $r = \lfloor n - \sqrt{n(k+g-1)} \rfloor$ ошибок.

Замечание 6. Если $\alpha > 2g - 2$, то из утверждения 7 видно, что $\alpha = k + g - 1$ и $d^* = n - k - g + 1$. Таким образом, в силу утверждения 8 при выполнении условия $\alpha > 2g - 2$ существует возможность использования СДСГ.

1.5. Мягкий декодер Судана — Гурусвами

Утверждение 9 [8, с. 2 (5)]. Пусть C — АГ-код длины n над полем $\mathbb{F}_q$, d^* — конструктивное расстояние кода и для некоторых $m, l \in \mathbb{N} \cup \{0\}$ выполнено неравенство

$$\frac{l^2}{m} + \frac{(m-l)^2}{m(q-1)} + \frac{n-m}{q} \geqslant n - d^* + \varepsilon,$$

где $\varepsilon \in \mathbb{R}$ — заданный заранее безразмерный параметр допустимого отклонения, $\varepsilon > 0$. Тогда существует алгоритм декодирования (мягкий списочный декодер Судана — Гурусвами — мягкий СДСГ) этого кода со сложностью, полиномиальной по n, исправляющий $n - m$ стираний и $m - l$ ошибок.

2. Результаты

2.1. Достаточные условия применимости АГ-кодов L-конструкции и декодера Судана — Гурусвами в ССШШ

Представим границы, в пределах которых АГ-коды и мягкий и классический СДСГ могут быть применены для построения (N, c)-ССШШ.

Пусть C — АГ-код L-конструкции над $\mathbb{F}_q$, n — его длина, k — размерность, g — род кривой, на которой определён код C, D — дивизор кода C, $\deg(D) = \alpha$. Рассмотрим неравенство

$$\alpha > 2g - 2. \tag{3}$$

По утверждению 7 при выполнении условия (3) $\alpha = k + g - 1$.

Теорема 1. Код C является c-ТА-кодом, если выполняется условие

$$c < \sqrt{n/\alpha}. \tag{4}$$

Доказательство. Согласно следствию 1, для того чтобы код C являлся c-ТА-кодом, достаточно выполнения следующего условия:

$$d > n - n/c^2. \tag{5}$$

Согласно утверждению 7 и замечанию 3, $d \geqslant d^* = n - \alpha$. Решив неравенство для конструктивного расстояния $d^* > n - n/c^2$ относительно c, получим значения c, которые удовлетворяют (5). Решая, получаем

$$c < \sqrt{n/(n-d^*)} = \sqrt{n/\alpha}.$$

Таким образом, для выполнения c-ТА достаточно выполнения (4). ■

Частный случай теоремы 1 приведён в [3, теорема 6, часть 2] в случае, когда выполнено условие (3). Доказательство этого частного случая можно получить, используя доказательство теоремы 1, имея в виду, что при выполнении (3) $\alpha = k + g - 1$.

Лемма 1. Максимально возможный радиус классического СДСГ равен r, где

$$r = \begin{cases} n - \left\lceil \sqrt{n(k+g-1)} \right\rceil, & \sqrt{n(k+g-1)} \notin \mathbb{N}, \\ n - \sqrt{n(k+g-1)} - 1, & \sqrt{n(k+g-1)} \in \mathbb{N}. \end{cases}$$

Доказательство. В силу утверждения 8 максимально возможный радиус при условии $\sqrt{n(k+g-1)} \notin \mathbb{N}$ равен $r = \lfloor n - \sqrt{n(k+g-1)} \rfloor$, так как в этом случае это максимальное из натуральных чисел r, удовлетворяющих неравенству $r < n - \sqrt{n(k+g-1)}$. С учётом того, что $\lfloor -x \rfloor = -\lceil x \rceil$ и $\forall k \in \mathbb{Z}\ \forall x \in \mathbb{R}\ (\lfloor k + x \rfloor = k + \lfloor x \rfloor)$, получаем

$$\begin{aligned} r &= \left\lfloor n - \sqrt{n(k+g-1)} \right\rfloor = \left\lfloor n + -(\sqrt{n(k+g-1)}) \right\rfloor = \\ &= n + \left\lfloor -(\sqrt{n(k+g-1)}) \right\rfloor = n - \left\lceil \sqrt{n(k+g-1)} \right\rceil. \end{aligned}$$

В силу утверждения 8 максимально возможный радиус при условии $\sqrt{n(k+g-1)} \in \mathbb{N}$ равен $r = n - \sqrt{n(k+g-1)} - 1$, так как в этом случае это максимальное из натуральных чисел r, удовлетворяющее неравенству $r < n - \sqrt{n(k+g-1)}$. ■

Замечание 7. Работу декодера всегда можно организовать с максимальным радиусом декодирования, выбирая соответствующим образом управляемые параметры алгоритма классического СДСГ [7, п. 6.3.3].

Теорема 2.

1) Если $\sqrt{n\alpha} \notin \mathbb{N}$, то классический СДСГ для кода C применим для построения (N, c)-ССШШ, если выполняются условие (3) и условие

$$c \leqslant \frac{n}{\lceil \sqrt{n\alpha}\ \rceil}. \tag{6}$$

2) Если $\sqrt{n\alpha} \in \mathbb{N}$, то классический СДСГ для кода C применим для построения (N, c)-ССШШ, если выполняются условие (3) и условие

$$c \leqslant \frac{n}{\sqrt{n\alpha} + 1}. \tag{7}$$

При выполнении условия (6) выполняется и условие (4), а при выполнении (7) выполняется (6).

Доказательство. Условие (3) обосновывается замечанием 6. По утверждению 7 при выполнении этого условия $\alpha = k + g - 1$. Из леммы 1 возьмём выражение для радиуса работы классического СДСГ: если $\sqrt{n\alpha} \notin \mathbb{N}$, то $r = n - \lceil \sqrt{n\alpha} \rceil$, иначе $r = n - \sqrt{n\alpha} - 1$. По следствию 1, радиус декодера должен удовлетворять условию

$$r = n - \lceil \sqrt{n\alpha} \rceil \geqslant n - n/c,$$

что эквивалентно

$$c \leqslant \frac{n}{n - r}. \tag{8}$$

Подставив в (8) значение r при $\sqrt{n\alpha} \notin \mathbb{N}$, получим (6), а подставив r при $\sqrt{n\alpha} \in \mathbb{N}$, получим (7).

Докажем, что из (6) следует (4). Условие (4) можно переписать в виде $c \leqslant n/\sqrt{n\alpha}$. В случае $\sqrt{n\alpha} \notin \mathbb{N}$ выполняется строгое неравенство $\lceil \sqrt{n\alpha} \rceil > \sqrt{n\alpha}$, и тогда $c \leqslant \leqslant n/\lceil \sqrt{n\alpha} \rceil < n/\sqrt{n\alpha}$. Значит, (6) накладывает более сильное условие на c, чем (4), следовательно, выполнение (6) влечёт за собой и выполнение (4).

Докажем, что из (7) следует (6). При $\sqrt{n\alpha} \in \mathbb{N}$ выполняется неравенство $\sqrt{n\alpha}+1 > > \lceil \sqrt{n\alpha} \rceil$, тогда $c \leqslant n/\sqrt{n\alpha}+1 < n/\lceil \sqrt{n\alpha} \rceil$. Значит, выполнение условия (7) влечёт за собой и выполнение (6). ■

Теорема 3. Предположим, что слово-потомок коалиции злоумышленников получено без стираний. Пусть

$$D = \sqrt{n(q-1)(n(q-1) - q(n-\alpha-\varepsilon))},$$

где ε — параметр допустимого отклонения для мягкого СДСГ ($\varepsilon \in \mathbb{R}$, $\varepsilon > 0$). Тогда мягкий СДСГ для кода C применим для построения (N, c)-ССШШ, то есть гарантируется определение как минимум одного члена коалиции злоумышленников, если выполняются условие (3) и условие

$$c \geqslant \frac{n - \lfloor (n(q-1) - D)/q \rfloor - 1}{\alpha}. \tag{9}$$

Доказательство. Пусть r_{soft} — радиус декодирования мягкого СДСГ. Из теоремы 9, учитывая, что $m = n$ (так как считаем, что стираний нет), получаем неравенство

$$\frac{(n - r_{\text{soft}})^2}{n} + \frac{r_{\text{soft}}^2}{n(q-1)} \geqslant n - d^* + \varepsilon.$$

Решая это неравенство относительно r_{soft}, получаем условия на радиус декодирования для мягкого декодера. Пусть $D = \sqrt{n(q-1)(n(q-1) - q(d^* - \varepsilon))}$. Тогда

$$r_{\text{soft}} \leqslant R_1 = \frac{n(q-1) - D}{q}, \quad r_{\text{soft}} \geqslant R_2 = \frac{n(q-1) + D}{q}.$$

При достаточно больших n, в частности при $n > (d^* - \varepsilon)(q-1)/(q-2)$, величина R_2 превосходит n, следовательно, второе ограничение на r_{soft} перестаёт иметь смысл, так как $r_{\text{soft}} < n$. Таким образом, существенным остаётся только первое условие. Тогда максимально возможное значение радиуса работы мягкого декодера равно

$$r_{\text{soft}} = \lfloor R_1 \rfloor = \left\lfloor \frac{n(q-1) - D}{q} \right\rfloor.$$

На шаге 4 на первой итерации алгоритма определения злоумышленников в [8, п. «Example»], то есть на итерации, где потенциально может быть определён первый пользователь из коалиции злоумышленников, накладывается ещё одно ограничение на радиус мягкого декодера — $r_{\text{soft}} \geqslant C = n - c(k+g-1) - 1$, так как на этом шаге из полученного списочным декодером списка пользователей выбираются те слова, что имеют одинаковые значения с текущим потомком хотя бы в $c(k+g-1)+1$ позициях.

По условию теоремы выполняется условие (3), то есть $\alpha = k + g - 1$. Получаем

$$\lfloor (n(q-1) - D)/q \rfloor \geqslant n - c(k+g-1) - 1,$$
$$c \geqslant \frac{n - \lfloor (n(q-1) - D)/q \rfloor - 1}{k+g-1},$$
$$c \geqslant \frac{n - \lfloor (n(q-1) - D)/q \rfloor - 1}{\alpha}.$$

Учитывая, что для АГ-кодов $d^* = n - \alpha$ (замечание 3), величину D можно переписать в виде $D = \sqrt{n(q-1)(n(q-1) - q(n-\alpha-\varepsilon))}$. ■

Теорема 4. Код C возможно использовать для организации (N, c)-ССШШ, если выполнено условие

$$\alpha \geqslant \log_q N + g - 1. \tag{10}$$

Доказательство. Из утверждения 7 и замечания 3 видно, что если $\alpha \geqslant 2g-1$, то $k = \alpha - g + 1$. Тогда условие $N \leqslant q^k$ возможности применения кода C в (N, c)-ССШШ (утверждение 2) переписывается в виде $N \leqslant q^{\alpha-g+1}$, откуда получаем условие (10). ■

Лемма 2. Пусть C — c-ТА-АГ-код над полем $\mathbb{F}_q$, где $q = 2^m$ для некоторого натурального m; n — длина кода, D — дивизор кода C, $\deg(D) = \alpha$, конструктивное расстояние кода d^* — нечётное число. Тогда код $\hat{C}$, являющийся удлинением кода C с помощью добавления проверки чётности [18, п. 1.9(I)], также является c-ТА-кодом.

Доказательство. По замечанию 3, для кода C конструктивное расстояние $d^* = n - \alpha$. Согласно [18, п. 1.9(I)], при добавлении позиции проверки чётности к двоичному $[n, k, d]_q$-коду с нечётным расстоянием d получим $[n+1, k, d+1]_q$-код. Значит, код $\hat{C}$ как удлинение $[n, k, \geqslant n-\alpha]_q$-кода C является $[n+1, k, \geqslant n-\alpha+1]_q$-кодом. Действуя аналогично доказательству теоремы 1 и решив неравенство для конструктивного расстояния

$$d^* = n - \alpha + 1 > n + 1 - (n+1)/c^2$$

относительно c, получим значения c, которые удовлетворяют достаточному условию c-ТА (5) для кода $\hat{C}$:

$$c < \sqrt{n + 1/\alpha}. \tag{11}$$

Так как C — c-ТА-код, выполнено условие (4). Очевидно, что из (4) вытекает и справедливость условия (11). Значит, $\hat{C}$ также является c-ТА-кодом. ■

Лемма 3. Пусть стоит задача построения (N, c)-ССШШ. Рассмотрим кривые рода g над $\mathbb{F}_q$. Пусть D — дивизор на некоторой кривой, $\deg(D) = \alpha$. Тогда АГ-код C с дивизором D возможно использовать в ССШШ, если $\alpha \geqslant \max\{\lceil \log_q N \rceil + g - 1, 2g-1\}$.

Доказательство. Для использования в (N, c)-ССШШ кода C должны выполняться условия $\alpha \geqslant \log_q N + g - 1$ (теорема 4) и $\alpha \geqslant 2g - 1$ (теоремы 2 и 3). Значит, $\alpha \geqslant \max\{\lceil \log_q N + g - 1 \rceil, 2g-1\}$; заметим, что $\lceil \log_q N + g - 1 \rceil = \lceil \log_q N \rceil + g - 1$. ■

Напомним, что $B(N, c)$ – нижний порог затрат памяти для ССШШ (определение 2).

Теорема 5. Рассмотрим следующие четыре неравенства:

Неравенство 1 имеет вид

— если $c^2 > \lfloor 2\sqrt{q} \rfloor$, то

$$0 \leqslant g \leqslant \min\left\{\lceil \log_q N \rceil, \frac{\lfloor B(N,c)/q + 1 \rfloor - q - 1}{\lfloor 2\sqrt{q} \rfloor}, \frac{q - c^2(\lceil \log_q N \rceil - 1)}{c^2 - \lfloor 2\sqrt{q} \rfloor}\right\};$$

— если $c^2 < \lfloor 2\sqrt{q} \rfloor$ и $q < c^2(\lceil \log_q N \rceil - 1)$, то

$$\max\left\{0, \frac{c^2(\lceil \log_q N \rceil - 1) - q}{\lfloor 2\sqrt{q} \rfloor - c^2}\right\} \leqslant g \leqslant \min\left\{\lceil \log_q N \rceil, \frac{\lfloor B(N,c)/q + 1 \rfloor - q - 1}{\lfloor 2\sqrt{q} \rfloor}\right\};$$

— иначе

$$0 \leqslant g \leqslant \min\left\{\lceil \log_q N \rceil, \frac{\lfloor B(N,c)/q + 1 \rfloor - q - 1}{\lfloor 2\sqrt{q} \rfloor}\right\}.$$

Неравенство 2 имеет вид

$$\lceil \log_q N \rceil \leqslant g \leqslant \min\left\{ \frac{\lfloor B(N,c)/q+1 \rfloor - q - 1}{\lfloor 2\sqrt{q} \rfloor}, \frac{q+c^2}{2c^2 - \lfloor 2\sqrt{q} \rfloor} \right\}.$$

Неравенство 3 имеет вид

$$\max\left\{0, \frac{\lfloor B(N,c)/q+1 \rfloor - q - 1}{\lfloor 2\sqrt{q} \rfloor}\right\} \leqslant g \leqslant \min\left\{ \lceil \log_q N \rceil, \frac{\lfloor B(N,c)/q+1 \rfloor - q - 1}{\lfloor 2\sqrt{q} \rfloor}\right\}.$$

Неравенство 4 имеет вид

$$\max\left\{ \lceil \log_q N \rceil, \frac{\lfloor B(N,c)/q+1 \rfloor - q - 1}{\lfloor 2\sqrt{q} \rfloor}\right\} \leqslant g \leqslant \frac{\lfloor B(N,c)/q+1 \rfloor - c^2 - 1}{2c^2}.$$

Если род кривой g удовлетворяет этим неравенствам, то на кривой рода g возможно построить c-ТА-код, причём

1) длина этого кода минимальна, при которой ещё выполняется достаточное условие c-ТА (4);
2) этот код возможно использовать для построения (N,c)-ССШШ, более эффективных относительно затрачиваемой памяти, чем КСШШ.

Доказательство. Для того чтобы на кривой рода g над полем $\mathbb{F}_q$ можно было бы построить c-ТА-код длины n с дивизором степени α, достаточно выполнения условия (4). Минимальная величина n, при которой возможно его выполнение, равна $c^2\alpha + 1$, причём, в соответствии с леммой 3, минимальное значение α равно $\max\{\lceil \log_q N \rceil + g - 1, 2g-1\}$. Таким образом, минимально возможное n, при которой выполнено (4), равно $c^2 \max\{\lceil \log_q N \rceil + g - 1, 2g-1\} + 1$.

Кроме того, на кривой должно быть достаточное количество точек для построения кода. В частности, учитывая возможность дальнейшего использования проверки чётности и утверждение 4, их должно быть не менее $q + g\lfloor 2\sqrt{q} \rfloor + 1$. Так получаем условие

$$n = c^2 \max\{\lceil \log_q N \rceil + g - 1, 2g-1\} + 1 \leqslant q + g\lfloor 2\sqrt{q} \rfloor + 1.$$

Для возможности использования такого кода в ССШШ, более эффективной, чем КССШ и тривиальная схема, по утверждению 1 должно выполняться условие

$$n = c^2 \max\{\lceil \log_q N \rceil + g - 1, 2g-1\} + 1 \leqslant q + g\lfloor 2\sqrt{q} \rfloor + 1 < B(N,c)/q.$$

Два последних неравенства можно объединить в одно:

$$c^2 \max\{\lceil \log_q N \rceil + g - 1, 2g-1\} + 1 \leqslant \min\{q + g\lfloor 2\sqrt{q} \rfloor + 1, \lfloor B(N,c)/q+1 \rfloor\}. \qquad (12)$$

Решим это неравенство относительно g. Составим четыре системы:

$$\begin{cases} \lceil \log_q N \rceil \geqslant g, \\ q + g\lfloor 2\sqrt{q} \rfloor + 1 \leqslant \lfloor B(N,c)/q+1 \rfloor, \\ c^2(\lceil \log_q N \rceil + g - 1) \leqslant q + g\lfloor 2\sqrt{q} \rfloor, \end{cases} \quad \begin{cases} \lceil \log_q N \rceil \leqslant g, \\ q + g\lfloor 2\sqrt{q} \rfloor + 1 \leqslant \lfloor B(N,c)/q+1 \rfloor, \\ c^2(2g-1) \leqslant q + g\lfloor 2\sqrt{q} \rfloor, \end{cases}$$

$$\begin{cases} \lceil \log_q N \rceil \geqslant g, \\ q + g\lfloor 2\sqrt{q} \rfloor + 1 \geqslant \lfloor B(N,c)/q+1 \rfloor, \\ c^2(\lceil \log_q N \rceil + g - 1) + 1 \leqslant \lfloor B(N,c)/q+1 \rfloor, \end{cases} \quad \begin{cases} \lceil \log_q N \rceil \leqslant g, \\ q + g\lfloor 2\sqrt{q} \rfloor + 1 \geqslant \lfloor B(N,c)/q+1 \rfloor, \\ c^2(2g-1) + 1 \leqslant \lfloor B(N,c)/q+1 \rfloor. \end{cases}$$

Первая система соответствует случаю, когда в (12) слева максимумом является $\lceil \log_q N \rceil + g - 1$, а справа минимумом $q + g\lfloor 2\sqrt{q} \rfloor + 1$. Три остальные системы соответствуют трём оставшимся случаям. Решения первой системы совпадают с решениями неравенства 1, второй — с решениями неравенства 2 и т. д. ■

2.2. Алгоритм построения АГ-кода L-конструкции, применимого в ССШШ

Предложим концептуальный алгоритм построения АГ-кода L-конструкции, который может быть применён для построения (N, c)-ССШШ. Он основан на утверждениях п. 2.1 и на способе построения АГ-кода п. 1.3. Полученный по алгоритму АГ-код является одноточечным АГ-кодом (см. замечание 5).

Алгоритм 1. Построение одноточечного АГ-кода, применимого в ССШШ

Вход: $c \in \mathbb{N} \setminus \{1\}$ — максимальная мощность коалиции злоумышленников;
$N \in \mathbb{N} \setminus \{1\}$ — число легальных пользователей в ССШШ;
q — мощность конечного поля, над которым строится код;
L — список кривых над полем $\mathbb{F}_q$;
InputDecoders $= \{\text{ClassicDecoder}, \text{SoftDecoder}, \dots\}$ — список известных списочных декодеров для АГ-кодов.

Выход: порождающая матрица q-ичного АГ-кода L-конструкции, который вместе с алгоритмом СДСГ может быть применён для построения (N, c)-ССШШ; список наиболее приоритетных для использования списочных декодеров OutputDecoders.

1: Составить множество $G = \{g_1, g_2, \dots, g_{|G|}\}$ допустимых значений рода кривой, решив четыре неравенства относительно g из теоремы 5; $i := 1$, $g := g_i$.
2: **Пока** $i \leqslant |G|$:
3: **Если** уравнение $(x-1)(x-2) = 2g$ не имеет натуральных корней, **то**
4: $i := i + 1$, $g := g_i$;
5: перейти на следующую итерацию цикла на шаг 3.
6: $\alpha := \max\{\lceil \log_q N \rceil + g - 1, 2g - 1\}$;
7: $\hat{n} := c^2\alpha + 1$.
8: **Пока** $\hat{n} \leqslant \min\{q + g\lfloor 2\sqrt{q} \rfloor, \lfloor B(N, c)/q + 1 \rfloor\}$:
9: **Если** в L есть гладкая плоская проективная кривая X, $|X| \geqslant \hat{n}$, рода g над полем $\mathbb{F}_q$, **то**
10: выбрать кривую X,
11: **иначе**
12: выйти из цикла, перейти на шаг 38.
13: **Если** $|X| \geqslant \hat{n} + 1$, **то**
14: ParityCheck := `false`;
15: $n := \hat{n}$;
16: **иначе**
17: **если** $|X| < \hat{n} + 1 \wedge \exists m \in \mathbb{N}\ (q = 2^m) \wedge \exists k \in \mathbb{N}\ (\hat{n} - \alpha = 2k + 1)$, **то**
ParityCheck := `true`;
18: $n := \hat{n} + 1$,
19: **иначе**
20: ParityCheck := `false`;
21: $\hat{n} := \hat{n} + 1$, перейти на новую итерацию цикла (шаг 8).
22: OutputDecoders := $\varnothing$.
23: **Если** SoftDecoder $\in$ InputDecoders, **то**
24: **Если** для n выполняется условие (9), **то**
25: к OutputDecoders добавить метку мягкого СДСГ SoftDecoder.
26: **Если** ClassicDecoder $\in$ InputDecoders, **то**
27: **Если** $\sqrt{n\alpha} \notin \mathbb{N}$, **то**
28: **Если** для n выполняется условие (6), **то**

29: к OutputDecoders добавить метку классического СДСГ ClassicDecoder,
30: **иначе** // $\sqrt{n\alpha} \in \mathbb{N}$
31: **Если** для n выполняется условие (7), **то**
32: к OutputDecoders добавить метку классического СДСГ ClassicDecoder,
33: // ... проверка условий на иные алгоритмы списочного декодирования.
34: **Если** OutputDecoders $= \varnothing$, **то**
35: $\hat{n} := \hat{n} + 1$, перейти на новую итерацию цикла на шаге 8,
36: **иначе**
37: выйти из цикла на шаг 41.
38: **Если** $\hat{n}q > B(N, c)$, **то**
39: **Вернуть** «Ошибка. Эффективнее использовать КСШШ».
40: $g := g + 1$.
41: Объявить точку $Q \in X$ вида $(X_1 : X_2 : 0)$ (см. замечание 5).
42: **Если** ParityCheck = `true`, **то**
43: объявить множество $Points := \{P_1, \ldots, P_{n-1}\} \subset X \setminus \{Q\}$,
44: **иначе**
45: объявить множество $Points := \{P_1, \ldots, P_n\} \subset X \setminus \{Q\}$.
46: Объявить дивизор $D := \alpha Q$ (определение 10).
47: Присвоить $k := \alpha - g + 1$.
48: Рассмотреть пространство Римана — Роха $L(D)$ (определение 15). Построить его базис из k линейно независимых элементов. Обозначить его $B = \{\phi_1, \ldots, \phi_l\}$.
49: Построить матрицу

$$G = \begin{bmatrix} \phi_1(P_1) & \phi_1(P_2) & \ldots & \phi_1(P_{|Points|}) \\ \phi_2(P_1) & \phi_2(P_2) & \ldots & \phi_2(P_{|Points|}) \\ \vdots & \vdots & \vdots & \vdots \\ \phi_l(P_1) & \phi_l(P_2) & \ldots & \phi_l(P_{|Points|}) \end{bmatrix}.$$

50: **Если** ParityCheck = `true`, **то**
51: добавить к G столбец проверки на чётность, то есть в i-ю строку добавить элемент $\sum\limits_{j=1}^{|Points|} \phi_i(P_j)$.
52: **Вернуть** G и DecoderList.

Замечание 8. В силу требований, накладываемых утверждением 1, в алгоритме минимизируется значение длины кода n. Минимизация n при заданном q делает проектируемую ССШШ более эффективной.

Теорема 6. Алгоритм 1 является корректным, то есть с его помощью можно либо построить q-ичный АГ-код L-конструкции, который вместе с предложенными списочными декодерами OutputDecoders может быть применён для построения (N, c)-ССШШ, либо убедиться в неэффективности ССШШ по отношению к КСШШ.

Доказательство. На шаге 1 включаем в множество G возможные значения рода g, для которых выполняется свойство c-ТА (теорема 5).

На шаге 2 запускаем цикл по G, в результате которого подбираются кривая X подходящих мощности и рода, а также степень дивизора, такие, что эти кривые и дивизор могут быть использованы для построения искомого АГ-кода.

Коды строятся только на плоских кривых, поэтому сначала в цикле проверяем (шаги 3–5), возможно ли рассмотреть гладкую плоскую проективную кривую текущего

рода g. Согласно формуле (1), если уравнение $(x-1)(x-2) = 2g$ не имеет натуральных корней, то таких кривых не существует. Тогда увеличиваем g и осуществляем проверку, пока не будет выбран подходящий род g.

Затем, согласно лемме 3, выбираем число α как $\max(\lceil \log_q N \rceil + g - 1, 2g - 1)$ (шаг 6). Заметим, что α — минимальное из удовлетворяющих лемме 3 чисел, оно гарантирует дальнейшую минимизацию длины кода (замечание 8).

На шаге 7 выбираем константу $\hat{n}$, которая будет использована в качестве значения длины кода по умолчанию. Инициализируем её значением $c^2\alpha + 1$, так как это наименьшее (замечание 8) значение, при котором искомый АГ-код является c-ТА-кодом (теорема 1, условие (4)).

На шаге 8 запускается цикл по $\hat{n}$, пока $\hat{n}$ не достигнет минимального из значений $q+g\lfloor 2\sqrt{q} \rfloor$, $\lfloor B(N,c)/q+1 \rfloor$. Первое значение является максимальной мощностью кривой рода g над полем $\mathbb{F}_q$, уменьшенной на единицу (утверждение 4). Это значение также совпадает с максимально возможной длиной АГ-кода на кривой рода g над полем $\mathbb{F}_q$ (пока без учёта возможного добавления проверки чётности). Параметр $\hat{n}$ не может быть больше $\lfloor B(N,c)/q + 1 \rfloor$ по утверждению 1. Значит, цикл перебирает возможные значения $\hat{n}$ как потенциальной длины АГ-кода.

В цикле на шагах 9–10 выбираем гладкую плоскую кривую X рода g, такую, что $|X| \geqslant \hat{n}$. Это ограничение достаточно для того, чтобы утверждать, что при первом входе в цикл построенный на кривой X одноточечный АГ-код (замечание 5) может быть c-ТА-кодом (с учётом добавления проверки чётности, лемма 11), а при последующих итерациях — чтобы обеспечивать также выполнение условий на некоторые списочные декодеры (например, на классический СДСГ на шагах 28 и 31). Если такие кривые выбрать невозможно, то есть не существует X, такого, что $|X| \geqslant \hat{n}$, то выходим из цикла по $\hat{n}$, переходим на шаг 40 и увеличиваем род g.

Далее определяем, необходимо и возможно ли использование проверки чётности [18, п. 1.9(I)]. Если на кривой уже более $\hat{n}$ точек, то использовать проверку чётности не нужно, количества точек на кривой хватит для построения одноточечного АГ-кода (замечание 5), поэтому мы не удлиняем код, присваиваем ParityCheck значение `false`, а константе n — значение $\hat{n}$ (шаги 13–15). Если на кривой не больше $\hat{n}$ точек, код строится над двоичным полем и конструктивное расстояние строящегося кода нечётно, то возможно использование проверки чётности (лемма 11). В этом случае мы удлиняем код, присваиваем ParityCheck значение `true`, а n — значение $\hat{n} + 1$. Таким образом, мы сначала построим АГ-код с длиной $\hat{n}$, а затем удлиним его до длины n.

В противном случае, когда на кривой не больше $\hat{n}$ точек, но невозможно использование проверки чётности, переходим на новую итерацию цикла на шаге 8, так как количества точек на кривой не хватит для построения искомого кода. Далее в качестве длины кода используется константа n.

После этого начинаем проверку выполнения условий использования методов списочного декодирования. Инициализируем список OutputDecoders пустым списком. Если во входном списке декодеров есть метка мягкого СДСГ, то проверяем выполнение условия его использования (9) на шаге 24. Если условие выполнено, то помещаем его метку в выходной список декодеров.

Аналогично, если во входном списке декодеров есть метка классического СДСГ, то проверяем выполнение условия его использования (6) или (7) на шагах 27–32. Для этого сначала анализируем выполнение условия $\sqrt{n\alpha} \notin \mathbb{N}$. Если $\sqrt{n\alpha} \notin \mathbb{N}$, то необходимо подобрать параметры к условиям теоремы 2, п. 1. Если на шаге 28 выполняется (6), то возможно построение АГ-кода, который вместе с классическим СДСГ может быть

использован для построения (N, c)-ССШШ. В этом случае помещаем метку классического СДСГ в OutputDecoders. Если $\sqrt{n\alpha} \in \mathbb{N}$, то необходимо подобрать параметры к условиям теоремы 2, п. 2. Если на шаге 31 выполняется (7), то также помещаем метку классического СДСГ в OutputDecoders.

На шаге 33 можно проверить выполнение условий для иных методов списочного декодирования.

На шаге 34 если список OutputDecoders не пуст, то существует списочный декодер, который применим для построения (N, c)-ССШШ совместно с одноточечным АГ-кодом на кривой X. Тогда выходим из циклов на шаг 41. Если список пуст, то увеличиваем $\hat{n}$ и продолжаем поиски.

Если поиски окажутся неудачными, то n станет так велико, что мы покинем вложенный цикл (проверка на шаге 8), затем пройдём проверку на эффективность проектируемой ССШШ (шаг 38) и, если она будет пройдена, увеличим род кривой g (шаг 40). Далее опять начнём искать кривые, но уже большего рода, а значит, с бóльшим количеством точек, что видно из утверждения 4 (при $g \to \infty$ количество точек на кривой стремится к ∞). В конце концов, увеличивая род и мощность кривых, мы можем найти кривую, для которой выполнятся все необходимые условия. Если этого не произойдёт, то мы либо достигнем для длины кода значения $\lfloor B(N, c)/q + 1 \rfloor$, либо исчерпаем все возможные значения рода g из G. Это будет означать, что рассматриваемые достаточные условия не гарантируют существования искомого АГ-кода.

Отметим, что в цикле мы найдём минимальное подходящее значение n (замечание 8), так как n на каждой итерации изменяется максимум на единицу и увеличивается ровно настолько, насколько это необходимо для выполнения накладываемых условий.

После выбора параметров выбираем точку Q (шаг 41). В замечаниии 5 указаны рекомендации к выбору этой точки. Затем выбираем множество точек кривой X, отличных от Q. Обозначим его как $Points$ (шаги 42–45). Мощность этого множества равна $n - 1$, если ParityCheck=`true`, и n в противном случае. Такое множество всегда существует, так как в случае ParityCheck=`true` кривая X выбрана так, чтобы $|X| \geqslant n$ (шаги 10, 16), в противном случае $|X| \geqslant n + 1$ (шаги 10, 13).

Затем на шаге 46 объявляем дивизор $D = \alpha Q$, который будет использован в качестве дивизора искомого АГ-кода.

Так как по построению $\alpha > 2g - 2$, размерность пространства Римана — Роха $L(D)$, ассоциированного с дивизором D, равна $\alpha - g + 1$ (утверждение 7). В алгоритме размерность $L(D)$ обозначена как k (шаг 47). Зная размерность, вычисляем базис пространства $L(D)$ — набор из k линейно независимых функций из $\mathbb{F}_q(X)$. Зная базис, легко написать порождающую матрицу АГ-кода: i-я строка матрицы есть результат действия отображения Ev_{Points} (2) на i-й элемент базиса (шаг 49). Длина такого кода равна $|Points|$. Поэтому если ParityCheck=`true`, то на шаге 51 в матрицу G добавляем столбец проверки чётности. Таким образом, длина кода станет равна выбранному параметру n.

На последнем шаге возвращаем матрицу G и выходной список декодеров. ■

Теорема 7. Алгоритм 1 полиномиально сводится к алгоритму вычисления базиса пространства Римана — Роха.

Доказательство. Решение неравенств 1–4 из теоремы 5 выполняется за константное число шагов. Цикл по g, начинающийся на шаге 2, работает полиномиальное от входных параметров c, N, q время, так как в нём перебираются значения мно-

жества G, мощность которого ограничена значениями многочленов от c, N, q (теорема 5). После шагов 3–7 начинается цикл по $\hat{n}$ на шаге 8, который также имеет полиномиальное от c, N, q число шагов — не более $\min\{q + g\lfloor 2\sqrt{q}\rfloor, \lfloor B(N,c)/q + 1\rfloor\}$, где $g \leqslant \max\{x : x \in G\}$. Внутри этого цикла производится поиск кривых в списке L, длина которого не зависит от входных параметров, проверка необходимости и возможности использования проверки чётности, а также проверка условий для декодеров из списка InputDecoders, длина которого является константой. После выхода из циклов или ошибки на шаге 38 производятся только тривиальные присвоения и объявления переменных, за исключением шага 41, где необходимо построить базис пространства Римана — Роха (замечание 4). ■

Для построения АГ-кода L-конструкции необходимо работать с проективной кривой. Однако при выборе кривой в алгоритме возможно рассмотрение сначала аффинных кривых, а затем использование их проективизаций.

Замечание 9. Выделим некоторые частные случаи АГ-кодов, которые могут быть получены по алгоритму. Известно, что все АГ-коды L-конструкции на кривых рода ноль являются или кодами Рида — Соломона, или их удлинениями [9, с. 314]. Количество точек на кривой рода 0 равно $q+1$ (утверждение 4). Это влечёт ограничение на длину АГ-кода, построенного по алгоритму 1: $n \leqslant q$. В этом случае достаточное условие того, что одноточечный АГ-код является c-ТА-кодом, выглядит следующим образом:

$$c < \sqrt{q/\alpha}.$$

Учитывая утверждение 4, получим, что достаточное условие того, что одноточечный АГ-код на кривой рода 1 с максимальным числом точек является c-ТА-кодом, имеет вид

$$c < \sqrt{(q + \lfloor 2\sqrt{q}\rfloor)/\alpha}.$$

Известно, что все АГ-коды L-конструкции на кривых рода 1 являются эллиптическими кодами, то есть кодами, построенными по эллиптическим кривым [9, п. 4.4.1, разд. «Коды рода один»].

Данная работа преимущественно посвящена построению АГ-кодов безотносительно методов их списочного декодирования. Несмотря на это, для более полного исследования получены также достаточные условия применимости списочных декодеров [7, 8] (теоремы 2, 3). Достаточность выполнения этих условий отражена в алгоритме 1 на шагах 23–32.

При применении классического СДСГ обычно используется следующая модель канала: отправитель посылает $x \in \mathbb{F}_q$, получатель получает некоторый $y \in \mathbb{F}_q$, и вероятность того, что $x = y$, равна $1-p$, где p — вероятность ошибки в канале. Классический СДСГ выбран для исследования, потому что он, во-первых, достигает границы Джонсона, которая является нижним порогом списочной «декодируемости» кода, то есть классический СДСГ удовлетворяет минимальным требованиям к списочному декодеру; во-вторых, применение классического СДСГ достаточно для построения ССШШ, более эффективных с точки зрения затрат памяти (утверждение 1), чем тривиальные схемы широковещательного шифрования или чем КСШШ (см. п. 2.3).

Мягкий СДСГ является расширением классического СДСГ. При его применении используется следующая модель канала: отправитель посылает $x \in \mathbb{F}_q$, получатель на своём конце имеет функции плотности вероятности $f_x(\cdot)$, где $f_x(a)$ — вероятность того,

что получатель получит на выходе значение a в случае, когда отправитель послал x. Построение таких функций в случае декодирования АГ-кодов описано в [8].

На шаге 33 алгоритма 1 могут быть размещены условия на иные методы списочного декодирования для АГ-кодов. При использовании этих методов они должны быть включены в список InputDecoders на входе алгоритма. Например, возможно использование списочного декодера, представленного в [16]. В данную работу рассмотрение этого алгоритма, имеющего радиус декодирования $n-\lceil\sqrt{2n(n-d)}+g-1\rceil-1$, не включено, так как он меньше аналогичного радиуса для классического СДСГ [7, введение в разд. 4].

Сравнение различных методов списочного декодирования относительно использования в ССШШ, в том числе сравнение характеристик классического и мягкого СДСГ, представляется темой отдельного исследования.

Отметим, что одноточечные АГ-коды на фиксированной кривой имеют максимально возможную длину. Это условие является важным с точки зрения выполнения достаточного условия c-ТА (4).

2.3. Пример работы алгоритма

Пусть на вход алгоритма поданы следующие параметры:

$$c = 2,\ N = 512,\ q = 8.$$

В качестве списка кривых L будем использовать таблицы кривых из [17]. В список декодеров занесём метки классического и мягкого СДСГ, рассматриваемых в работе.

Применим алгоритм 1. Сначала покажем, что достаточные условия c-ТА и условия применимости СДСГ не гарантируют существование q-ичных кодов Рида — Соломона, применимых для построения $(512, 2)$-ССШШ. Затем построим одноточечный АГ-код на кривой рода 1, который вместе с мягким СДСГ может быть применён для организации $(512, 2)$-ССШШ, и построим одноточечный АГ-код на кривой рода 3, который может быть применён вместе с классическим СДСГ.

На шаге 1 сначала вычисляем величину $B(N, c) = N = 512$. Затем определяем множество возможных значений рода кривой g. После решения неравенств получаем

$$G = \{1, 2, 3\} \cup \{3\} \cup \varnothing \cup \varnothing = \{1, 2, 3\}.$$

Так как значение $g = 0$ не входит в множество G, можно сделать вывод, что при поданных на вход параметрах достаточные условия c-ТА не гарантируют существования искомого АГ-кода на кривой рода 0. В силу замечания 9, это означает, что достаточные условия не гарантируют существование кодов Рида — Соломона, которые вместе с СДСГ могут быть применены для организации $(512, 2)$-ССШШ.

Полагаем $g = 1$ и заходим в цикл (шаг 2). Уравнение $(x-1)(x-2) = 2$ имеет натуральный корень 3, поэтому можем рассматривать плоские кривые рода 1. На шагах 6 и 7 присваиваем $\alpha := \max\{\lceil\log_8 512+1-1\rceil, 2\cdot 1-1\} = 3$ и $\hat{n} := 2^2\cdot 3+1 = 13$. Затем заходим в цикл по $\hat{n}$ на шаге 8, так как выполняются условия $\hat{n} = 13 \leqslant q+g\lfloor 2\sqrt{q}\rfloor+1 = 14$ и $\hat{n} = 13 \leqslant \lfloor B(N, c)/q + 1\rfloor = 65$. В цикле на шаге 10 в качестве кривой X выберем кривую, заданную следующим многочленом:

$$X_2^2X_3 + X_1X_2X_3 + X_2X_3^2 - X_1^3 - X_3^3 = 0. \tag{13}$$

Можно проверить, что на этой кривой находится 14 точек над полем $\mathbb{F}_8$, таким образом, она удовлетворяет всем условиям, наложенным на шаге 9. Так как $|X| \geqslant \hat{n} = 13$,

присваиваем ParityCheck:= `false` и $n := \hat{n} = 13$. На шаге 22 инициализируем список возможных для использования декодеров. Затем, так как на шаге 23 видим, что SoftDecoder $\in$ InputDecoders, переходим на шаг 24, где проверяем условие (9) и убеждаемся в том, что оно выполнено: $c = 2 \geqslant (13 - 7 - 1)/3 \geqslant 5/3$. На шаге 26 видим, что ClassicDecoder $\in$ InputDecoders; анализируя на следующем шаге значение величины $\sqrt{n\alpha} = \sqrt{13 \cdot 3}$, убеждаемся, что оно не принадлежит множеству $\mathbb{N}$, и на шаге 27 проходим по первой ветке. На шаге 28 видим, что условие (6) не выполнено. На шаге 37, так как на шаге 34 обнаружено, что список возможных декодеров не пуст, выходим из цикла на шаг 41. Таким образом, все необходимые параметры для построения кода подобраны.

Рассмотрим кривую (13). Поле $\mathbb{F}_8$ будем рассматривать как $\mathbb{F}_2[\xi]/(\xi^3 + \xi + 1)$. Выпишем все точки кривой:

$$Q = (0:1:0), P_1 = (1:0:1), P_2 = (\xi^4 : \xi^4 : 1), P_3 = (\xi^2 : \xi^2 : 1),$$
$$P_4 = (\xi : \xi : 1), P_5 = (\xi^3 : \xi^4 : 1), P_6 = (\xi^6 : \xi : 1), P_7 = (\xi^5 : \xi : 1), P_8 = (\xi^4 : 1 : 1),$$
$$P_9 = (\xi^5 : \xi^2 : 1), P_{10} = (\xi^6 : \xi^4 : 1), P_{11} = (\xi^2 : 1 : 1), P_{12} = (\xi^3 : \xi^2 : 1), P_{13} = (\xi : 1 : 1).$$

В качестве точки Q выберем точку $(0 : 1 : 0)$ (шаг 41), а в качестве $Points$ — множество $\{P_i : 1 \leqslant i \leqslant 13\}$ (шаг 45). Объявим дивизор $D = 3Q$ (шаг 46) и $l = 3 - 1 + 1 = 3$ (шаг 47).

На шаге 48 необходимо вычислить базис пространства Римана — Роха. Вычислим сначала дивизоры пересечения $X_{\text{pr}} \cdot X_1$, $X_{\text{pr}} \cdot X_2$, $X_{\text{pr}} \cdot X_3$, используя соображения, изложенные в замечании 4.

Многочлен X_1 равен нулю на кривой X_{pr} только в точке Q. Тогда, в силу теоремы 6, $I(P_1; X_{\text{pr}}; X_1) = 3$. Значит, $X_{\text{pr}} \cdot X_1 = 3Q$.

Многочлен X_2 равен нулю на кривой X_{pr} только в точке P_1. Тогда, в силу теоремы 6, $I(P_2; X_{\text{pr}}; X_1) = 3$. Значит, $X_{\text{pr}} \cdot X_2 = 3P_1$.

Многочлен X_3 равен нулю на кривой X_{pr} только в точке Q. Тогда, в силу теоремы 6, $I(Q; X_{\text{pr}}; X_3) = 3$. Значит, $X_{\text{pr}} \cdot X_3 = 3Q$.

Рассмотрим функции 1, X_1/X_3, X_2/X_3. Их количество совпадает с $l = 3$. В силу замечаний 1 и 2 имеем

$$(X_1/X_3) = 3Q - 3Q = 0, \ (X_2/X_3) = 3P_1 - 3Q.$$

Кроме того,

$$(1) + D = 3Q \geqslant 0, \quad (X_1/X_3) + D = 0 + 3Q = 3Q \geqslant 0,$$
$$(X_2/X_3) + D = 3P_1 - 3Q + 3Q = 3P_1 \geqslant 0,$$

то есть все эти функции лежат в пространстве $L(D)$. Можно проверить, что они линейно независимы в пространстве Римана — Роха $L(D)$. Значит, $\{1, X_1/X_3, X_2/X_3\}$ — полный линейно независимый набор функций из $L(D)$, то есть искомый базис. Построим порождающую матрицу G:

$$\begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & \xi^4 & \xi^2 & \xi & \xi^3 & \xi^6 & \xi^5 & \xi^4 & \xi^5 & \xi^6 & \xi^2 & \xi^3 & \xi \\ 0 & \xi^4 & \xi^2 & \xi & \xi^4 & \xi & \xi & 1 & \xi^2 & \xi^4 & 1 & \xi^2 & 1 \end{bmatrix}.$$

Так как ParityCheck = `false`, то на шагах 50–52 не выполняем никаких действий. Матрица G и список OutputDecoders = {SoftDecoder} с меткой мягкого СДСГ являются

результатом работы алгоритма. Получен $[13, 3, \geqslant 10]_8$-код, который совместно с мягким СДСГ может быть применён для организации $(512, 2)$-ССШШ.

Рассмотрим, какой код был бы получен в результате работы алгоритма, если бы на входе в список декодеров не был включен только жёсткий СДСГ. В этом случае мы бы не прошли проверку на шагах 23, 27 и 28, на шаге 34 получили пустой список OutputDecoders, увеличили $\hat{n}$ и перешли на шаг 8. При $g = 1$ на шаге 8 при $\hat{n} = 14$ не заходим на следующую итерацию, так как условие $\hat{n} = 14 \leqslant q + g\lfloor 2\sqrt{q} \rfloor = 13$ неверно. Поэтому переходим на проверку на шаге 38, проходим её и на шаге 40 увеличиваем значение рода: $g = 2$. При $g = 2$ уравнение на шаге 3 не имеет корней, поэтому увеличиваем g до 3.

Затем на шаге 6 присваиваем $\alpha := \max\{\lceil \log_8 512 + 3 - 1 \rceil, 2 \cdot 3 - 1\} = 5$, а на шаге 7 $\hat{n} := 2^2 \cdot 5 + 1 = 21$. Далее попадаем на начало цикла по $\hat{n}$ (шаг 8). Заходим в цикл, так как выполняются условия $\hat{n} = 21 \leqslant q + g\lfloor 2\sqrt{q} \rfloor = 23$ и $\hat{n} = 21 \leqslant \lfloor B(N, c)/q + 1 \rfloor = 65$. На шаге 10 в качестве кривой X выберем кривую, заданную следующим многочленом:

$$X_1^3 X_2 + X_2^3 X_3 + X_3^3 X_1 = 0. \tag{14}$$

На этой кривой находится 24 точки над полем $\mathbb{F}_8$, таким образом, она удовлетворяет всем условиям, наложенным на шаге 9.

Так как $|X| \geqslant 22$, присваиваем ParityCheck := `false` и $n := \hat{n} = 21$. Пропускаем проверку на шаге 23, так как SoftDecoder $\notin$ InputDecoders. На шаге 27 анализируем значение величины $\sqrt{n\alpha} = \sqrt{21 \cdot 5}$, убеждаемся, что она не принадлежит множеству $\mathbb{N}$. Значит, проходим по первой ветке на шаг 28 и видим, что условие (6) не выполнено. Тогда на шаге 35 увеличиваем значение $\hat{n}$ до 22 и переходим на новую итерацию цикла.

На шаге 10 можем выбрать ту же самую кривую. Снова присваиваем ParityCheck := := `false` и $n := \hat{n} = 22$. Пройдя все проверки, попадаем на шаг 27 и видим, что снова $\sqrt{n\alpha} = \sqrt{22 \cdot 5} \notin \mathbb{N}$. Значит, снова проходим по первой ветке на шаг 28. Там убеждаемся, что условие (6) выполнено, добавляем в OutputDecoders метку классического СДСГ ClassicDecoder и на шаге 37 выходим из циклов, попадая на шаг 41. Таким образом, все необходимые параметры для построения кода подобраны.

Рассмотрим кривую (14). Поле $\mathbb{F}_8$, как и ранее, будем рассматривать как $\mathbb{F}_2[\xi]/(\xi^3 + \xi + 1)$. Выпишем все точки кривой:

$$P_1 = (1 : 0 : 0),\ Q = (0 : 1 : 0),\ P_2 = (0 : 0 : 1),\ P_3 = (1 : \xi : 1),\ P_4 = (1 : \xi^2 : 1),$$
$$P_5 = (1 : \xi^4 : 1),\ P_6 = (\xi : 1 : 1),\ P_7 = (\xi : \xi^2 : 1),\ P_8 = (\xi : \xi^6 : 1),\ P_9 = (\xi^2 : 1 : 1),$$
$$P_{10} = (\xi^2 : \xi^4 : 1),\ P_{11} = (\xi^2 : \xi^5 : 1),\ P_{12} = (\xi^3 : \xi^2 : 1),\ P_{13} = (\xi^3 : \xi^3 : 1),\ P_{14} = (\xi^3 : \xi^5 : 1),$$
$$P_{15} = (\xi^4 : 1 : 1),\ P_{16} = (\xi^4 : \xi : 1),\ P_{17} = (\xi^4 : \xi^3 : 1),\ P_{18} = (\xi^5 : \xi : 1),\ P_{19} = (\xi^5 : \xi^5 : 1),$$
$$P_{20} = (\xi^5 : \xi^6 : 1),\ P_{21} = (\xi^6 : \xi^3 : 1),\ P_{22} = (\xi^6 : \xi^4 : 1),\ P_{23} = (\xi^6 : \xi^6 : 1).$$

В качестве Q выберем точку $(0 : 1 : 0)$, в качестве $Points$ — множество $\{P_i : 2 \leqslant i \leqslant 23\}$. Объявим дивизор $D = 5Q$ и $l = 5 - 3 + 1 = 3$.

Затем необходимо вычислить базис пространства Римана — Роха. Вычисляем дивизоры пересечения $X_{\text{pr}} \cdot X_1$, $X_{\text{pr}} \cdot X_2$, $X_{\text{pr}} \cdot X_3$.

Многочлен X_1 равен нулю на кривой X_{pr} только в точках P_2 и Q. Так как $X_1 = 0$ не является касательной к кривой в точке Q, то $I(Q; X_{\text{pr}}; X_1) = 1$. Тогда, в силу теоремы 6, $I(P_2; X_{\text{pr}}; X_1) = 3$. Значит, $X_{\text{pr}} \cdot X_1 = 3P_2 + Q$.

Многочлен X_2 равен нулю на кривой X_{pr} только в точках P_1 и P_2. Так как $X_2 = 0$ не является касательной к кривой в точке P_2, то $I(P_2; X_{\text{pr}}; X_2) = 1$. Тогда, в силу теоремы 6, $I(P_1; X_{\text{pr}}; X_1) = 3$. Значит, $X_{\text{pr}} \cdot X_2 = 3P_1 + P_2$.

Многочлен X_3 равен нулю на кривой X_{pr} только в точках P_1 и Q. Так как $X_3 = 0$ не является касательной к кривой в точке P_1, то $I(P_1; X_{\mathrm{pr}}; X_3) = 1$. Тогда, в силу теоремы 6, $I(Q; X_{\mathrm{pr}}; X_3) = 3$. Значит, $X_{\mathrm{pr}} \cdot X_3 = 3Q + P_1$.

Рассмотрим функции 1, X_2/X_3, X_1X_2/X_3^2. Их количество совпадает с $l = 3$. В силу замечаний 1 и 2 имеем

$$(X_1/X_3) = 3P_2 - P_1 - 2Q, \quad (X_2/X_3) = 2P_1 + P_2 - 3Q$$

и, кроме того,

$$(1) + D = 5Q \geqslant 0, \quad (X_2/X_3) + D = 2P_1 + P_2 - 3Q + 5Q = 2P_1 + P_2 + 2Q \geqslant 0,$$
$$\left(X_1X_2/X_3^2\right) + D = (X_1/X_2) + (X_2/X_3) + D =$$
$$= 3P_2 - P_1 - 2Q + 2P_1 + P_2 - 3Q + 5Q = P_1 + 4P_2 \geqslant 0,$$

то есть все эти функции лежат в пространстве $L(D)$. Можно проверить, что они линейно независимы в пространстве Римана — Роха $L(D)$. Значит, $\{1, X_2/X_3, X_1X_2/X_3^2\}$ — полный линейно независимый набор функций из $L(D)$, то есть искомый базис.

Построим порождающую матрицу G:

$$\begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & \xi & \xi^2 & \xi^4 & 1 & \xi^2 & \xi^6 & 1 & \xi^4 & \xi^5 & \xi^2 & \xi^3 & \xi^5 & 1 & \xi & \xi^3 & \xi & \xi^5 & \xi^6 & \xi^3 & \xi^4 & \xi^6 \\ 0 & \xi & \xi^2 & \xi^4 & \xi & \xi^3 & 1 & \xi^2 & \xi^6 & 1 & \xi^5 & \xi^6 & \xi & \xi^4 & \xi^5 & 1 & \xi^6 & \xi^3 & \xi^4 & \xi^2 & \xi^3 & \xi^5 \end{bmatrix}.$$

Так как ParityCheck = `false`, на шагах 50–52 не выполняем никаких действий. Матрица G является результатом работы алгоритма в случае, если в списке InputDecoders только классический СДСГ. Получен $[22, 3, \geqslant 17]_8$-код, который совместно с классическим СДСГ может быть применён для организации $(512, 2)$-ССШШ.

2.4. Сравнение АГ-кодов с другими классами кодов

Выгода в затратах памяти при использовании АГ-кодов на кривых высоких родов обсуловлена тем, что, в отличие от классических кодов Рида — Соломона и Рида — Маллера, длина АГ-кода ограничена не мощностью поля, а лишь мощностью кривой, на которой строится код и которая может быть существенно больше, чем мощность поля [9, теорема 3.1.25]. Таким образом, можно строить ССШШ над полями меньшей мощности, незначительно увеличивая n, в то время как при использовании классических кодов приходится увеличивать мощность поля. Наиболее наглядно это можно видеть в случаях, когда $q = 2^m$ для некоторого m. Отметим, что именно такие поля наиболее часто используются в технических и программных реализациях схем защиты.

Рассмотрим некоторые примеры ССШШ, на которых сравним классические коды Рида — Соломона и Рида — Маллера с АГ-кодами. Покажем, что существуют как параметры ССШШ, при которых более выгодно использование исключительно классических кодов, так и параметры, при которых более выгодно использование АГ-кодов на кривых рода больше 0.

Пример 1. Как показано в рассмотренном в п. 2.3 примере, достаточные условия не гарантируют построения кода Рида — Соломона над полем $\mathbb{F}_8$, применимого в ССШШ с параметрами $c = 2$, $N = 512$. Для обслуживания такой ССШШ построены $[13, 3, \geqslant 10]_8$- и $[22, 3, \geqslant 17]_8$-коды на кривых рода 1 и 3 соответственно, которые должны применяться с мягким СДСГ в первом случае и с классическим СДСГ во втором. Заметим, что для этих кодов величины nq, важные с точки зрения затрачиваемой в ССШШ памяти (см. таблицу на с. 70), равны 104 и 176 соответственно.

Попробуем построить классические коды, применимые для организации такой схемы. Так, для этих параметров возможно построение $[16, 4, 13]_{16}$-кода Рида — Соломона. Однако для этого кода $nq = 256$, что больше, чем при использовании АГ-кодов на кривых рода 1 и 3, следовательно, хуже с точки зрения объёма используемой памяти.

При заданных параметрах можно построить 2-ТА-код Рида — Маллера с $q = 8$, $r = 1$, $m = 2$. Для этого кода $n = 8^2 = 64$, $k = 3$, $d = 8^2 - 1 \cdot 8 = 58$, поэтому он имеет самое большое значение $nq = 512$.

Таким образом, для рассмотренных в примере значений c и N использование АГ-кодов более выгодно, чем классических кодов Рида — Соломона и Рида — Маллера.

Пример 2. В [8, п. «Example»] рассмотрен пример ССШШ с параметрами $c = 3$, $N = 2^{20} = 1\,048\,576$. Для неё в [8] построен $[31, 4, 28]_{32}$-код Рида — Соломона, который применим в таких ССШШ совместно как с классическим, так и с мягким СДСГ. Заметим, что алгоритм 1 также выдаст для таких значений c, N, q этот код. Действительно, для этих параметров выбор такого кода представляется наиболее удачным, так как построение АГ-кода над полем $\mathbb{F}_{32}$ на кривых более высоких родов с целью оптимизации памяти не имеет смысла: увеличение рода кривых приведёт только к увеличению n, а уменьшение величины q до 16, например, не удастся, так как для кодов, построенных при параметрах $c = 3$, $N = 2^{20}$, $q = 16$, перестанет выполняться достаточное условие c-ТА (4).

Для обслуживания такой ССШШ также возможно построение 3-ТА $(1, 2)$-кода Рида — Маллера над полем $\mathbb{F}_{16}$, но такой код показывает большую неэффективность в использовании памяти: учитывая, что $n = q^m$, для него $nq = 256 \cdot 16 = 4096$.

Значит, для рассмотренных в примере параметров использование АГ-кодов представляется менее выгодным, чем использование классического кода Рида — Соломона.

Пример 3. Пусть поставлена задача построения ССШШ с параметрами $N = 2^{20} + 1$, $c = 3$. В этом случае код Рида — Соломона над полем $\mathbb{F}_{32}$ построить уже не удастся. Действительно, для обслуживания такого числа пользователей с помощью кода Рида — Соломона над полем $\mathbb{F}_{32}$ величина $q^k = 32^k$ должна быть больше, чем 2^{20} (утверждение 2). Это накладывает ограничение на размерность кода: $k > 4$. Но уже при $k = 5$ получаем, что $d = n - 4$. Это ограничение совместно с достаточным условием c-ТА (4) налагает на длину кода Рида — Соломона условие $n > 36$. Но код Рида — Соломона не может иметь такой длины над полем $\mathbb{F}_{32}$. Таким образом, для такой ССШШ возможно построение лишь $[64, 4, 61]_{16}$-кода Рида — Соломона, для которого $nq = 4096$. Ситуация аналогична для всех N, таких, что $2^{20} < N < 2^{24}$.

Легко проверить, что с помощью алгоритма 1 для таких параметров АГ-код над полем $\mathbb{F}_{32}$ можно построить только при условии существования над $\mathbb{F}_{32}$ кривой рода 3 с максимальным числом точек, в частности с 65 точками. Тогда возможно построение $[64, 5, \geqslant 57]_{32}$-АГ-кода, который может быть применён в такой ССШШ совместно с мягким СДСГ. Однако в ходе исследования такой кривой найти не удалось. Известные кривые рода 3 над полем $\mathbb{F}_{32}$ имеют максимум 64 точки [17], что недостаточно для построения искомого АГ-кода даже с использованием столбца проверки чётности.

Значит, для заданных параметров использование АГ-кодов менее выгодно, чем классических кодов.

Пример 4. Пусть поставлена задача построения ССШШ с параметрами

$$2^{35} < N < 2^{36},\ c = 2.$$

Для обслуживания такого числа пользователей с помощью классического кода Рида — Соломона необходимо использовать поле $\mathbb{F}_{32}$. Действительно, для поля $\mathbb{F}_{16}$ получаем,

что величина $q^k = 16^k$ должна быть больше, чем 2^{35} (утверждение 2). Это накладывает ограничение на размерность кода: $k > 8$. Уже при $k = 9$ получаем, что $d = n - 8$. Это ограничение совместно с достаточным условием c-ТА (4) налагает на длину кода Рида — Соломона условие $n > 32$. Но код Рида — Соломона не может иметь такой длины над полем $\mathbb{F}_{16}$. Таким образом, для такой ССШШ возможно построение лишь $[29, 8, 22]_{32}$- кода Рида — Соломона, для которого $nq = 928$. Он может быть применён в этой ССШШ совместно с мягким СДСГ (при параметре допустимого отклонения не более 0,1).

С помощью алгоритма 1 для таких же параметров возможно построить АГ-код над полем $\mathbb{F}_{16}$ на кривой рода 6 с 65 точками [17]. Получим $[57, 9, \geqslant 43]_{16}$-АГ-код, который может быть применён в такой ССШШ совместно с мягким СДСГ (при параметре допустимого отклонения не более 0,1). Для такого кода $nq = 912$.

Для применения классического СДСГ можно использовать $[30, 8, 23]_{32}$-код Рида — Соломона с $nq = 960$ или $[58, 9, \geqslant 44]_{16}$-АГ-код с $nq = 928$.

Возможно также построение 2-ТА $(2, 3)$-кода Рида — Маллера над полем $\mathbb{F}_q$, но, учитывая, что для него длина равна $n = q^m = 16^3$, можно заключить, что его использование является наименее эффективным относительно затрат памяти.

Таким образом, для заданных параметров использование АГ-кодов более выгодно, чем классических.

Заключение

В [3] представлены достаточные условия эффективного использования линейных кодов и методов списочного декодирования в ССШШ. В работе эти результаты применены для АГ-кодов L-конструкции [6, 9, 12] и классического и мягкого алгоритмов СДСГ [7, 8]. Таким образом, получены точные достаточные условия применения этих кодов и декодеров в ССШШ, исследована связь между ними.

На основе полученных утверждений представлен концептуальный алгоритм построения АГ-кода L-конструкции, который, совместно с алгоритмами СДСГ, может быть эффективно использован в ССШШ.

ЛИТЕРАТУРА

1. *Stinson D. R. and Wei R.* Combinatorial properties and constructions of traceability schemes and frameproof codes // SIAM J. Discr. Math. 1998. V. 11. No. 1. P. 41–53.
2. *Staddon J. N., Stinson D. R., and Wei R.* Combinatorial properties of frameproof and traceability codes // IEEE Trans. Inform. Theory. 2001. V. 47. No. 3. P. 1042–1049.
3. *Silverberg A., Staddon J., and Walker J.* Applications of list decoding to tracing traitors // IEEE Trans. Inform. Theory. 2003. V. 49. No. 5. P. 1312–1318.
4. *Деундяк В. М., Мкртичян В. В.* Исследование границ применения схемы защиты информации, основанной на РС-кодах // Дискретный анализ и исследование операций. 2011. Т. 18. № 3. С. 21–38.
5. *Деундяк В. М., Евпак С. А., Мкртичян В. В.* Исследование свойств q-ичных помехоустойчивых кодов Рида — Маллера как кодов для защиты от копирования // Проблемы передачи информации. 2015. Т. 51. № 4. С. 99–111.
6. *Гоппа В. Д.* Алгебраико-геометрические коды // Известия АН СССР. Сер. матем. 1982. Т. 46. № 4. С. 762–781.
7. *Guruswami V. and Sudan M.* Improved decoding of Reed — Solomon and algebraic-geometric codes // Foundations of Computer Science. Palo Alto: IEEE, 1998. P. 28–37.

8. *Fernandez M. and Soriano M.* Identification of traitors in algebraic-geometric traceability codes // IEEE Trans. Signal Proc. 2004. V. 52. Iss. 10. P. 3073–3077.
9. *Влэдуц С. Г., Ногин Д. Ю., Цфасман М. А.* Алгеброгеометрические коды. Основные понятия. М.: МЦНМО, 2003. 504 с.
10. *Fiat A. and Naor M.* Broadcast encryption // LNCS. 1994. V. 773. P. 480–491.
11. *Chor B., Fiat A., and Naor M.* Tracing traitors // LNCS. 1994. V. 839. P. 257–270.
12. *Hoholdt T., van Lindt J., and Pellikaan R.* Algebraic geometry codes // Handbook of Coding Theory / eds. V. S. Pless, W. C. Huffman, and R. A. Brualdi. V. 1. Amsterdam: Elsevier, 1998. P. 871–961.
13. *Лэнг С.* Алгебра. М.: Мир, 1968. 564 с.
14. *Hess F.* Computing Riemann — Roch spaces in algebraic function fields and related topics // J. Symbolic Comput. 2002. V. 33. No. 4. P. 425–445.
15. `http://magma.maths.usyd.edu.au/magma/` — Magma Computational Algebra System.
16. *Shokrollahi A. and Wasserman H.* List decoding of algebraic-geometric codes // IEEE Trans. Inform. Theory. 1999. V. 45. No. 2. P. 432–437.
17. *Van Der Geer G. and Van Der Vlugt M.* Tables of curves with many points // Mathematics of Computation. 2000. V. 69. No. 230. P. 797–810.
18. *Мак-Вильямс Ф. Дж., Слоэн Н. Дж. А.* Теория кодов, исправляющих ошибки. М.: Связь, 1979. 744 с.

REFERENCES

1. *Stinson D. R. and Wei R.* Combinatorial properties and constructions of traceability schemes and frameproof codes. SIAM J. Discr. Math., 1998, vol. 11, no. 1, pp. 41–53.
2. *Staddon J. N., Stinson D. R., and Wei R.* Combinatorial properties of frameproof and traceability codes. IEEE Trans. Inform. Theory, 2001, vol. 47, no. 3, pp. 1042–1049.
3. *Silverberg A., Staddon J., and Walker J.* Applications of list decoding to tracing traitors. IEEE Trans. Inform. Theory, 2003, vol. 49, no. 5, pp. 1312–1318.
4. *Deundyak V. M. and Mkrtichyan V. V.* Issledovaniye granits primeneniya skhemy zashchity informatsii, osnovannoy na RS-kodakh [Research of applying bounds of the information protection scheme based on RS-codes]. Diskretn. Anal. Issled. Oper., 2011, vol. 18, no. 3, pp. 21–38. (in Russian)
5. *Deundyak V. M., Evpak S. A., and Mkrtichyan V. V.* Analysis of properties of q-ary Reed — Muller error-correcting codes viewed as codes for copyright protection. Probl. Inform. Transm., 2015, vol. 51, no. 4, pp. 398–408.
6. *Goppa V. D.* Algebraiko-geometricheskiye kody [Algebraic-geometric codes]. Izv. Akad. Nauk SSSR. Ser. Mat., 1982, vol. 46, no. 4, pp. 762–781. (in Russian)
7. *Guruswami V. and Sudan M.* Improved decoding of Reed — Solomon and algebraic-geometric codes. Foundations of Computer Science, Palo Alto, IEEE, 1998, pp. 28–37.
8. *Fernandez M. and Soriano M.* Identification of traitors in algebraic-geometric traceability codes. IEEE Trans. Signal Proc., 2004, vol. 52, iss. 10, pp. 3073–3077.
9. *Vladut S. G., Nogin D. Y., and Tsfasman M. A.* Algebrogeometricheskie kody. Osnovnye ponyatiya. [Algebraic Geometric Codes. Basic Notions]. Moscow, MCCME Publ., 2003. 504 p. (in Russian)
10. *Fiat A. and Naor M.* Broadcast encryption. LNCS, 1994, vol. 773, pp. 480–491.
11. *Chor B., Fiat A., and Naor M.* Tracing traitors. LNCS, 1994, vol. 839, pp. 257–270.
12. *Hoholdt T., van Lindt J., and Pellikaan R.* Algebraic geometry codes. Handbook of Coding Theory, eds. V. S. Pless, W. C. Huffman, and R. A. Brualdi, vol. 1. Amsterdam, Elsevier, 1998, pp. 871–961.

13. *Lang S.* Algebra. Springer, 2002.
14. *Hess F.* Computing Riemann — Roch spaces in algebraic function fields and related topics. J. Symbolic Computation, 2002, vol. 33, no. 4, pp. 425–445.
15. `http://magma.maths.usyd.edu.au/magma/` — Magma Computational Algebra System.
16. *Shokrollahi A. and Wasserman H.* List decoding of algebraic-geometric codes. IEEE Trans. Inform. Theory, 1999, vol. 45, no. 2, pp. 432–437.
17. *Van Der Geer G. and Van Der Vlugt M.* Tables of curves with many points. Mathematics of Computation, 2000, vol. 69, no. 230, pp. 797–810.
18. *MacWilliams F. J. and Sloane N. J. A.* The Theory of Error-Correcting Codes. Elsevier, 1977. 744 p.

УДК 519.725

О СПИСОЧНОМ ДЕКОДИРОВАНИИ ВЕЙВЛЕТ-КОДОВ НАД КОНЕЧНЫМИ ПОЛЯМИ ХАРАКТЕРИСТИКИ ДВА

Д. В. Литичевский

Челябинский государственный университет, г. Челябинск, Россия

Доказывается, что вейвлет-код над полем $\mathrm{GF}(2^m)$ с длиной кодовых и информационных слов $n = 2^m - 1$ и $(n-1)/2$ соответственно, у которого среди коэффициентов спектрального представления порождающего многочлена имеется $d+1$ последовательных нулей, $0 < d < (n-3)/2$, допускает списочное декодирование за полиномиальное время. Шаги алгоритма, осуществляющего списочное декодирование с исправлением до $e < n - \sqrt{n(n-d-2)}$ ошибок, реализованы в виде программы. Приведены примеры её применения для списочного декодирования зашумленных кодовых слов. Отмечено, что неравенство Варшамова — Гилберта при достаточно больших n не позволяет судить о существовании вейвлет-кодов с максимальным кодовым расстоянием $(n-1)/2$.

Ключевые слова: *вейвлет-коды, полифазное кодирование, декодирование списком.*

DOI 10.17223/20710410/44/7

ON LIST DECODING OF WAVELET CODES OVER FINITE FIELDS OF CHARACTERISTIC TWO

D. V. Litichevskiy

Chelyabinsk State University, Chelyabinsk, Russia

E-mail: litichevskiydv@gmail.com

In this paper, we consider wavelet code defined over the field $\mathrm{GF}(2^m)$ with the code length $n = 2^m - 1$ and information words length $(n-1)/2$ and prove that a wavelet code allows list decoding in polynomial time if there are $d+1$ consecutive zeros among the coefficients of the spectral representation of its generating polynomial and $0 < d < (n-3)/2$. The steps of the algorithm that performs list decoding with correction up to $e < n - \sqrt{n(n-d-2)}$ errors are implemented as a program. Examples of its use for list decoding of noisy code words are given. It is also noted that the Varshamov — Hilbert inequality for sufficiently large n does not allow to judge about the existence of wavelet codes with a maximum code distance $(n-1)/2$.

Keywords: *wavelet codes, polyphase coding, list decoding.*

Введение

Вейвлет-коды, согласно [1], являются подклассом квазициклических кодов с циклическим сдвигом кодовых слов на две позиции. Первоначально их порождающие матрицы строились с помощью ортогональных фильтров масштабирующей функции и вейвлет-функции. Описание этих подходов доступно в [2, 3]. Однако практическое

применение описанных методик было затруднено необходимостью построения масштабирующих функций с заданными свойствами [4, 5]. На основании результатов [6] о факторизации параунитарных матриц в работах [7, 8], а также в работах других авторов трудность с построением требуемых порождающих многочленов вейвлет-кодов была преодолена.

В дальнейшем класс вейвлет-кодов был расширен путём использования биортогональных наборов фильтров [9, 10]. Это упростило построение порождающих многочленов и позволило находить вейвлет-коды с требуемыми свойствами.

В [11] предложена схема помехоустойчивого кодирования, основанная на использовании биортогональных наборов фильтров точного восстановления. Использование лифтинговой схемы из [12] расширило возможности построения вейвлет-кодов с заданными характеристиками, в частности позволило построить биортогональные вейвлет-коды с максимально возможным и заданным кодовым расстоянием над конечными полями нечётной характеристики.

Однако предложенная в [11] схема кодирования не позволяет строить коды с максимально возможным кодовым расстоянием над полем характеристики два. Поэтому в работе [13] предложена иная схема помехоустойчивого кодирования, названная полифазной, и доказано, что с помощью этой схемы возможно построение над конечным полем $\mathrm{GF}(2^m)$ биортогональных вейвлет-кодов с длиной кодовых и информационных слов n и $(n-1)/2$ соответственно с максимально возможным и заданным кодовым расстоянием, где m — натуральное число, $n = 2^m - 1$.

В полифазной схеме кодирования для вычисления кодового многочлена $c(x)$ используется пара комплементарных фильтров (h, g), где $h(x) = \sum\limits_{j=0}^{n-1} h_j x^j$ и $g(x) = \sum\limits_{j=0}^{n-1} g_j x^j$. Пара фильтров (h, g) называется комплементарной, если определитель их полифазной матрицы $P(x)$

$$P(x) = \begin{bmatrix} h_e(x) & g_e(x) \\ h_o(x) & g_o(x) \end{bmatrix}$$

равен 1, где $h_e(x)$, $h_o(x)$ и $g_e(x)$, $g_o(x)$ — полифазные компоненты фильтров $h(x)$ и $g(x)$ соответственно.

Определение 1 [13]. Полифазные компоненты кодового многочлена $c(x)$ в кольце $\mathrm{GF}_{2^m}[x]/(x^n - 1)$ определяются как

$$\begin{bmatrix} c_e(x^2) \\ c_o(x^2) \end{bmatrix} = \begin{bmatrix} h_e(x^2) & g_e(x^2) \\ h_o(x^2) & g_o(x^2) \end{bmatrix} \begin{bmatrix} 1 \\ x^2 \end{bmatrix} v(x^2) = \begin{bmatrix} h_e(x^2) + x^2 g_e(x^2) \\ h_o(x^2) + x^2 g_o(x^2) \end{bmatrix} v(x^2),$$

где $v(x) = \sum\limits_{j=0}^{(n-3)/2} v_j x^j$ — информационный многочлен.

Тогда кодовый многочлен $c(x)$ имеет вид

$$c(x) = c_e(x) + x c_o(x) = f(x) v(x^2) \bmod (x^n - 1), \tag{1}$$

все операции умножения многочленов выполняются в кольце $\mathrm{GF}_{2^m}[x]/(x^n - 1)$.

Определение 2. Многочлен $f(x) = h(x) + x^2 g(x) \bmod (x^n - 1)$ называется порождающим многочленом вейвлет-кода.

Процедура построения порождающих многочленов $f(x)$ для вейвлет-кодов с максимально возможным и заданным кодовым расстоянием при помощи процедуры лифтинга (см. [12]) описана в работе [13].

Существование вейвлет-кодов с максимально возможным и заданным кодовым расстоянием над конечными полями характеристики два согласуется с результатом Э. Берлекэмпа [14], согласно которому над конечным полем $\mathrm{GF}(q)$ существуют линейные коды с произвольно большой длиной кодового слова n, скоростью R и кодовым расстоянием d, которые удовлетворяют соотношению

$$\frac{d}{n} \geqslant \delta(R),$$

где $\delta(R)$ — наименьшее решение уравнения

$$R = 1 - H_q(\delta(R));$$

$H_q(p) = p\log_q(q-1) - p\log_q p - (1-p)\log_q(1-p)$ — q-значная энтропия Шеннона. При этом неравенство Варшамова — Гильберта, которое, согласно [15], записывается в виде

$$V_q(2e, n) \leqslant q^{n(1-R)},$$

где $V_q(2e, n)$ — объём (число элементов) шара радиуса $2e$ в пространстве слов длины n, состоящих из символов алфавита мощности q, напротив, не позволяет судить о существовании найденных в [13] вейвлет-кодов, поскольку оно является только достаточным условием существования кода.

В [13] также доказано, что найденные вейвлет-коды с максимально возможным кодовым расстоянием являются кодами Рида — Соломона, построенными во временной области, а коды с заданным кодовым расстоянием являются подпространствами кодов Рида — Соломона во временной области, поэтому к ним применим алгоритм помехоустойчивого декодирования Берлекэмпа — Уэлча, описанный в [16].

Одним из важнейших свойств кода является существование для него алгоритма, осуществляющего декодирование списком за полиномиальное время от параметров кода. В классической задаче декодирования требуется, чтобы исправление ошибки в принятом кодовом слове осуществлялось однозначно. В задаче декодирования списком допускается на выходе декодера иметь до L вариантов декодирования при некотором фиксированном L. Код допускает декодирование списком длины L с исправлением e ошибок, если множество кодовых слов обладает следующим свойством: любой шар радиуса e в кодовом пространстве содержит не более L кодовых слов. Тогда утверждение о том, что алгоритм позволяет осуществлять списочное декодирование кода с исправлением e ошибок, означает, что любой шар радиуса e в кодовом пространстве содержит не больше некоторого заранее фиксированного для кода количества кодовых слов, формирующих возвращаемый алгоритмом список. В [17] показано существование кодов, допускающих декодирование списком длины $n+1$ с исправлением до $e < \sqrt{n(n-d)}$ ошибок, вблизи границы Хэмминга

$$q^k V_q(e, n) \leqslant q^n, \tag{2}$$

где q — мощность алфавита, над которым построен код; n и k — длины кодовых и информационных слов соответственно; $V_q(e, n)$ — объём шара радиуса e в пространстве слов длины n, состоящих из символов алфавита мощности q. К сожалению, сложность

описания таких кодов, а также экспоненциальная зависимость процедур их кодирования и декодирования от длины кодового слова n значительно ограничивают область их применения.

Однако для кода Рида — Соломона $\mathrm{RS}[n, k, d = n - k + 1]$ разработаны алгоритмы списочного декодирования, сложность которых полиномиально зависит от длины кодового слова. В кодах Рида — Соломона $\mathrm{RS}[n, k]$ со спектральной схемой кодирования информационному слову $v_0, v_1, \ldots, v_{k-1}$ ставится в соответствие кодовое слово $y_0, y_1, \ldots, y_{n-1}$, где $y_i = v(x_i) = \sum\limits_{\ell=0}^{k-1} v_\ell x_i^\ell$. Здесь $x_0, x_1, \ldots, x_{n-1}$, $x_i \neq x_j$ при $i \neq j$, — произвольные зафиксированные для конкретного кода элементы поля $\mathrm{GF}(q)$, над которым определён код.

Первоначальная версия алгоритма списочного декодирования для кода Рида — Соломона, описанная в [18], позволяла исправлять до $e < n - \sqrt{2n(k-1)}$ ошибок. Это означает, что в любом шаре радиуса $\leqslant e$ содержится не более L кодовых слов, при этом L полиномиально зависит от параметров кода. На вход алгоритма подаётся принятое искажённое кодовое слово $\widetilde{y}_0, \widetilde{y}_1, \ldots, \widetilde{y}_{n-1}$, на выходе получается список кодовых слов, содержащихся в шаре радиуса e с центром в принятом искажённом кодовом слове. Построение списка кодовых слов состоит из двух этапов: интерполяционного и факторизационного. На интерполяционном этапе алгоритма выполняется поиск отличного от нуля многочлена от двух переменных $R(x, y) \in \mathrm{GF}(q)[x, y]$, такого, что $R(x, y)$ обращается в нуль во всех точках $(x_j, \widetilde{y}_j)$, $j = 0, \ldots, n-1$, и его $(1, k-1)$-взвешенная степень не превосходит фиксированного значения, являющегося функцией от параметров кода n и k. (w_x, w_y)-Взвешенная степень монома $r_{j_1 j_2} x^{j_1} y^{j_2}$ равна $j_1 w_x + j_2 w_y$; (w_x, w_y)-взвешенная степень многочлена $R(x, y) = \sum\limits_{j_1, j_2} r_{j_1 j_2} x^{j_1} y^{j_2}$ равна максимальной среди всех (w_x, w_y)-взвешенных степеней его мономов с ненулевыми коэффициентами $r_{j_1 j_2}$. На факторизационном шаге осуществляется поиск всех многочленов $v(x) \in \mathrm{GF}(q)[x]$ степени не выше $k-1$, таких, что $y - v(x)$ делит $R(x, y)$ и $v(x_i) = \widetilde{y}_i$ не менее чем в $n - e$ точках. Найденные многочлены формируют искомый список кодовых слов.

Позднее в [19] представлена улучшенная версия алгоритма, позволяющая исправлять уже до $e < n - \sqrt{n(k-1)} = n - \sqrt{n(n-d)}$ ошибок, что является улучшением упомянутого результата из [17], поскольку при таком же радиусе шаров количество попадающих в них кодовых слов получается меньшим и их список восстанавливается за полиномиальное время. При этом общая структура алгоритма не претерпела изменений, он по-прежнему состоит из интерполяционного и факторизационного шагов. Однако на многочлен $R(x, y)$ при построении на интерполяционном шаге накладываются дополнительные требования, а именно: должны существовать натуральные r и l, связанные соотношениями

$$l > \sqrt{nr(k-1)(r+1)}, \quad l \leqslant r(n-e),$$

такие, что каждая из точек (x_i, y_i), $i = 0, \ldots, n-1$, должна быть нулём кратности r многочлена $R(x, y)$, а его $(1, k-1)$-взвешенная степень не должна превосходить l. Многочлен $R(x, y)$ имеет в точке (a, b) нуль кратности r, если многочлен $R(u+a, v+b)$ не имеет мономов, $(1, 1)$-взвешенная степень которых меньше r.

Возможность списочного декодирования вейвлет-кодов с заданным кодовым расстоянием над полем нечётной характеристики, описанных в [11], была изучена в [20]. Данная работа является её продолжением, в ней рассматривается возможность спи-

сочного декодирования вейвлет-кодов с заданным кодовым расстоянием над полем характеристики два, описанных в [13].

В работе доказывается, что если для порождающего многочлена $f(x)$ некоторого вейвлет-кода, определённого над полем $\mathrm{GF}(2^m)$ ($m \in \mathbb{N}$) с примитивным элементом α, с длиной кодовых и информационных слов $n = 2^m - 1$ и $(n-1)/2$ соответственно имеют место равенства

$$f(\alpha^j) = 0 \ \text{ при } \ j = j^*, \ldots, j^* + d\,,$$

где $0 \leqslant j^* \leqslant n - 1 - d$ и $0 < d < (n-3)/2$, то вейвлет-код с кодовым расстоянием $d+2$ допускает списочное декодирование.

Помимо этого, в работе приводится одна из возможных реализаций алгоритма списочного декодирования для допускающих его вейвлет-кодов, определённых над полем характеристики два, работающая за полиномиальное время, а также даются примеры её использования.

Устанавливается факт, что неравенство Варшамова — Гильберта не позволяет судить о существовании вейвлет-кодов с заданным кодовым расстоянием над полем характеристики два.

1. Допустимость списочного декодирования вейвлет-кода

Выберем в поле $\mathrm{GF}(2^m)$ примитивный элемент α. Рассмотрим вейвлет-код с длиной кодовых слов $n = 2^m - 1$, длиной информационных слов $(n-1)/2$, порождающим многочленом $f(x)$ и процедурой кодирования (1). Символом $W[n, (n-1)/2, d]$ будем обозначать кодовое пространство $(n, (n-1)/2)$-вейвлет-кода с кодовым расстоянием d.

Лемма 1. Если для порождающего многочлена $f(x)$ вейвлет-кода с длиной кодовых и информационных слов n и $(n-1)/2$ соответственно выполняются соотношения

$$f(\alpha^j) = 0 \ \text{ при } \ j = j^*, \ldots, j^* + d, \quad 0 < d < (n-3)/2,$$

то кодовое расстояние вейвлет-кода не меньше $d+2$.

Доказательство. Выберем d так, чтобы выполнялось $0 < d < (n-3)/2$. Будем считать, что для многочлена $f(x)$ степени не больше $n-1$ имеют место равенства

$$f(\alpha^j) = 0 \ \text{ при } \ j = j^*, \ldots, j^* + d,$$

где $0 \leqslant j^* \leqslant n - 1 - d$. Согласно [21], многочлен $f(x)$ является порождающим многочленом вейвлет-кода, то есть найдётся комлементарная пара фильтров $h(x)$ и $g(x)$, такая, что $f(x) = h(x) + ax^2 g(x) \bmod (x^n - 1)$, где $a \in \mathrm{GF}(2^m)$, $a \neq 0$. Преобразование Фурье кодового многочлена $c(x)$, равное

$$c(\alpha^i) = v(\alpha^{2i}) f(\alpha^i), \ i = 0, \ldots, n-1,$$

запишется в виде

$$\big(C_0, \ldots, C_{j^*-1}, \underbrace{0, \ldots, 0}_{d+1}, C_{j^*+d+1}, \ldots, C_{n-1}\big).$$

Спектральный многочлен $C(y) = C_0 + C_1 y + \ldots + C_{n-1} y^{n-1}$, $y \in \mathrm{GF}(2^m)$, представим в виде

$$\begin{gathered} C(y) = C_0 + \cdots + C_{j^*-1} y^{j^*-1} + C_{j^*+d+1} y^{j^*+d+1} + \cdots + C_{n-1} y^{n-1} = \\ = C_{j^*+d+1} y^{j^*+d+1} + \cdots + C_{n-1} y^{n-1} + C_0 y^n + \cdots + C_{j^*-1} y^{n+j^*-1} = \\ = y^{j^*+d+1} (C_{j^*+d+1} + \cdots + C_{n-1} y^{n-j^*-d-2} + C_0 y^{n-j^*-d-1} + \cdots + C_{j^*-1} y^{n-d-2}). \end{gathered}$$

Так как коэффициент $c_i = C(\alpha^{-i})$ равен нулю тогда и только тогда, когда α^{-i} является корнем многочлена $C(y)$, а число его отличных от нуля корней не превосходит $n-d-2$, вес кодового слова построенного вейвлет-кода не может быть меньше $n-(n-d-2)$, следовательно, кодовое расстояние будет не меньше $d+2$. ■

Теорема 1 (о допустимости списочного декодирования вейвлет-кода).

Существует алгоритм, позволяющий осуществлять списочное декодирование вейвлет-кода $W[n,(n-1)/2,d+2]$, $0 < d < (n-3)/2$, со схемой кодирования (1), для порождающего многочлена которого выполняются соотношения

$$f(\alpha^j) = 0 \ \text{ при } \ j = j^*, \ldots, j^* + d.$$

Доказательство. Опираясь на полученное в лемме 1 представление спектрального многочлена $C(y)$, c_i запишем в виде

$$c_i = \alpha^{-i(j^*+d+1)} \left(\sum_{j=0}^{n-j^*-d-2} C_{j+j^*+d+1}\alpha^{-ij} + \sum_{j=n-j^*-d-1}^{n-d-2} C_{j+j^*+d-n+1}\alpha^{-ij} \right),$$

где $i = 0, \ldots, n-1$, иначе

$$c_i\alpha^{i(j^*+d+1)} = \sum_{j=0}^{n-j^*-d-2} C_{j+j^*+d+1}\alpha^{-ij} + \sum_{j=n-j^*-d-1}^{n-d-2} C_{j+j^*+d-n+1}\alpha^{-ij}. \tag{3}$$

Полученное в (3) представление задаёт код Рида — Соломона $RS(n, n-d-1)$, в котором кодовое слово $s = \{s_i\}_{i=0}^{n-1}$ получается из некоторого информационного слова $\beta = \{\beta_j\}_{j=0}^{n-d-2}$ по формулам

$$s_i = \sum_{j=0}^{n-d-2} \beta_j\alpha^{-ij}, \quad i = 0, \ldots, n-1, \tag{4}$$

поскольку α^{-1} также является примитивным элементом поля $\mathrm{GF}(2^m)$.

Список возможных спектральных многочленов $C(y)$ может быть получен при помощи любого алгоритма списочного декодирования кода Рида — Соломона $RS(n, n-d-1)$ с процедурой кодирования (4). Однако в этот список могут попасть последовательности, не порождённые кодовыми словами вейвлет-кода.

Согласно процедуре кодирования (1) вейвлет-кода $W[n,(n-1)/2,d+2]$, значения C_j, $j = 0, \ldots, n-1$, могут быть найдены как

$$v(\alpha^{2j})f(\alpha^j) = C_j, \ j = 0, \ldots, n-1.$$

Данные равенства задают систему линейных уравнений относительно коэффициентов информационного многочлена $v(x)$, решая которую, можно либо найти коэффициенты $v(x)$, либо показать, что не существует кодового многочлена $c(x)$, соответствующего значениям C_j, $j = 0, \ldots, n-1$. По свойствам порождающего многочлена $f(x)$, $(d+1)$ уравнений системы являются вырожденными. С учётом результатов о декодировании кода Рида — Соломона полученную систему линейных уравнений можно записать в виде

$$\begin{cases} v(\alpha^{2j})f(\alpha^j) = \beta_{j+n-j^*-d-1}, \ j = 0, \ldots, j^*-1, \\ v(\alpha^{2j})f(\alpha^j) = \beta_{j-j^*-d-1}, \ j = j^*+d+1, \ldots, n-1. \end{cases} \tag{5}$$

Система уравнений, заданная (5), содержит $(n-1)/2$ неизвестных и $n-d-1 > (n-1)/2$ уравнений. Это означает, что рассматриваемый вейвлет-код $W[n,(n-1)/2, d+2]$ является подпространством кода Рида — Соломона $\mathrm{RS}[n, n-d-1]$. Поэтому для него длина списка и число исправляемых ошибок не превосходят таковых для кода $\mathrm{RS}[n, n-d-1]$. ■

Замечание 1. На основании теоремы 1 алгоритм списочного декодирования вейвлет-кода $W[n,(n-1)/2,d+2]$ с порождающим многочленом $f(x)$ состоит из следующих шагов:

1) вместо полученного зашумлённого кодового слова $\widetilde{c} = \{\widetilde{c}_i\}_{i=0}^{n-1}$ вейвлет-кода $W[n,(n-1)/2,d+2]$ рассматривается зашумлённое слово $\widetilde{s} = \{\widetilde{s}_i\}_{i=0}^{n-1}$, $s_i = \widetilde{c}_i\alpha^{i(j^*+d+1)}$, кода Рида — Соломона $\mathrm{RS}[n, n-d-1]$;
2) к зашумлённому слову $\widetilde{s}$ применяется алгоритм списочного декодирования кода Рида — Соломона $\mathrm{RS}[n, n-d-1]$ с процедурой кодирования (4), в результате получаем список информационных слов β;
3) для каждого найденного информационного слова β решаем систему уравнений (5). Из найденных векторов v формируем список информационных слов вейвлет-кода $W[n,(n-1)/2,d+2]$.

Замечание 2. Существует алгоритм, позволяющий осуществлять списочное декодирование вейвлет-кода $W[n,(n-1)/2,d+2]$, $0 < d < (n-3)/2$, для порождающего многочлена которого выполняются соотношения

$$f(\alpha^j) = 0 \ \text{ при } \ j = j^*, \ldots, j^*+d.$$

Замечание 3. При использовании в качестве алгоритма списочного декодирования кода Рида — Соломона улучшенной версии алгоритма Гурусвами — Судана, описанной в [19], алгоритм списочного декодирования вейвлет-кода $W[n,(n-1)/2,d+2]$ будет исправлять до $n-\sqrt{n(n-d-2)}$ ошибок и работать за полиномиальное время от параметров кода.

Замечание 4. Предельный случай $d = (n-3)/2$ в теореме не рассматривается, так как, согласно результатам работы [13], он соответствует вейвлет-коду $W[n,(n-1)/2,(n+3)/2]$ с максимально возможным кодовым расстоянием, то есть является кодом Рида — Соломона, построенным во временной области. Поэтому списочное декодирование в этом случае может быть осуществлено при помощи алгоритмов списочного декодирования кодов Рида — Соломона $\mathrm{RS}[n,(n-1)/2]$.

Замечание 5. О применимости неравенства Варшамова — Гилберта в вопросе существования вейвлет-кода с максимальным кодовым расстоянием.

Рассмотрим вейвлет-код $W[n,(n-1)/2,d+2]$, $0 < d < (n-3)/2$, определённый над полем $\mathrm{GF}(2^m)$, $n = 2^m-1$. При $n = 2^m-1$ и $k = (n-1)/2$ неравенство Варшамова — Гилберта (см. [15]) имеет вид

$$\sum_{j=0}^{d} n^j C_{n-1}^j < (n+1)^{(n+1)/2}. \tag{6}$$

Учитывая поведение C_{n-1}^j при больших n и то, что $d < (n-3)/2$, получаем, что самым большим слагаемым в сумме является $n^d C_{n-1}^d$. Оценим снизу значение суммы

$\sum\limits_{j=0}^{d} n^j C_{n-1}^j$ при максимально допустимом $d = (n-5)/2$:

$$\sum_{j=0}^{d} n^j C_{n-1}^j > n^d C_{n-1}^d = n^{(n-5)/2} C_{n-1}^{(n-5)/2}.$$

Применив к полученному выражению формулу Стирлинга, находим

$$n^{(n-5)/2} C_{n-1}^{(n-5)/2} \sim n^{(n-5)/2} \frac{\sqrt{n-1}}{\sqrt{2\pi\left(\frac{n-5}{2}\right)\left(\frac{n+3}{2}\right)}} \frac{(n-1)^{n-1}}{\left(\frac{n-5}{2}\right)^{(n-5)/2}\left(\frac{n+3}{2}\right)^{(n+3)/2}} =$$

$$= \frac{n^{(n-5)/2}\sqrt{n-1}}{\sqrt{\frac{\pi}{2}(n-5)(n+3)}} \frac{2^{n-1}(n-1)^{n-1}}{(n-5)^{(n-5)/2}(n+3)^{(n+3)/2}}.$$

Правую часть представим в виде

$$\frac{n^{(n-5)/2}\sqrt{n-1}}{\sqrt{\frac{\pi}{2}(n-5)(n+3)}} \frac{2^{n-1}(n-1)^{n-1}}{(n-5)^{(n-5)/2}(n+3)^{(n+3)/2}} =$$

$$= \frac{2^{n-1}n^{(n-5)/2}\sqrt{n-1}}{\sqrt{\frac{\pi}{2}(n-5)(n+3)}} \left(\frac{n-1}{n-5}\right)^{(n-5)/2} \left(\frac{n-1}{n+3}\right)^{(n+3)/2} =$$

$$= \frac{2^{n-1}n^{(n-5)/2}\sqrt{n-1}}{\sqrt{\frac{\pi}{2}(n-5)(n+3)}} \left(1+\frac{4}{n-5}\right)^{((n-5)/4)\cdot 2} \left(1-\frac{4}{n+3}\right)^{(-(n+3)/4)(-2)}.$$

Поделив полученное выражение на правую часть неравенства (6), приходим к соотношению

$$\frac{2^{n-1}\sqrt{n-1}}{\sqrt{\frac{\pi}{2}(n-5)(n+3)}} \frac{n^{(n-5)/2}}{(n+1)^{(n+1)/2}} \left(1+\frac{4}{n-5}\right)^{((n-5)/4)\cdot 2} \left(1-\frac{4}{n+3}\right)^{(-(n+3)/4)(-2)} =$$

$$= \frac{2^{n-1}\sqrt{n-1}}{n^3\sqrt{\frac{\pi}{2}(n-5)(n+3)}} \left(\frac{n}{n+1}\right)^{(n+1)/2} \left(1+\frac{4}{n-5}\right)^{((n-5)/4)\cdot 2} \left(1-\frac{4}{n+3}\right)^{(-(n+3)/4)(-2)} =$$

$$= \frac{2^{n-1}\sqrt{n-1}}{n^3\sqrt{\frac{\pi}{2}(n-5)(n+3)}} \left(1-\frac{1}{n+1}\right)^{-(n+1)(-1/2)} \times$$

$$\times \left(1+\frac{4}{n-5}\right)^{((n-5)/4)\cdot 2} \left(1-\frac{4}{n+3}\right)^{(-(n+3)/4)(-2)}.$$

Полученное выражение при достаточно больших n больше 1 и стремится к бесконечности при $n \to +\infty$. Поэтому, начиная с некоторого n, код $W[n, (n-1)/2, d+2 = (n-1)/2]$ не удовлетворяет неравенству Варшамова — Гилберта.

Таким образом, неравенство Варшамова — Гилберта не позволяет судить о существовании найденных в [13] вейвлет-кодов.

2. Примеры

Чтобы проиллюстрировать работу алгоритма, описанного в замечании 1, приведём примеры его использования для списочного декодирования зашумлённых кодовых слов нескольких вейвлет-кодов.

Пример 1. Рассмотрим некоторый вейвлет-код, определённый над полем $\mathrm{GF}(8)$ с неприводимым многочленом $1+x+x^3$ и порождающим элементом α, с длиной кодовых и информационных слов 7 и 3 соответственно, порождающим многочленом

$$f(x) = 2x^2 + 5x^3 + 6x^4 + x^6$$

и процедурой кодирования (1). При этом комплементарные фильтры h и g, использованные при построении вейвлет-кода, равны соответственно

$$h(x) = 3 + 2x + 7x^2 + 6x^3 + 4x^4 + 2x^5,$$
$$g(x) = 5 + 3x + 2x^2 + 2x^3 + x^4 + 3x^5 + 2x^6.$$

Для того чтобы при помощи леммы 1 получить кодовое расстояние построенного вейвлет-кода, вычислим значения кодового многочлена $f(x)$ в точках $1, \alpha, \ldots, \alpha^6$. Получаем, что $f(\alpha^j) = 0$ при $j = 0, \ldots, 2$, следовательно, параметры j^* и d равны 0 и 2 соответственно, поэтому, согласно лемме 1, рассматриваемый вейвлет-код имеет кодовое расстояние 4 и может быть обозначен как $W[7, 3, 4]$.

Для иллюстрации работы алгоритма рассмотрим кодовое слово

$$c = (0, 0, 0, 0, 0, 0, 0)$$

и соответствующее ему зашумлённое кодовое слово

$$\widetilde{c} = (1, 7, 0, 0, 0, 0, 0).$$

Описанный в п. 1 алгоритм позволяет найти все информационные слова вейвлет-кода $W[7, 3, 4]$, соответствующие кодовые слова которых попадают в шар радиуса 2 с центром в зашумлённом кодовом слове $\widetilde{c}$. На первом шаге алгоритм преобразует принятое зашумлённое кодовое слово $\widetilde{c}$ вейвлет-кода $W[7, 3, 4]$ в зашумлённое кодовое слово кода Рида — Соломона $\mathrm{RS}[7, 4]$

$$\widetilde{s} = (1, 2, 0, 0, 0, 0, 0).$$

На втором шаге алгоритм применяет к зашумлённому кодовому слову $\widetilde{s}$ процедуру списочного декодирования кодов Рида — Соломона и получает список информационных слов β кода Рида — Соломона $\mathrm{RS}[7, 4]$

$$(0, 0, 0, 0),\ (1, 1, 0, 1),\ (5, 7, 0, 3).$$

На третьем шаге алгоритм решает систему из трёх линейных уравнений с тремя неизвестными

$$\begin{cases} v_0 + 5v_1 + 7v_2 = \beta_0, \\ 4v_0 + 3v_1 + 6v_2 = \beta_1, \\ 5v_0 + 6v_1 + 4v_2 = \beta_3 \end{cases}$$

для каждого информационного слова β из списка, полученного на втором шаге, и получает список информационных слов v вейвлет кода $W[7, 3, 4]$

$$(0, 0, 0),\ (4, 0, 2),\ (0, 0, 2).$$

Полученным информационным словам соответствуют кодовые слова

$$(0,0,0,0,0,0,0),\ (1,7,3,0,5,0,0),\ (1,7,0,2,0,0,4)$$

вейвлет-кода $W[7,3,4]$ с процедурой кодирования (1) и порождающим многочленом $f(x)$, каждое из которых попадает в шар радиуса 2 с центром в зашумлённом кодовом слове $\widetilde{c}$.

Пример 2. Рассмотрим некоторый вейвлет-код, определённый над полем $\mathrm{GF}(32)$ с неприводимым многочленом $1 + x^3 + x^5$ и порождающим элементом α, с длиной кодовых и информационных слов 31 и 15 соответственно, порождающим многочленом

$$\begin{aligned} f(x) = {} & 7 + 24x + 2x^2 + 5x^4 + 29x^5 + 2x^6 + 18x^7 + 3x^8 + x^9 + 15x^{10} + 22x^{11} + x^{12} + \\ & + 16x^{13} + 29x^{14} + 17x^{15} + 6x^{16} + 16x^{17} + 17x^{18} + 25x^{19} + 21x^{20} + 26x^{21} + \\ & + 10x^{22} + 30x^{23} + 18x^{24} + 6x^{25} + 24x^{26} + 4x^{27} + 31x^{28} + 14x^{29} + 15x^{30} \end{aligned}$$

и процедурой кодирования (1). При этом комплементарные фильтры h и g, использованные при построении вейвлет-кода, равны соответственно

$$\begin{aligned} h(x) = {} & 23 + 13x + 27x^2 + x^3 + 15x^4 + 13x^5 + x^6 + 16x^7 + x^8 + 21x^9 + 28x^{10} + 30x^{11} + \\ & + 12x^{12} + 19x^{13} + 17x^{14} + 4x^{15} + x^{16} + 19x^{17} + 14x^{18} + 3x^{20} + 5x^{21} + 6x^{22}, \\ g(x) = {} & 25 + x + 10x^2 + 16x^3 + 3x^4 + 2x^5 + 2x^6 + 20x^7 + 19x^8 + 8x^9 + 13x^{10} + 3x^{11} + \\ & + 12x^{12} + 21x^{13} + 7x^{14} + 3x^{15} + 31x^{16} + 25x^{17} + 22x^{18} + 31x^{19} + 12x^{20} + 30x^{21} + \\ & + 18x^{22} + 6x^{23} + 24x^{24} + 4x^{25} + 31x^{26} + 14x^{27} + 15x^{28} + 16x^{29} + 21x^{30}. \end{aligned}$$

Для того чтобы при помощи леммы 1 получить кодовое расстояние построенного вейвлет-кода, вычислим значения кодового многочлена $f(x)$ в точках $1, \alpha, \ldots, \alpha^{30}$. Получаем, что $f(\alpha^j) = 0$ при $j = 0, \ldots, 2$, следовательно, параметры j^* и d равны 0 и 13 соответственно, поэтому, согласно лемме 1, рассматриваемый вейвлет-код имеет кодовое расстояние 15 и может быть обозначен как $W[31, 15, 15]$.

Для иллюстрации работы алгоритма рассмотрим кодовое слово

$$c = (0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0)$$

и соответствующее ему зашумлённое кодовое слово

$$\widetilde{c} = (1,3,1,8,6,12,19,17,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0).$$

Описанный в п. 1 алгоритм позволяет найти все информационные слова вейвлет-кода $W[31, 15, 15]$, соответствующие кодовые слова которых попадают в шар радиуса 8 с центром в зашумлённом кодовом слове $\widetilde{c}$. На первом шаге алгоритм преобразует принятое зашумлённое кодовое слово $\widetilde{c}$ вейвлет-кода $W[31, 15, 15]$ в зашумлённое кодовое слово кода Рида — Соломона $\mathrm{RS}[31, 17]$

$$\widetilde{s} = (1,5,5,3,29,28,2,20,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0).$$

На втором шаге алгоритм применяет к зашумлённому кодовому слову $\widetilde{s}$ процедуру списочного декодирования кодов Рида — Соломона и получает список информационных слов β кода Рида — Соломона $\mathrm{RS}[31, 17]$

$$(0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0),$$
$$(28,22,21,17,8,15,26,0,8,6,4,12,20,5,26,7,21).$$

На третьем шаге алгоритм решает систему из семнадцати линейных уравнений с пятнадцатью неизвестными (в связи с большой размерностью она не приведена) для каждого информационного слова β из списка, полученного на втором шаге, и получает список информационных слов v вейвлет-кода $W[31, 15, 15]$

$$(0,0,0,0,0,0,0,0,0,0,0,0,0,0,0),$$
$$(21,31,18,19,6,12,11,8,25,9,24,8,6,5,22).$$

Полученным информационным словам соответствуют кодовые слова

$$(0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0),$$
$$(12,3,1,8,6,12,19,17,31,0,2,0,5,0,17,0,0,27,0,23,0,0,0,0,0,0,0,11,0,0,0)$$

вейвлет-кода $W[31, 15, 15]$ с процедурой кодирования (1) и порождающим многочленом $f(x)$, каждое из которых попадает в шар радиуса 8 с центром в зашумленном кодовом слове $\widetilde{c}$.

Заключение

В работе доказано, что вейвлет-код, определённый над полем $\mathrm{GF}(2^m)$, с длиной кодовых и информационных слов $n = 2^m - 1$ и $(n-1)/2$ соответственно и процедурой кодирования (1), с порождающим многочленом $f(x)$, для которого выполняются соотношения

$$f(\alpha^j) = 0 \ \text{ при } \ j = j^*, \ldots, j^* + d,\ 0 < d < \frac{n-3}{2},$$

имеет кодовое расстояние $\geqslant d + 2$ и допускает списочное декодирование. На основании приведённых доказательств описаны шаги алгоритма, позволяющего осуществлять списочное декодирование вейвлет-кодов.

Алгоритм реализован в виде программы, реализующей декодирование вейвлет-кода $W[n, (n-1)/2, d+2]$ с исправлением до $e < n - \sqrt{n(n-d-2)}$ ошибок за полиномиальное время от параметров n и d (получено авторское свидетельство № 2017619148). Для осуществления списочного декодирования кода Рида — Соломона используется улучшенная версия алгоритма Гурусвами — Судана из [19]. Написанная программа была успешно применена для декодирования зашумлённых кодовых слов кодов $W[7, 3, 4]$ и $W[31, 15, 15]$.

Доказано, что неравенство Варшамова — Гилберта не позволяет судить о существовании вейвлет-кода $W[n, (n-1)/2, d+2]$, $0 < d < (n-3)/2$, над полем характеристики два с кодовым расстоянием $(n-1)/2$ при достаточно больших n.

ЛИТЕРАТУРА

1. *Мак-Вильямс Ф. Дж., Слоэн Н. Дж. А.* Теория кодов, исправляющих ошибки. М.: Связь, 1974. 744 с.
2. *Fekri F., McLaughlin S. W., Mersereau R. M., and Schafer R. W.* Double circulant self-dual codes using finite-field wavelet transforms // Applied Algebra, Algebraic Algorithms and Error-Correcting Codes. Berlin: Springer, 1999. P. 355–363.
3. *Fekri F., McLaughlin S. W., Mersereau R. M., and Schafer R. W.* Error Control Coding using Finite-Field Wavelet Transforms. Atlanta: Center for Signal Image Processing, 1999. 13 p.
4. *Daubechies I.* Десять лекций по вейвлетам. Ижевск: НИЦ «Регулярная и хаотическая динамика», 2001. 464 с.
5. *Mallat S.* Wavelet Tour of Signal Processing. 2nd ed. Boston: Academic Press, 1999. 799 p.

6. *Phoong S. M. and Vaidyanathan P. P.* Paraunitary filter banks over finite fields // IEEE Trans. Signal Processing. 1997. V. 45. No. 6. P. 1443–1457.
7. *Fekri F., Mersereau R. M., and Schafer R. W.* Theory of paraunitary filter banks over fields of characteristic two // IEEE Trans. Inform. Theory. 2002. V. 48. No. 11. P. 2964–2979.
8. *Fekri F. and Delgosha F.* Finite-Field Wavelet Transforms with Applications in Cryptography and Coding. Upper Saddle River: Prentice Hall, 2010. 304 p.
9. *Caire G., Grossman R. L., and Poor H. V.* Wavelet transforms associated with finite cyclic groups // IEEE Trans. Inform. Theory. 1993. V. 39. No. 4. P. 1157–1166.
10. *Fekri F., Mersereau R. M., and Schafer R. W.* Theory of wavelet transform over finite fields // Proc. IEEE Intern. Conf. Acoustics, Speech, and Signal Processing. 1999. V. 3. P. 1213–1216.
11. *Черников Д. В.* Помехоустойчивое кодирование с использованием биортогональных наборов фильтров // Сибирские электрон. матем. известия. 2015. Т. 12. С. 704–713.
12. *Doubechies I. and Sweldens W.* Factoring wavelet transforms into lifting steps // J. Fourier Anal. Appl. 1998. V. 4. No. 3. P. 247–269.
13. *Соловьев А. А., Черников Д. В.* Биортогональные вейвлет-коды в полях характеристики два // Челяб. физ.-мат. журн. 2017. Т. 2. № 1. С. 66–79.
14. *Берлекэмп Э.* Алгебраическая теория кодирования. М.: Мир, 1971. 479 с.
15. *Сидельников В. М.* Теория кодирования. М.: Физматлит, 2008. 324 с.
16. *Berlekamp E. R. and Welch L. R.* Error Correction of Algebraic Block Codes. US Patent 4633470A. 30.12.1986.
17. *Ромащенко А. Е., Румянцев А. Ю., Шень А.* Заметки по теории кодирования. М.: МЦНМО, 2011. 80 с.
18. *Sudan M.* Decoding of Reed Solomon codes beyond the error-correction bounds // J. Complexity. 1997. V. 13. No. 1. P. 180–193.
19. *Guruswami V. and Sudan M.* Improved decoding of Reed — Solomon and algebraic-geometric codes // IEEE Trans. Inform. Theory. 1999. V. 45. No. 6. P. 1757–1767.
20. *Литичевский Д. В.* Списочное декодирование биортогональных вейвлет-кодов с заданным кодовым расстоянием в поле нечётной характеристики // Прикладная дискретная математика. 2018. № 39. С. 72–77.
21. *Соловьев А А.* Комплементарное представление многочленов над конечными полями // Челяб. физ.-мат. журн. 2017. Т. 2. Вып. 2. С. 199–209.

REFERENCES

1. *MacWilliams F. J. and Sloane N. J. A.* The Theory of Error-Correcting Codes. Elsevier, 1977. 744 p.
2. *Fekri F., McLaughlin S. W., Mersereau R. M., and Schafer R. W.* Double circulant self-dual codes using finite-field wavelet transforms. Applied Algebra, Algebraic Algorithms and Error-Correcting Codes. Berlin, Springer, 1999, pp. 355–363.
3. *Fekri F., McLaughlin S. W., Mersereau R. M., and Schafer R. W.* Error Control Coding using Finite-Field Wavelet Transforms. Atlanta, Center for Signal Image Processing, 1999, 13 pp.
4. *Daubechies I.* Desyat' lektsiy po veyvletam [Ten Lectures on Wavelets]. Izhevsk, SRC "Regular and Chaotic Dynamics", 2001. 464 p. (in Russian)
5. *Mallat S.* Wavelet Tour of Signal Processing, 2nd ed. Boston, Academic Press, 1999. 799 p.
6. *Phoong S. M. and Vaidyanathan P. P.* Paraunitary filter banks over finite fields. IEEE Trans. Signal Processing, 1997, vol. 45, no. 6, pp. 1443–1457.
7. *Fekri F., Mersereau R. M., and Schafer R. W.* Theory of paraunitary filter banks over fields of characteristic two. IEEE Trans. Inform. Theory, 2002, vol. 48, no. 11, pp. 2964–2979.

8. *Fekri F. and Delgosha F.* Finite-Field Wavelet Transforms with Applications in Cryptography and Coding. Upper Saddle River, Prentice Hall, 2010. 304 p.
9. *Caire G., Grossman R. L., and Poor H. V.* Wavelet transforms associated with finite cyclic groups. IEEE Trans. Inform. Theory, 1993. vol. 39, no. 4, pp. 1157–1166.
10. *Fekri F., Mersereau R. M., and Schafer R. W.* Theory of wavelet transform over finite fields // Proc. IEEE Intern. Conf. Acoustics, Speech, and Signal Processing, 1999, vol. 3, pp. 1213–1216.
11. *Chernikov D. V.* Pomekhoustoychivoye kodirovaniye s ispol'zovaniyem biortogonal'nykh naborov fil'trov [Error-correcting codes using biorthogonal filter banks]. Siberian Electronic Mathematical Rep., 2015, vol. 12, pp. 704–713. (in Russian)
12. *Doubechies I. and Sweldens W.* Factoring wavelet transforms into lifting steps. J. Fourier Anal. Appl., 1998, vol. 4, no. 3, pp. 247–269.
13. *Soloviev A. A. and Chernikov D. V.* Biortogonal'nyye veyvlet kody v polyakh kharakteristiki dva [Biorthogonal wavelet codes in the fields of characteristic two]. Chelyabinsk Physics and Mathematics J., 2017, vol. 2, no. 1, pp. 66–79. (in Russian)
14. *Berlekamp E. R.* Algebraic Coding Theory. N.Y., McGraw-Hill Book Company, 1968. 470 p.
15. *Sidelnikov V. M.* Teoriya kodirovaniya [Theory of Coding]. Moscow, Fizmatlit Publ., 2008. 324 p. (in Russian)
16. *Berlekamp E. R. and Welch L. R.* Error Correction of Algebraic Block Codes. US Patent 4633470A, 30.12.1986.
17. *Romashchenko A. E., Rumyantsev A. J., and Shen A.* Zametki po teorii kodirovaniya [Notes on the theory of coding]. Moscow, MCCME Publ., 2011. 80 p. (in Russian)
18. *Sudan M.* Decoding of Reed Solomon codes beyond the error-correction bounds. J. Complexity, 1997, vol. 13, no. 1, pp. 180–193.
19. *Guruswami V. and Sudan M.* Improved decoding of Reed — Solomon and algebraic-geometric codes. IEEE Trans. Inform. Theory, 1999, vol. 45, no. 6, pp. 1757–1767.
20. *Litichevskiy D. V.* Spisochnoye dekodirovaniye biortogonal'nykh veyvlet-kodov s zadannym kodovym rasstoyaniyem v pole nechetnoy kharakteristiki [List decoding of the biorthogonal wavelet code with predetermined code distance on a field with odd characteristic]. Prikladnaya Diskretnaya Matematika, 2018, no. 39. pp. 72–77. (in Russian)
21. *Soloviev A. A.* Komplementarnoe predstavlenie polinomov nad konechnymi polyami [Complementary representation of polynomials over finite fields]. Chelyabinsk Physics and Mathematics J., 2017, iss. 2, pp. 199–209. (in Russian)

МАТЕМАТИЧЕСКИЕ ОСНОВЫ ИНФОРМАТИКИ И ПРОГРАММИРОВАНИЯ

УДК 510.52

О ГЕНЕРИЧЕСКОЙ НЕРАЗРЕШИМОСТИ ДЕСЯТОЙ ПРОБЛЕМЫ ГИЛЬБЕРТА ДЛЯ ПОЛИНОМИАЛЬНЫХ ДЕРЕВЬЕВ[1]

А. Н. Рыбалов

Институт математики им. С. Л. Соболева СО РАН, г. Омск, Россия

Генерический подход к алгоритмическим проблемам предложен Каповичем, Мясниковым, Шуппом и Шпильрайном в 2003 г. В рамках этого подхода изучается поведение алгоритмов на множестве «почти всех» входов (это множество называется генерическим) и игнорируется его поведение на остальных входах, на которых алгоритм может работать медленно или вообще не останавливаться. В работе изучается генерическая сложность десятой проблемы Гильберта для диофантовых уравнений, представляемых в виде полиномиальных деревьев. Полиномиальное дерево — это бинарное дерево, листья которого помечены переменными или константой 1, а внутренние вершины содержат операции сложения, вычитания или умножения. Любой полином от многих переменных с целыми коэффициентами можно представить в виде такого полиномиального дерева. Доказывается, что проблема разрешимости диофантовых уравнений, представляемых в виде полиномиальных деревьев, является генерически неразрешимой.

Ключевые слова: *генерическая сложность, диофантовы уравнения.*

DOI 10.17223/20710410/44/8

ON GENERIC UNDECIDABILITY OF HILBERT'S TENTH PROBLEM FOR POLYNOMIAL TREES

A. N. Rybalov

Sobolev Institute of Mathematics, Omsk, Russia

E-mail: alexander.rybalov@gmail.com

Generic-case approach to algorithmic problems was suggested by Miasnikov, Kapovich, Schupp and Shpilrain in 2003. This approach studies behavior of an algorithm on typical (almost all) inputs and ignores the rest of inputs. We study generic complexity of the Hilbert's tenth problem for systems of Diophantine equations represented by so-called polynomial trees. Polynomial tree is a binary tree, which leafs are marked by variables or the constant 1, and internal vertices are marked by operations of addition, subtraction and multiplication. Every polynomial with integer coefficients can be represented by a polynomial tree. We prove generic undecidability of the decidability problem for Diophantine equations represented by polynomial trees. To prove

[1]Исследование поддержано программой фундаментальных научных исследований СО РАН I.1.1.4, проект № 0314-2019-0004.

this theorem, we use the method of generic amplification, which allows to construct generically undecidable problems from the problems undecidable in the classical sense. The main ingredient of this method is a technique of cloning, which unites inputs of the problem together in the large enough sets of equivalent inputs. Equivalence is understood in the sense that the problem is solved similarly for them.

Keywords: *generic complexity, Diophantine equations.*

Введение

Генерический подход в применении к алгоритмическим проблемам впервые предложен в 2003 г. в [1]. В рамках этого подхода изучается поведение алгоритма на множестве «почти всех» входов (это множество называется генерическим) и игнорируется его поведение на остальных входах, на которых алгоритм может работать медленно или вообще не останавливаться. Такой подход имеет приложение в криптографии, где требуется, чтобы алгоритмические проблемы были трудными для «почти всех» входов.

В 1970 г. Ю. В. Матиясевич [2], основываясь на работах М. Дэвиса, Дж. Робинсон и Х. Патнема, доказал, что проблема определения разрешимости диофантовых уравнений в целых числах, известная как десятая проблема Гильберта, алгоритмически неразрешима. В [3 – 5] показано, что основные функции шифрования многих криптографических систем с открытым ключом, среди которых система RSA и системы, основанные на трудноразрешимости проблемы дискретного логарифма, записываются на языке диофантовых уравнений. Эффективная генерическая разрешимость этих уравнений приводит к взлому соответствующих систем, поэтому актуальной является задача изучения генерической сложности проблемы разрешимости диофантовых уравнений в различных её постановках. Например, в [6, 7] доказано, что десятая проблема Гильберта остаётся неразрешимой на строго генерических подмножествах входов при представлении диофантовых уравнений с помощью так называемых арифметических схем. В [8] изучена генерическая сложность десятой проблемы Гильберта для систем диофантовых уравнений в форме Сколема.

В данной работе изучается генерическая сложность десятой проблемы Гильберта для диофантовых уравнений, представляемых в виде полиномиальных деревьев. Полиномиальное дерево — это бинарное дерево, листья которого помечены переменными или константой 1, а внутренние вершины содержат операции сложения, вычитания или умножения. Любой полином от многих переменных с целыми коэффициентами можно представить в виде такого полиномиального дерева. В работе доказывается, что проблема разрешимости диофантовых уравнений, представляемых в виде полиномиальных деревьев, является генерически неразрешимой.

1. Генерические алгоритмы

Пусть I — множество всех входов, а I_n — множество входов размера n. Для любого подмножества $S \subseteq I$ определим последовательность

$$\rho_n(S) = \frac{|S_n|}{|I_n|},\ n = 1, 2, 3, \ldots,$$

где $S_n = S \cap I_n$ — множество входов из S размера n. Здесь $|A|$ — число элементов в множестве A. *Асимптотической плотностью* S назовём предел (если он существует)

$$\rho(S) = \lim_{n\to\infty} \rho_n(S).$$

Множество S называется *генерическим*, если $\rho(S) = 1$, и *пренебрежимым*, если $\rho(S) = 0$. Очевидно, что S генерическое тогда и только тогда, когда его дополнение $I \setminus S$ пренебрежимо.

Алгоритм $\mathcal{A}$ с множеством входов I и множеством выходов $J \cup \{?\}$ $(? \notin J)$ называется *генерическим*, если

1) $\mathcal{A}$ останавливается на всех входах из I;
2) множество $\{x \in I : \mathcal{A}(x) = ?\}$ является пренебрежимым.

Здесь через $\mathcal{A}(x)$ обозначается результат работы алгоритма $\mathcal{A}$ на входе x. Генерический алгоритм $\mathcal{A}$ *вычисляет функцию* $f : I \to J$; если для $x \in I$ не выполнено $\mathcal{A}(x) = ?$, то $f(x) = \mathcal{A}(x)$. Ситуация $\mathcal{A}(x) = ?$ означает, что $\mathcal{A}$ не может вычислить функцию f на аргументе x. Но условие 2 гарантирует, что $\mathcal{A}$ корректно вычисляет f на почти всех входах (входах из генерического множества). Множество $S \subseteq I$ называется *генерически разрешимым*, если существует генерический алгоритм, вычисляющий его характеристическую функцию.

2. Представление диофантовых уравнений в виде полиномиальных деревьев

Здесь и далее под размером двоичного дерева понимается число листьев. *Полиномиальным деревом* называется бинарное дерево, листья которого помечены переменными из множества $\{x_1, \ldots, x_n\}$ (n — размер дерева) или константой 1, а внутренние вершины помечены знаками операций сложения, вычитания или умножения. Каждое полиномиальное дерево Φ с n листьями представляет некоторый полином $P_\Phi(x_1, \ldots, x_n)$ с целыми коэффициентами следующим образом:

1) $P_\Phi(x_1, \ldots, x_n) = 1$, если Φ состоит из одного листа, помеченного константой 1;
2) $P_\Phi(x_1, \ldots, x_n) = x_i$, если Φ состоит из одного листа, помеченного переменной x_i;
3) $P_\Phi(x_1, \ldots, x_n) = P_\Psi(x_1, \ldots, x_n) * P_\Sigma(x_1, \ldots, x_n)$, если Φ состоит из левого поддерева Ψ, правого поддерева Σ и корня, помеченного операцией $* \in \{+, -, \times\}$.

С другой стороны, легко видеть, что для любого полинома с целыми коэффициентами $P(x_1, \ldots, x_n)$ найдется полиномиальное дерево Φ (возможно, размера больше n) такое, что $P(x_1, \ldots, x_n) = P_\Phi(x_1, \ldots, x_n)$. Таким образом, будем представлять диофантовы уравнения вида $P(x_1, \ldots, x_n) = 0$ с помощью полиномиальных деревьев.

Будем называть полиномиальное дерево *нормализованным*, если при нумерации листьев дерева слево направо лист L_k может быть помечен переменной x_i, где $i \leqslant k$. Очевидно, что любое полиномиальное дерево можно нормализовать при помощи подходящего перенумерования переменных. В дальнейшем будем рассматривать только нормализованные полиномиальные деревья. Обозначим через $\mathcal{T}$ множество всех нормализованных полиномиальных деревьев.

Напомним, что числа Каталана C_n определяются следующим образом:

$$C_n = \frac{1}{n+1}\binom{2n}{n}.$$

Здесь $\binom{2n}{n}$ — биномиальный коэффициент.

Лемма 1. Число нормализованных полиномиальных деревьев размера n есть

$$|\mathcal{T}_n| = 3^{n-1}(n+1)!C_{n-1}.$$

Доказательство. Любое дерево из $\mathcal{T}$ размера n есть размеченное бинарное дерево с n листьями и $n-1$ внутренними вершинами. Известно (см., например, [9]), что существует C_{n-1} неразмеченных бинарных деревьев с n листьями. Каждая внутренняя вершина такого дерева может быть помечена тремя символами $\{+,-,\times\}$, поэтому есть всего 3^{n-1} таких разметок. Занумеруем листья дерева слева направо. Из условия нормализованности следует, что лист с номером k может быть помечен одной из k переменных $x_1,\ldots,x_k$ или константой 1, поэтому существует $(n+1)!$ разметок для листьев. Это показывает, что $|\mathcal{T}_n| = 3^{n-1}(n+1)!C_{n-1}$. ■

В дальнейшем понадобится следующее утверждение о числах Каталана.

Лемма 2. Для $n > m$ имеет место

$$\frac{C_{n-m}}{C_n} > \frac{1}{4^m}.$$

Доказательство. Оценим отношение чисел Каталана:

$$\frac{C_{n-m}}{C_n} = \frac{n+1}{n-m+1}\,\frac{\binom{2(n-m)}{n-m}}{\binom{2n}{n}} = \frac{n+1}{n-m+1}\,\frac{\dfrac{(2(n-m))!}{(n-m)!(n-m)!}}{\dfrac{(2n)!}{n!n!}} >$$

$$> \frac{(2(n-m))!n!n!}{(n-m)!(n-m)!(2n)!} = \frac{n!}{(n-m)!}\,\frac{2(n-m)\ldots(n-m+1)}{2n\ldots(n+1)} =$$

$$= \frac{n(n-1)\ldots(n-m+1)2(n-m)\ldots(n-m+1)}{2n\ldots(n+1)} =$$

$$= \frac{(n\ldots(n-m+1))^2}{2n\ldots(2(n-m)+1)} > \Big(\frac{(n-1)\ldots(n-m)}{2(n-1)\ldots(2n-2m)}\Big)^2 > \frac{1}{2^{2m}} = \frac{1}{4^m}.$$

Лемма доказана. ■

3. Основной результат

Для произвольного полиномиального дерева Φ рассмотрим множество $\mathrm{eq}(\Phi)$ полиномиальных деревьев вида, представленного на рис. 1, где Ψ — произвольное полиномиальное дерево.

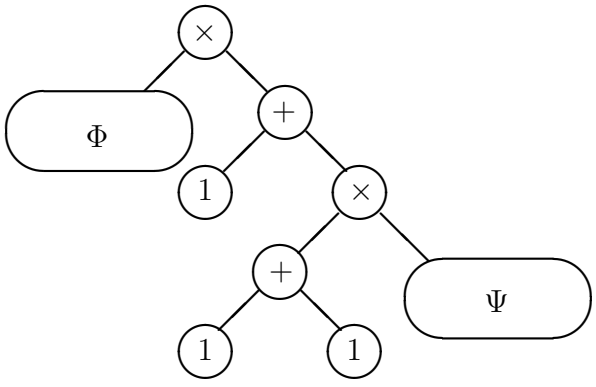

Рис. 1

Пусть m — размер дерева Φ. Заметим, что для любого $\Sigma \in \mathrm{eq}(\Phi)$ размера $n > m$ имеет место

$$P_\Sigma(x_1,\ldots,x_n) = P_\Phi(x_1,\ldots,x_m)(1+2P_\Psi(x_1,\ldots,x_n)).$$

Из этого следует, что диофантово уравнение $P_\Sigma(x_1,\dots,x_m)=0$ разрешимо в целых числах тогда и только тогда, когда разрешимо в целых числах диофантово уравнение $P_\Phi(x_1,\dots,x_k)=0$.

Лемма 3. Для любого полиномиального дерева Φ множество $\mathrm{eq}(\Phi)$ не является пренебрежимым.

Доказательство. Пусть m — размер дерева Φ и $n > m+3$. Любое дерево $\Sigma \in \mathrm{eq}(\Phi)_n$ имеет вид, приведённый на рис. 1. В нём произвольно может выбираться только поддерево Ψ с числом листьев $n-m-3$. Посчитаем, сколькими способами это можно сделать. Есть C_{n-m-4} неразмеченных бинарных деревьев с $n-m-3$ листьями. Внутренние $n-m-4$ вершин можно пометить арифметическими операциями 3^{n-m-4} способами. Занумеруем листья дерева Φ слева направо. Из условия нормализованности следует, что лист поддерева Ψ с номером $k > m+3$ может быть помечен одной из k переменных $x_1,\dots,x_k$ или константой 1, поэтому существует $(m+5)(m+6)\dots(n+1)$ разметок для листьев поддерева Ψ. Итого имеем

$$|\mathrm{eq}(\Phi)_n| = 3^{n-m-4}(m+5)(m+6)\dots(n+1)C_{n-m-4}.$$

По лемме 1 $|\mathcal{T}_n| = 3^{n-1}(n+1)!C_{n-1}$. Следовательно,

$$\rho_n(\mathrm{eq}(\Phi)) = \frac{|\mathrm{eq}(\Phi)_n|}{|\mathcal{T}_n|} = \frac{3^{n-m-4}(m+5)(m+6)\dots(n+1)C_{n-m-4}}{3^{n-1}(n+1)!C_{n-1}} =$$
$$= \frac{1}{3^{m+3}(m+4)!}\,\frac{C_{n-m-4}}{C_{n-1}} > \frac{1}{3^{m+3}(m+4)!}\,\frac{1}{4^{m+3}} = \mathrm{const} > 0.$$

Здесь использована лемма 2 для оценки отношения чисел Каталана. Таким образом, множество $\mathrm{eq}(\Phi)$ не является пренебрежимым. ■

Теперь докажем основной результат статьи.

Теорема 1. Проблема разрешимости диофантовых уравнений, представляемых в виде полиномиальных деревьев, не является генерически разрешимой.

Доказательство. Допустим, что существует генерический алгоритм $\mathcal{A}$, определяющий разрешимость диофантовых уравнений, заданных полиномиальными деревьями, на некотором генерическом множестве полиномиальных деревьев. Используя этот алгоритм, построим алгоритм $\mathcal{B}$, который будет определять разрешимость диофантовых уравнений для всех полиномиальных деревьев. Тем самым получим противоречие с неразрешимостью Десятой проблемы Гильберта.

Алгоритм $\mathcal{B}$ на дереве Φ работает следующим образом: перебирает элементы $\mathrm{eq}(\Phi)$ в порядке возрастания размера до тех пор, пока не получит дерево $\Sigma \in \mathrm{eq}(\Phi)$, такое, что $\mathcal{A}(\Sigma) \neq ?$. Ответ $\mathcal{A}(\Sigma)$ и будет правильным ответом для исходного дерева Φ.

То, что всегда найдётся такое дерево Σ, следует из того, что множество $\{\Psi \in \mathcal{T} : \mathcal{A}(\Psi) = ?\}$ пренебрежимо, а множество $\mathrm{eq}(\Phi)$, по лемме 3, не является пренебрежимым. Теорема доказана. ■

ЛИТЕРАТУРА

1. *Kapovich I., Miasnikov A., Schupp P., and Shpilrain V.* Generic-case complexity, decision problems in group theory and random walks // J. Algebra. 2003. V. 264. No. 2. P. 665–694.
2. *Матиясевич Ю. В.* Диофантовость перечислимых множеств // Доклады АН СССР. 1970. Т. 191. № 2. С. 279–282.
3. *Myasnikov A. and Romankov V.* Diophantine cryptography in free metabelian groups: Theoretical base // Groups, Complexity, Cryptology. 2014. V. 6. No. 2. P. 103–120.

4. *Романьков В. А.* Диофантова криптография на бесконечных группах // Прикладная дискретная математика. 2012. № 2 (16). С. 15–42.
5. *Романьков В. А.* Алгебраическая криптография. Омск: ОмГУ, 2013.
6. *Rybalov A.* Generic complexity of the Diophantine problem // Groups, Complexity, Cryptology. 2013. V. 5. No. 1. P. 25–30.
7. *Рыбалов А.* О генерической неразрешимости Десятой проблемы Гильберта // Вестник Омского университета. 2011. № 4. С. 19–22.
8. *Рыбалов А.* О генерической сложности проблемы разрешимости систем диофантовых уравнений в форме Сколема // Прикладная дискретная математика. 2017. № 37. С. 100–106.
9. *Кнут Д.* Искусство программирования. М.: Вильямс, 2010. 720 с.

REFERENCES

1. *Kapovich I., Miasnikov A., Schupp P., and Shpilrain V.* Generic-case complexity, decision problems in group theory and random walks. J. Algebra, 2003, vol. 264, no. 2, pp. 665–694.
2. *Matiyasevich Yu. V.* Diofantovost' perechislimykh mnozhestv [Diophantineity of enumerable sets]. Doklady Akademii Nauk USSR, 1970, vol. 191, no. 2, pp. 279–282. (in Russian)
3. *Myasnikov A. and Romankov V.* Diophantine cryptography in free metabelian groups: Theoretical base. Groups, Complexity, Cryptology, 2014, vol. 6, no. 2, pp. 103–120.
4. *Roman'kov V. A.* Diofantova kriptografiya na beskonechnykh gruppakh [Diophantine cryptography over infinite groups]. Prikladnaya Diskretnaya Matematika, 2012, no. 2 (16), pp. 15–42. (in Russian)
5. *Roman'kov V. A.* Algebraicheskaya Kriptografiya [Algebraic Cryptography]. Omsk, OmSU Publ., 2013. (in Russian)
6. *Rybalov A.* Generic complexity of the Diophantine problem. Groups, Complexity, Cryptology, 2013, vol. 5, no. 1, pp. 25–30.
7. *Rybalov A.* O genericheskoy nerazreshimosti Desyatoy problemy Gil'berta [On generic undecidability of Hilbert Tenth problem]. Vestnik Omskogo Universiteta, 2011, no. 4, pp. 19–22. (in Russian)
8. *Rybalov A.* O genericheskoy slozhnosti problemy razreshimosti sistem diofantovyh uravneniy v forme Skolema [On generic complexity of decidability problem for diophantine systems in the Skolem's form]. Prikladnaya Diskretnaya Matematika, 2017, no. 37, pp. 100–106. (in Russian)
9. *Knuth D. E.* The Art of Computer Programming. Reading, Massachusetts, Addison-Wesley, 1997.

ВЫЧИСЛИТЕЛЬНЫЕ МЕТОДЫ В ДИСКРЕТНОЙ МАТЕМАТИКЕ

УДК 519.7

ДЕКОМПОЗИЦИОННЫЙ ПОДХОД К ИССЛЕДОВАНИЮ ФОРМАЛЬНЫХ КОНТЕКСТОВ

В. В. Быкова*, Ч. М. Монгуш**

*Сибирский федеральный университет, г. Красноярск, Россия

**Тувинский государственный университет, г. Кызыл, Республика Тыва, Россия

Исследуется #P-полная задача нахождения всех формальных понятий заданного контекста и предлагается декомпозиционный метод её решения. В качестве частей разложения предлагается использовать фрагменты исходного контекста, названные боксами. Доказано, что разделение контекста на боксы «безопасно» относительно формальных понятий: при декомпозиции ни одно формальное понятие не теряется и не появляются новые формальные понятия. Доказано, что число боксов, возникающих на каждой итерации разложения, равно числу единичных элементов 0,1-матрицы, представляющей исходный формальный контекст. Предлагается уменьшать число боксов на каждой отдельной итерации процесса декомпозиции с помощью построения взаимно непересекающихся цепей боксов. Приводятся результаты вычислительных экспериментов, свидетельствующие о существенном повышении производительности алгоритмов нахождения всех формальных понятий при применении предлагаемого декомпозиционного метода.

Ключевые слова: *анализ формальных понятий, декомпозиция формального контекста.*

DOI 10.17223/20710410/44/9

DECOMPOSITIONAL APPROACH TO RESEARCH OF FORMAL CONTEXTS

V. V. Bykova*, Ch. M. Mongush**

*Siberian Federal University, Krasnoyarsk, Russia

**Tuvan State University, Kyzyl, Tuva, Russia

E-mail: bykvalen@mail.ru, mongushchod91@yandex.ru

The problem of finding all formal concepts of a given formal context is investigated. The problem arises when data mining is presented in the form of a binary object-attribute matrix, i.e. a matrix the rows of which correspond to objects, and the columns correspond to features that take a value from the two-element set $\{0, 1\}$. Here the value 1 of the element of a matrix is interpreted as the presence of corresponding attribute to the object, and 0 is as its absence. Such a representation of a set of data allows to be used the algebraic approach of R. Wille and B. Ganter, known in the literature as Formal Concept Analysis. Within of this approach the

initial object-attribute matrix is called the formal context, and any of its maximal full submatrix is a formal concept. In the problem of finding all formal concepts, it is required to find the set of all formal concepts for a given formal context. This problem belongs to combinatorial enumeration problems and is #P-complete. The high computational complexity of the problem is due to the fact that in the general case the number of formal concepts exponentially depends on the size of the initial formal context. Currently many algorithms have been developed to solve the problem, among them NextClosure, Close-by-One, Norri. The execution time of these algorithms in the worst case exponentially depends on the dimension of the initial context, and therefore they are unsuitable for practical analysis of contexts of large dimension. We propose a decomposition method for solving the problem under consideration. In this method, some fragments of the initial context defined in a certain constructive way are called boxes. We prove that the division of context into boxes is "safety" relatively of the formal concepts. This means that the formal concepts are not lost and new formal concepts do not arise at decomposition. We prove that the number of boxes arising at each iteration of the decomposition is equal to the number of unit elements of the 0,1-matrix representing the initial formal context. We show how a partial order relation can be defined on a set of boxes. We also show that the number of boxes at each separate iteration of the decomposition process can be reduced by using constructing mutually disjoint chains of boxes. The results of computational experiments are given, indicating that the application of the proposed decomposition method allows significantly to increase the performance of algorithms for finding all formal concepts of a given context.

Keywords: *formal concept analysis, formal context decomposition.*

Введение

Во многих задачах интеллектуального анализа данных изучаемая предметная область описывается с помощью объектно-признаковой таблицы, в которой каждый столбец соответствует некоторому признаку, а каждая строка определяет признаковое описание отдельного объекта. Подобное представление множества данных позволяет при их обработке применять математический аппарат общей алгебры и теории графов, например, объектно-признаковые таблицы можно исследовать с помощью методов анализа формальных понятий. Анализ формальных понятий (АФП, англ. Formal Concept Analysis — FCA) как прикладное направление теории решеток Г. Биркгофа возникло с появлением работ Б. Гантера и Р. Вилле [1 – 3]. В АФП объектно-признаковая таблица представляется формальным контекстом, отражающим наличие или отсутствие признаков, характерных для изучаемого множества объектов, и моделируется 0,1-матрицей. Каждое формальное понятие определяется парой замкнутых множеств, интерпретируемых как объём и содержание этого понятия. В матричной форме формальному понятию соответствует некоторая максимально полная подматрица 0,1-матрицы, представляющей формальный контекст.

С помощью методов АФП решаются различные прикладные задачи, связанные с классификацией и кластеризацией данных, выявлением зависимостей между данными и семантическим анализом естественно-языковых текстов [4 – 7]. В них формальные понятия трактуются как перекрёстные ассоциации, кластеры или бикластеры. В рамках АФП решение указанных задач сводится к нахождению всех формальных понятий исходного формального контекста с последующим связыванием их в решётку. Полу-

ченная решётка служит концептуальной моделью исследуемой предметной области и основой для решения прикладных задач.

При всей привлекательности методов АФП их практическое применение ограничивается высокой трудоёмкостью процесса извлечения множества формальных понятий из исходного контекста. Задача нахождения всех формальных понятий формального контекста детально изучена в [2 – 9]. Известно, что данная задача относится к классу #P-полных задач, поскольку число формальных понятий может экспоненциально зависеть от размера исходного контекста [4]. В настоящее время актуальны исследования по повышению производительности алгоритмов установления всех формальных понятий за счёт параллельных вычислений и применения декомпозиционного подхода [3, 5]. Основная цель этих исследований — сделать более доступными методы АФП для анализа больших данных.

В данной работе рассматривается задача нахождения всех формальных понятий заданного контекста. Предлагается декомпозиционный метод её решения, в котором частями разложения выступают фрагменты исходного контекста, названные боксами. Такая декомпозиция позволяет разлагать контекст без потери формальных понятий и тем самым снижать время выполнения алгоритмов решения рассматриваемой задачи. Исследуется структура боксов с целью оценки числа фрагментов разложения, получаемых на каждой итерации декомпозиции. Устанавливаются правила остановки итерационного процесса разложения. Приводятся результаты вычислительных экспериментов, подтверждающих результативность предложенного метода декомпозиции.

1. Основные положения и обозначения

Рассмотрим основные положения и типовые обозначения АФП [2, 3].

Пусть определены два непустых конечных множества: множество объектов G (нем. Gegenstände) и множество признаков или свойств M (нем. Merkmale). Пусть также задано непустое отношение инцидентности $I \subseteq G \times M$. Данное отношение содержит информацию о выполнимости свойств из M на объектах из G, т. е. $(g, m) \in I$ означает, что объект g обладает признаком m, и наоборот — признак m присущ объекту g. Тройку $K = (G, M, I)$ принято называть формальным контекстом.

Далее будем полагать, что множества G и M линейно упорядочены (например, лексикографически). В этом случае формальный контекст $K = (G, M, I)$ однозначно задаётся 0,1-матрицей $T = (t_{ij})$: $t_{ij} = 0$ при $(g_i, m_j) \notin I$ и $t_{ij} = 1$ при $(g_i, m_j) \in I$ $(i = 1, 2, \ldots, |G|;\ j = 1, 2, \ldots, |M|)$.

Выберем в $K = (G, M, I)$ два произвольных элемента $g \in G$, $m \in M$ и определим для них отображения $(\cdot)'$ следующим образом:

$$g' = \{m \in M : (g, m) \in I\}, \quad m' = \{g \in G : (g, m) \in I\}. \tag{1}$$

Согласно этому определению, множество g' устанавливает набор признаков, присущих объекту g, а множество m' задает семейство объектов, обладающих признаком m. Отображения $(\cdot)'$ легко обобщаются на множества $A \subseteq G$ и $B \subseteq M$:

$$A' = \bigcap_{g \in A} g', \quad B' = \bigcap_{m \in B} m'.$$

Для всякого формального контекста $K = (G, M, I)$ и любых подмножеств $B_1, B_2 \subseteq M$ верны следующие свойства:

— *антимонотонность*: если $B_1 \subseteq B_2$, то $(B_2)' \subseteq (B_1)'$;

— *экстенсивность*: $B_1 \subseteq (B_1)''$, где $(B_1)'' = ((B_1)')' \subseteq M$.

Аналогичные свойства справедливы для подмножеств множества G. Известно, что отображения $(\cdot)'$ являются соответствиями Галуа и для них верны следующие равенства [1]:

$$\big((A')'\big)' = (A'')' = A', \quad ((B')')' = (B'')' = B'. \tag{2}$$

Двойное применение $(\cdot)'$ определяет оператор замыкания $(\cdot)''$ на 2^M в алгебраическом смысле. Этому оператору присущи следующие свойства:

— *рефлексивность*: для любого $B \subseteq M$ всегда $B \subseteq B''$;
— *монотонность*: если $B_1 \subseteq B_2 \subseteq M$, то $(B_1)'' \subseteq (B_2)'' \subseteq M$;
— *идемпотентность*: для любого $B \subseteq M$ всегда $(B'')'' = B''$.

Множество $(B)''$ можно трактовать как набор признаков, которые неизменно появляются в объектах формального контекста $K = (G, M, I)$ вместе с признаками из B, причём это множество является наибольшим по включению в пределах этого формального контекста. Если $B = B''$, то B называется замкнутым множеством относительно оператора $(\cdot)''$.

Пара множеств (A, B), $A \subseteq G$, $B \subseteq M$, таких, что $A' = B$ и $B' = A$, называется формальным понятием формального контекста $K = (G, M, I)$ с объёмом A и содержанием B. Далее в ряде случаев определение «формальный» перед словами «контекст» или «понятие» будет опускаться.

Из (2) и определения оператора $(\cdot)''$ вытекает справедливость следующего высказывания: пара множеств (A, B) является формальным понятием тогда и только тогда, когда $A = A''$ и $B = B''$. Очевидно также, что всякое формальное понятие уникально в заданном контексте, т. е. отличается от других формальных понятий объёмом и/или содержанием. Если формальный контекст представлен 0,1-матрицей T, то при $A \neq \varnothing$ и $B \neq \varnothing$ формальному понятию (A, B) отвечает максимальная полная подматрица матрицы T. Строки этой подматрицы соответствуют элементам из A, а столбцы — элементам из B. Здесь под полной подматрицей понимается подматрица, все элементы которой равны 1; полная подматрица является максимальной, если она не содержится в других полных подматрицах.

Обозначим через FC множество всех формальных понятий формального контекста $K = (G, M, I)$. Пусть $(A_1, B_1), (A_2, B_2) \in FC$. Множество FC частично упорядочено отношением

$$(A_1, B_1) \sqsubseteq (A_2, B_2)$$

тогда и только тогда, когда $A_1 \subseteq A_2$. Отметим, что последнее эквивалентно условию $B_2 \subseteq B_1$. Каждое формальное понятие $(A, B) \in FC$ определяет для исследуемой предметной области совокупность однородных объектов A со своим специфичным набором признаков B. Если в контексте $K = (G, M, I)$ нет признаков, которые присущи всем объектам из G, то множество FC содержит формальное понятие $(G, \varnothing)$. Если в контексте нет объектов, обладающих всеми признаками из M, то $(\varnothing, M) \in FC$. Если имеют место оба случая одновременно, то $(G, \varnothing) \in FC$ и $(\varnothing, M) \in FC$. Эти формальные понятия называются тривиальными.

По определению формального контекста $K = (G, M, I)$ отношение I не пустое. Следовательно, отвечающая формальному контексту матрица T всегда ненулевая, а соответствующее ему множество FC не является пустым.

Определим на FC операции пересечения $\sqcap$ и объединения $\sqcup$ через одноимённые теоретико-множественные операции $\cap$ и $\cup$ следующим образом:

$$(A_1, B_1) \sqcap (A_2, B_2) = (A_1 \cap A_2, (A_1 \cap A_2)'),$$
$$(A_1, B_1) \sqcup (A_2, B_2) = ((B_1 \cap B_2)', B_1 \cap B_2).$$

Тогда упорядоченное множество $(FC, \sqsubseteq)$ образует решётку $L = (FC, \sqcap, \sqcup)$, которая называется решёткой формальных понятий контекста $K = (G, M, I)$. Нулем решётки $L = (FC, \sqcap, \sqcup)$ является формальное понятие (M', M), содержащее все признаки контекста $K = (G, M, I)$, а единицей — формальное понятие (G, G'), в котором объём — множество всех объектов рассматриваемого контекста.

2. О задаче нахождения всех формальных понятий и родственных с ней задачах

В задаче нахождения всех формальных понятий требуется найти множество FC для заданного контекста $K = (G, M, I)$. Данная задача относится к комбинаторным перечислительным задачам и является #P-полной [4]. Высокая вычислительная сложность задачи обусловлена тем, что в общем случае число формальных понятий экспоненциально зависит от размера исходного контекста. Например, это имеет место для контекста вида $K = (G, G, \neq)$. Такому контексту соответствует 0,1-матрица, в которой все элементы равны единице, за исключением диагональных элементов. Легко убедиться, что такой контекст содержит ровно $2^{|G|}$ формальных понятий.

На сегодняшний день для определения множества FC и построения решётки $L = (FC, \sqcap, \sqcup)$ разработано много алгоритмов, в их числе NextClosure, Close-by-One, Norris [3 – 5]. Время их выполнения алгоритмов в худшем случае составляет $O(|FC| \cdot |G|^2 \cdot |M|)$. Поскольку величина $|FC|$ может экспоненциально зависеть от $|G|$ и $|M|$, то время выполнения также может быть экспоненциальным. Повысить производительность алгоритмов определения множества FC и построения решётки $L = (FC, \sqcap, \sqcup)$ можно за счёт параллельных вычислений и применения декомпозиционного подхода [3, 5].

Важно отметить, что задача нахождения всех формальных понятий контекста $K = (G, M, I)$ эквивалентна задаче определения всех максимальных полных подматриц 0,1-матрицы T, отвечающей этому контексту. Существуют и другие родственные с ней задачи, например задачи, связанные с нахождением биклик в заданном двудольном графе. В самом деле, бинарную матрицу T можно рассматривать в качестве матрицы смежности двудольного графа, две доли которого соответствуют множествам строк и столбцов матрицы T. Тогда всякая полная подматрица матрицы T определяет в заданном двудольном графе полный двудольный подграф, т. е. биклику, а максимальная полная подматрица — максимальную биклику этого графа. К поиску максимальных биклик сводятся следующие теоретико-графовые задачи:

- в заданном двудольном графе найти все максимальные биклики;
- для заданного двудольного графа найти наименьшее покрытие всех рёбер максимальными бикликами;
- в заданном двудольном графе найти наибольшую биклику;
- для заданного двудольного графа найти наименьшее бикликовое разбиение множества его вершин;
- является ли заданный двудольный граф (k, l)-редким? По определению такой двудольный граф не содержит биклик размера $k \times l$.

Все эти задачи относятся к классу #P-полных или NP-полных задач [10, 11]. Известно, что в общем случае число максимальных биклик графа экспоненциально зависит от числа вершин [12 – 14]. Доказано, что двудольный граф на n вершинах может содержать до $2^{n/2} \approx 1{,}41^n$ максимальных биклик [15, 16]. К родственным можно также отнести перечислительные задачи, связанные с нахождением всех неприводимых покрытий 0,1-матрицы и поиском информативных фрагментов описаний объектов в дискретных процедурах распознавания [17, 18]. Однако большинство известных алгоритмов решения этих задач неприемлемо долго работают на исходных данных большой размерности. Повысить производительность некоторых из них, а также существующих алгоритмов решения рассматриваемой задачи можно путём применения декомпозиционного подхода, излагаемого далее.

3. Метод декомпозиции контекста без потери формальных понятий

Декомпозиционный подход к решению задачи нахождения всех формальных понятий заданного контекста — это сведение её к конечной серии подзадач. Каждая из этих подзадач — уменьшенная копия исходной задачи, которая решается на некоторой части заданного контекста. Процесс декомпозиции направлен на последовательное уменьшение размеров частей контекста. В итоге формируется конечное множество различных частей (в общем случае разного размера и имеющих непустое пересечение). Процесс декомпозиции реализуется итерационно, поскольку рекурсия в подобных случаях более трудоёмка по времени [19]. Для эффективной организации процесса декомпозиции требуется определить правило разложения контекста на части (что является частью и как её выделять в контексте); оценку числа частей, получаемых на каждой итерации разложения; правило остановки процесса разложения. Кроме того, для всякого декомпозиционного метода решения задачи обязательны правила восстановления искомого решения из решений, полученных для подзадач. Полиномиальность по времени процедур разделения исходных данных решаемой задачи на части — требование, при выполнении которого достигается эффект декомпозиции.

Опишем предлагаемый метод декомпозиции формального контекста и докажем, что он позволяет разлагать контекст без потери формальных понятий, а также установим правила эффективной организации процесса декомпозиции.

Пусть $K = (G, M, I)$ — контекст, FC — множество всех его формальных понятий и T — соответствующая ему 0,1-матрица. Контекст $K_1 = (G_1, M_1, I_1)$ назовём частью $K = (G, M, I)$, если $G_1 \subseteq G$, $M_1 \subseteq M$ и для любых $x \in G_1$, $y \in M_1$ отношение $(x, y) \in I_1$ верно тогда и только тогда, когда $(x, y) \in I$. Заметим, что контексту $K_1 = (G_1, M_1, I_1)$ отвечает подматрица матрицы T, у которой удалены строки, соответствующие объектам из $G \setminus G_1$, и столбцы, соответствующие признакам из $M \setminus M_1$. Всякое нетривиальное формальное понятие из FC можно рассматривать в роли части контекста $K = (G, M, I)$. Части $K_1 = (G_1, M_1, I_1)$ и $K_2 = (G_2, M_2, I_2)$ контекста $K = (G, M, I)$ будем считать различными, если $G_1 \neq G_2$ и/или $M_1 \neq M_2$.

Требуется разложить контекст $K = (G, M, I)$ на конечное множество различных частей так, чтобы выполнялись следующие условия:

1) каждая часть содержит, по крайней мере, одно формальное понятие из FC;
2) ни одно формальное понятие из FC не теряется и не возникают новые формальные понятия.

Разложение, удовлетворяющее условиям 1 и 2, назовём «безопасным» относительно формальных понятий. Если 0,1-матрица T полная, то результирующее множество состоит только из одной части, представляющей сам контекст $K = (G, M, I)$, и эта

часть содержит только одно формальное понятие (G, M). Очевидно, что наибольшее число различных частей, на которые можно «безопасно» разложить контекст, равно числу $|FC|$ формальных понятий контекста $K = (G, M, I)$. Поскольку существуют контексты, для которых число формальных понятий экспоненциально зависит от $|G|$ и $|M|$, целесообразно оценить число частей, получаемых на каждой итерации разложения, и определить правило остановки для реализации всего процесса разложения за полиномиальное время.

Пусть $g \in G$ и $m \in M$ — произвольные элементы контекста $K = (G, M, I)$. Пары множеств (g'', g') и (m', m'') образуют формальные понятия, первое из которых назовём объектным, а второе — признаковым формальным понятием контекста $K = (G, M, I)$.

Обозначим $O = \{(g'', g') : g \in G\} \subseteq FC$ множество всех объектных формальных понятий и $S = \{(m', m'') : m \in M\} \subseteq FC$ множество всех признаковых формальных понятий контекста $K = (G, M, I)$.

Утверждение 1. Всякое объектное формальное понятие (g'', g') формального контекста $K = (G, M, I)$ имеет самое большое по размеру содержание среди других формальных понятий, имеющих в объёме объект $g \in G$; признаковое формальное понятие (m', m'') обладает самым большим объёмом среди других формальных понятий, имеющих в содержании признак $m \in M$.

Доказательство. Справедливость утверждения 1 непосредственно следует из свойств оператора $(\cdot)''$ и определения формального понятия. ■

Пара формальных понятий $(g'', g') \in O$, $(m', m'') \in S$ определяет бокс $\omega = (m', g', J)$ как часть контекста $K = (G, M, I)$, если

$$(g'', g') \sqsubseteq (m', m'')\,, \tag{3}$$

что эквивалентно $g'' \subseteq m'$ (или $m'' \subseteq g'$). Про такой бокс будем говорить, что он образован элементами $g \in G$ и $m \in M$. Далее вместо $\omega = (m', g', J)$ будем кратко писать $\omega = (m', g')$ или (m', g').

Утверждение 2. Для всякого формального контекста $K = (G, M, I)$ и любых $(g'', g') \in O$, $(m', m'') \in S$ отношение порядка $(g'', g') \sqsubseteq (m', m'')$ выполняется тогда и только тогда, когда $(g, m) \in I$.

Доказательство. Пусть $(g'', g') \sqsubseteq (m', m'')$. Тогда $g'' \subseteq m'$, $m'' \subseteq g'$. Согласно рефлексивности оператора $(\cdot)''$, имеем $\{g\} \subseteq g'' \subseteq m'$, $\{m\} \subseteq m'' \subseteq g'$. Из (1) следует $(g, m) \in I$. Докажем обратное. Пусть $(g, m) \in I$. Это означает, что $\{g\} \subseteq m'$, $\{m\} \subseteq g'$. В силу монотонности оператора $(\cdot)''$ верны включения $g'' \subseteq (m')''$, $m'' \subseteq (g')''$. Отсюда в силу рефлексивности оператора $(\cdot)''$ и равенств (2) имеем $\{g\} \subseteq g'' \subseteq m'$, $\{m\} \subseteq$ $\subseteq m'' \subseteq g'$. Следовательно, $(g'', g') \sqsubseteq (m', m'')$. ■

Из утверждения 2 следует, что число различных боксов, порождаемых всевозможными элементами формального контекста $K = (G, M, I)$, не превышает веса 0,1-матрицы T, т. е. величины $\|T\|$ — числа единичных элементов этой матрицы. Очевидно, что $1 \leqslant \|T\| \leqslant |G| \cdot |M|$.

Будем говорить, что формальное понятие $(A, B) \in FC$ вложено в бокс (m', g') контекста $K = (G, M, I)$, и записывать $(A, B) \preceq (m', g')$, если $A \subseteq m'$, $B \subseteq g'$. Всякий бокс (m', g') не является пустым, поскольку, согласно (3), он всегда содержит формальные понятия $(g'', g') \in O$ и $(m', m'') \in S$.

Утверждение 3. Всякое нетривиальное формальное понятие (A, B) контекста $K = (G, M, I)$, которое вложено в бокс (m', g'), образованный элементами $g \in G$ и $m \in M$, всегда содержит эти элементы и их замыкания, т. е. если $(A, B) \preceq (m', g')$, то

1) $g \in A$ и $m \in B$;
2) $g'' \subseteq A$ и $m'' \subseteq B$.

Доказательство. Если $(A, B) \preceq (m', g')$, то $A \subseteq m'$, $B \subseteq g'$. В силу антимонотонности отображений $(\cdot)'$ верно $m'' \subseteq A'$, $g'' \subseteq B'$. Для формального понятия (A, B) по определению $A = B'$, $B = A'$. Тогда, $m'' \subseteq B, g'' \subseteq A$. В силу рефлексивности оператора $(\cdot)''$ имеем $\{m\} \subseteq B, \{g\} \subseteq A$. Отсюда следует справедливость обоих высказываний утверждения 3. ■

Согласно утверждению 3, пару (g'', m'') можно рассматривать в качестве типичного представителя не только бокса (m', g'), но и всех формальных понятий контекста $K = (G, M, I)$, вложенных в этот бокс. Это правомерно, поскольку подматрица, соответствующая боксу (m', g'), во всех строках из g'' и всех столбцах из m'' имеет единичные элементы. Соответствие между боксами и формальными понятиями контекста устанавливает следующая теорема.

Теорема 1. Для всякого формального контекста $K = (G, M, I)$, множества FC всех его формальных понятий и любой пары множеств (A, B), $\varnothing \neq A \subseteq G$, $\varnothing \neq B \subseteq M$, справедливы следующие высказывания:

1) если $(A, B) \in FC$, то всегда в $K = (G, M, I)$ существует бокс $\omega = (m', g')$, $g \in G$ и $m \in M$, возможно, не единственный, в который это формальное понятие вложено;
2) если (A, B) — формальное понятие некоторого бокса $\omega = (m', g')$ формального контекста $K = (G, M, I)$, то оно также принадлежит FC.

Доказательство. Пусть (A, B) — произвольное формальное понятие контекста $K = (G, M, I)$ и $\varnothing \neq A \subseteq G$, $\varnothing \neq B \subseteq M$. По определению для него верны равенства

$$(A, B) = (B', A') = (A'', B'')\,. \tag{4}$$

Рассмотрим некоторый объект $g \in A$ и найдём соответствующее ему объектное формальное понятие (g'', g'). Поскольку $\{g\} \subseteq A$, в силу антимонотонности отображений $(\cdot)'$, монотонности оператора $(\cdot)''$ и равенств (4) справедливы отношения

$$A' \subseteq g', \quad g'' \subseteq A'' = A. \tag{5}$$

Аналогично для произвольного признака $m \in B$ и признакового формального понятия (m', m'') верны отношения

$$B' \subseteq m', \quad m'' \subseteq B'' = B. \tag{6}$$

Из (4)–(6) вытекает, что $g'' \subseteq m'$ и $m'' \subseteq g'$. Следовательно, пара формальных понятий (g'', g') и (m', m'') определяет бокс $\omega = (m', g')$. Кроме того, $A = B' \subseteq m'$, $B = A' \subseteq$ $\subseteq g'$. Это означает, что формальное понятие (A, B) вложено в бокс $\omega = (m', g')$. Если выбрать другой объект из A и/или другой признак из B, то получим тот же самый бокс или, возможно, другой бокс, содержащий формальное понятие (A, B). Первое высказывание теоремы 1 доказано.

Докажем второе высказывание. Пусть (A, B) — формальное понятие некоторого бокса $\omega = (m', g')$ как части контекста $K = (G, M, I)$. Далее результаты отображений $(\cdot)'$, вычисленные для $\omega = (m', g')$, а не контекста $K = (G, M, I)$ в целом, будем

отмечать символом ω в нижнем индексе. В этих обозначениях имеем

$$A = B'_\omega \subseteq m', \quad B = A'_\omega \subseteq g'. \tag{7}$$

Отношения (7) отражают вложенность понятия (A, B) в бокс $\omega = (m', g')$. Если понятие (A, B) совпадает с объектным (g'', g') или признаковым формальным понятием (m', m''), по которым образован бокс $\omega = (m', g')$, то второе высказывание тривиальным образом выполняется.

Пусть формальное понятие (A, B) отлично от (g'', g') и (m', m'') и для него верны отношения (7). Требуется показать, что объём и содержание формального понятия (A, B) не могут выйти за границы бокса $\omega = (m', g')$ при вычислении результатов отображений $(\cdot)'$ применительно к контексту $K = (G, M, I)$, т. е. обязательно верны отношения

$$B'_\omega = B' \subseteq m', \quad A'_\omega = A' \subseteq g'. \tag{8}$$

Заметим, что по утверждению 3 всегда $g \in A$ и $m \in B$. Если предположить, что (8) не выполняются, например $m' \subset B'$, то это будет противоречить утверждению 1, согласно которому формальное понятие (m', m'') обладает самым большим объёмом среди других формальных понятий, имеющих в содержании признак $m \in M$. Справедливость (8) означает, что (A, B) является не только формальным понятием бокса $\omega = (m', g')$, но и формальным понятием исходного контекста $K = (G, M, I)$. ■

Согласно теореме 1, разложение контекста $K = (G, M, I)$ на боксы является «безопасным» для любого формального понятия из FC. В теореме 1 исключены случаи, когда FC содержит хотя бы одно из тривиальных формальных понятий $(G, \varnothing)$, $(\varnothing, M)$. Поскольку всегда верны отношения

$$(\varnothing, M) \sqsubseteq (G, \varnothing), \quad (\varnothing, M) \sqsubseteq (G, G'), \quad (M', M) \sqsubseteq (G, \varnothing),$$

контекст $K = (G, M, I)$ можно рассматривать как бокс (G, M). Следовательно, даже в этих исключительных случаях каждый бокс содержит по крайней мере одно формальное понятие из FC, при этом ни одно формальное понятие из FC не теряется.

Из теоремы 1 вытекает важное практическое следствие: искомое множество FC может быть восстановлено путём объединения множеств формальных понятий, выявленных в боксах контекста $K = (G, M, I)$.

Очевидно, что процесс разложения заданного контекста на боксы может быть организован итерационно, поскольку каждый выявленный на первой итерации бокс можно рассматривать в качестве исходного контекста и вновь подвергать декомпозиции. Оценку числа боксов, получаемых на каждой итерации разложения, устанавливает утверждение 2. Определим правила остановки итерационного процесса разложения. Для этого введём понятие плотности бокса.

Пусть $|m'| \cdot |g'|$ — размер бокса (m', g'), а $\|(m', g')\|$ — число его единичных элементов. Плотностью бокса (m', g') назовём величину

$$\sigma(m', g') = \frac{\|(m', g')\|}{|m'| \cdot |g'|}.$$

Верны естественные границы $0 < \sigma(m', g') \leqslant 1$.

Утверждение 4. Если бокс (m', g'), образованный элементами $g \in G$ и $m \in M$, имеет плотность $\sigma(m', g') = 1$, то $g'' = m'$, $m'' = g'$.

Доказательство. Поскольку $\sigma(m', g') = 1$, то $|m'| \cdot |g'| = \|(m', g')\|$. Это означает, что для любого объекта $g \in m'$ и для всякого признака $m \in g'$ верно $(g, m) \in I$. Отсюда $g'' = m'$, $m'' = g'$. ■

Утверждение 5. Всякий бокс (m', g') с плотностью $\sigma(m', g') = 1$ содержит ровно одно нетривиальное формальное понятие (A, B) контекста $K = (G, M, I)$, совпадающее с ним, т. е. $A = m'$ и $B = g'$.

Доказательство. Пусть $(A, B) \preceq (m', g')$, тогда $A \subseteq m'$, $B \subseteq g'$. Из утверждения 4 следует, что $A \subseteq m' = g''$ и $B \subseteq g' = m''$, а значит, $A \subseteq g''$ и $B \subseteq m''$. Между тем по утверждению 3 верны обратные включения $g'' \subseteq A$ и $m'' \subseteq B$. Следовательно, $A = m'$, $B = g'$. ■

Из утверждения 5 следует, что бокс (m', g') с плотностью 1 вырождается в нетривиальное формальное понятие и не подлежит дальнейшему разложению.

Заметим, что время построения одного бокса составляет $\mathrm{O}(|G| \cdot |M|)$. Согласно утверждению 2, число боксов, возникающих на каждой отдельной итерации процесса декомпозиции, сопоставимо с $\mathrm{O}(|G| \cdot |M|)$. Если ограничить число итераций некоторой константой, то процесс разложения исходного контекста на боксы можно осуществить за полиномиальное время. Дополнительно можно установить ограничение на плотность формируемых боксов.

На практике число боксов, возникающих на каждой отдельной итерации процесса декомпозиции, в ряде случаев может быть уменьшено за счёт удаления вложенных и кратных боксов. Рассмотрим для контекста $K = (G, M, I)$ множество боксов

$$\Omega = \left\{\omega_1, \omega_2, \ldots, \omega_{\|T\|}\right\},$$

где $\omega_i = (m_i', g_i')$, $i = 1, 2, \ldots, \|T\|$. Будем говорить, что бокс $\omega_1 = (m_1', g_1')$ вложен в бокс $\omega_2 = (m_2', g_2')$, и писать $\omega_1 \preceq \omega_2$, если верны теоретико-множественные включения

$$m_1' \subseteq m_2', \quad g_1' \subseteq g_2'.$$

При $m_1' = m_2'$ и $g_1' = g_2'$ боксы ω_1 и ω_2 назовём кратными. Будем считать, что боксы ω_1 и ω_2 сравнимы между собой, если $\omega_1 \preceq \omega_2$ или $\omega_2 \preceq \omega_1$, иначе несравнимы. Таким образом, множество Ω частично упорядочено относительно введённого отношения порядка. С учётом теоремы 1 справедливо

Следствие 1. Для любых $\omega_1, \omega_2 \in \Omega$, таких, что $\omega_1 \preceq \omega_2$, все формальные понятия бокса ω_1 также являются формальными понятиями бокса ω_2 и контекста $K = (G, M, I)$.

Известно, что в частично упорядоченном множестве всегда можно найти взаимно непересекающиеся цепи [20]. Непустое подмножество $\{\omega_{i1}, \omega_{i2}, \ldots, \omega_{il}\}$ множества Ω является цепью, если все элементы этого подмножества попарно сравнимы между собой и линейно упорядочены: $\omega_{i1} \preceq \omega_{i2} \preceq \ldots \preceq \omega_{il}$. Элемент ω_{il} называется максимальным элементом, а величина l — длиной этой цепи. Цепь называется максимальной, если её объединение с любым не принадлежащим ей элементом цепью не является. Две цепи называются взаимно непересекающимися, если они не содержат общих элементов. Число максимальных взаимно непересекающихся цепей и длина самой длинной такой цепи определяются теоремой Дилоурса [20]. Существует алгоритм построения взаимно непересекающихся цепей частично упорядоченного множества, основанный на вычислении максимального паросочетания двудольного графа. В работе [20] доказано, что

время выполнения данного алгоритма полиномиально относительно мощности исходного частично упорядоченного множества и что построенные цепи максимальные и взаимно непересекающиеся.

Согласно следствию 1, максимальный элемент всякой цепи сохраняет все формальные понятия остальных элементов этой цепи. Данные элементы могут быть удалены и тем самым уменьшено число боксов, получаемых на каждой отдельной итерации разложения. Существуют случаи, когда указанный приём не даёт эффекта, например, когда все элементы множества Ω несравнимы между собой или когда множество Ω линейно упорядочено. Однако эти случаи крайне редки для реальных контекстов.

4. Вычислительные эксперименты

Для оценки результативности предложенного метода декомпозиции формального контекста были выполнены вычислительные эксперименты. Эксперименты проводились с помощью программы FCACorpus, базирующейся на алгоритме Close-by-One нахождения всех формальных понятий [21]. Использовались формальные контексты, описывающие коллекции тувинских текстов. Для каждого контекста $K = (G, M, I)$ осуществлялось нахождение множества FC без разложения и с разложением на боксы. Результаты приведены в таблице, где $|G|$ — количество объектов; $|M|$ — количество признаков исходного контекста $K = (G, M, I)$; $\|T\|$ — вес матрицы, соответствующей этому контексту; $|FC|$ — число найденных формальных понятий; N — количество образованных боксов: t — время выполнения программы. Эксперименты выполнялись на компьютере с процессором Intel Core i7-720QM Processor (6M Cache, 1.60 GHz) и ОЗУ размером 4 ГБ.

Результаты экспериментов

Случаи	$\|G\|$	$\|M\|$	$\|\|T\|\|$	$\|FC\|$	N	t, мс
Без разложения на боксы	100	10	480	592	–	788
С разложением на боксы				592	470	420
Без разложения на боксы	500	20	5100	5780	–	79449
С разложением на боксы				5780	5080	13253
Без разложения на боксы	1000	30	14700	14506	–	394520
С разложением на боксы				14506	13876	180144

Как видно из таблицы, значения $|FC|$ во всех случаях (без разложения и с разложением на боксы) полностью совпадают. Это подтверждает «безопасность» разложения контекста на боксы относительно формальных понятий. Число N боксов, образованных при разложении контекста, не превышает величины $\|T\|$, что свидетельствует о правильности утверждения 2. Эксперименты показывают, что применение предложенного метода декомпозиции даёт значительный выигрыш по времени: время выполнения программы FCACorpus при разложении контекста на боксы уменьшается в несколько раз.

Заключение

Представленный метод декомпозиции позволяет повысить производительность алгоритмов решения задачи нахождения всех формальных понятий и применять их для предметных областей, описываемых контекстами большой размерности. Данный метод применим также для теоретико-графовых задач, связанных с нахождением биклик в двудольном графе. Возможны другие методы разложения формального контекста на части, однако они неизменно должны быть «безопасными» относительно формальных понятий.

ЛИТЕРАТУРА

1. *Биркгоф Г.* Теория решеток. М.: Наука, 1984. 568 с.
2. *Ganter B. and Wille R.* Formal Concept Analyses: Mathematical Foundations. Springer Science and Business Media, 2012. 314 p.
3. *Ganter B. and Obiedkov S. A.* Conceptual Exploration. Berlin, Heidelberg: Springer, 2016. 315 p.
4. *Kuznetsov S. O. and Obiedkov S. A.* Comparing Performance of Algorithms for Generating Concept Lattices // J. Experimental and Theoretical Artificial Intelligence. 2002. V. 14. No. 2. P. 189–216.
5. *Simon A. A.* Best-of-Breed approach for designing a fast algorithm for computing fixpoints of Galois Connections // Inform. Sci. 2015. V. 295. No. 2. P. 633–649.
6. *Aslanyan L., Alipour D., and Heidari M.* Comparative analysis of attack graphs // Mathem. Problems of Computer Sci. 2013. No. 40. P. 85–95.
7. *Heydari M., Morales L., Shields C. O., and Sudborough I. H.* Computing Cross Associations for Attack Graphs and other Applications // Proc. 40th Ann. Hawaii Intern. Conf. on System Sciences. Big Island, Hawaii, 2007. P. 270.
8. *Li J., Liu G., Li H., and Wong L.* Maximal biclique subgraphs and closed pattern pairs of the adjacency matrix: A one-to-one correspondence and mining algorithms // J. IEEE Trans. Knowledge and Data Engineering. 2007. No. 19. P. 1625–1637.
9. *Bein D., Morales L., Bein W., et al.* Clustering and the biclique partition problem // Proc. 41st Ann. Hawaii Intern. Conf. on System Sciences. Big Island, Hawaii, 2008. P. 475–483.
10. *Гэри М., Джонсон Д.* Вычислительные машины и труднорешаемые задачи. М.: Мир, 1982. 416 с.
11. *Дугинов О. И.* Сложность задач покрытия графа наименьшим числом полных двудольных графов // Труды Института математики. 2014. Т. 22. Вып. 1. С. 51–69.
12. *Prisner E.* Bicliques in graphs I: bounds on their number // Combinatorica. 2000. V. 20. P. 109–117.
13. *Wood D. R.* On the maximum number of cliques in a graph // Graphs and Combinatorics. 2007. No. 23. P. 1–16.
14. *Moon J. W. and Moser L.* On cliques in graphs // J. Mathematics. 1965. No. 3. P. 23–28.
15. *Pottosina S., Pottosin Y., and Sedliak B.* Finding maximal complete bipartite subgraphs in a graph // J. Appl. Math. 2008. V. 1. No. 1. P. 75–81.
16. *Vania M. F. Dias, Celina M. H. de Figueiredo, and Jayme L. S.* Generating bicliques of a graph in lexicographic order // Theor. Comput. Sci. 2005. No. 337. P. 240–248.
17. *Дюкова Е. В., Журавлев Ю. И.* Дискретный анализ признаковых описаний в задачах распознавания большой размерности // Журн. вычислит. матем. и матем. физики. 2000. Т. 40. № 8. С.1264–1278.
18. *Дюкова Е. В., Инякин А. С.* О процедурах классификации, основанных на построении покрытий классов // Журн. вычислит. матем. и матем. физики. 2003. Т. 43. № 12. С.1884–1895.
19. *Быкова В. В.* Математические методы анализа рекурсивных алгоритмов // Журн. Сибирского федерального университета. Математика и физика. 2008. Т. 3. № 1. С. 372–384.
20. *Harzheim E.* Ordered Sets. New York: Springer, 2005. 390 p.
21. *Монгуш Ч. М., Быкова В. В.* Программа FCACorpus концептуального моделирования тувинских текстов методами анализа формальных понятий. Свидетельство о государственной регистрации программы для ЭВМ № 2018618907, выдано Федеральной службой по интеллектуальной собственности РФ, 2018.

REFERENCES

1. *Birkhoff G.* Lattice Theory. AMS, Providence, 1967. 423 p.
2. *Ganter B. and Wille R.* Formal Concept Analyses: Mathematical Foundations. Springer Science and Business Media, 2012. 314 p.
3. *Ganter B. and Obiedkov S. A.* Conceptual Exploration. Berlin, Heidelberg, Springer, 2016. 315 p.
4. *Kuznetsov S. O. and Obiedkov S. A.* Comparing performance of algorithms for generating concept lattices. J. Experimental and Theoretical Artificial Intelligence, 2002, vol. 14, no. 2, pp. 189–216.
5. *Simon A. A.* Best-of-Breed approach for designing a fast algorithm for computing fixpoints of Galois Connections. Information Sci., 2015, vol. 295, no. 2, pp. 633–649.
6. *Aslanyan L., Alipour D., and Heidari M.* Comparative analysis of attack graphs. Mathem. Problems of Computer Sci., 2013, no. 40, pp. 85–95.
7. *Heydari M., Morales L., Shields C. O., and Sudborough I. H.* Computing cross associations for attack graphs and other applications. Proc. 40th Ann. Hawaii Intern. Conf. on System Sciences, Big Island, Hawaii, 2007, pp. 270.
8. *Li J., Liu G., Li H., and Wong L.* Maximal biclique subgraphs and closed pattern pairs of the adjacency matrix: A one-to-one correspondence and mining algorithms. J. IEEE Trans. Knowledge and Data Engineering, 2007, no. 19, pp. 1625–1637.
9. *Bein D., Morales L., Bein W., et al.* Clustering and the biclique partition problem. Proc. 41st Ann. Hawaii Intern. Conf. on System Sciences, Big Island, Hawaii, 2008, pp. 475–483.
10. *Garey M. and Johnson D.* Computers and Intractability. N.Y., Freeman and Co, 1979. 340 p.
11. *Duginov O. I.* Slozhnost' zadach pokrytiya grafa naimen'shim chislom polnykh dvudol'nykh grafov [The complexity of the problems of covering a graph with the smallest number of complete bipartite graphs]. Proc. Institute of Mathematics, 2014, vol. 22, iss. 1, pp. 51–69. (in Russian)
12. *Prisner E.* Bicliques in graphs I: bounds on their number. Combinatorica, 2000, vol. 20, pp. 109–117.
13. *Wood D. R.* On the maximum number of cliques in a graph. Graphs and Combinatorics, 2007, no. 23, pp. 1–16.
14. *Moon J. W. and Moser L.* On cliques in graphs. J. Mathematics, 1965, no. 3, pp. 23–28.
15. *Pottosina S., Pottosin Y., and Sedliak B.* Finding maximal complete bipartite subgraphs in a graph. J. Appl. Math., 2008, vol. 1, no. 1, pp. 75–81.
16. *Vania M. F. Dias, Celina M. H. de Figueiredo, and Jayme L. S.* Generating bicliques of a graph in lexicographic order. Theoretical Computer Science, 2005, no. 337, pp. 240–248.
17. *Dyukova E. V. and Zhuravlev Yu. I.* Diskretnyy analiz priznakovykh opisaniy v zadachakh ras-poznavaniya bol'shoy razmernosti [Discrete analysis of feature descriptions in high-dimensional recognition tasks]. J. Comput. Mathem. and Mathem. Physics, 2000, vol. 40, no. 8, pp. 1264–1278. (in Russian)
18. *Dyukova E. V. and Inyakin A. S.* O protsedurakh klassifikatsii, osnovannykh na postroyenii pokrytiy klassov [About classification procedures based on classroom construction]. J. Comput. Mathem. and Mathem. Physics, 2003, vol. 43, no. 12, pp.1884–1895. (in Russian)
19. *Bykova V. V.* Matematicheskiye metody analiza rekursivnykh algoritmov [Mathematical methods for analyzing recursive algorithms]. J. Siberian Federal University. Mathematics and Physics, 2008, vol. 3, no. 1, pp. 372–384. (in Russian)
20. *Harzheim E.* Ordered Sets. New York, Springer, 2005. 390 p.
21. *Mongush Ch. M. and Bykova V. V.* Programma FCACorpus kontseptual'nogo modelirovaniya tuvinskikh tekstov metodami analiza formal'nykh ponyatiy [The program FCACorpus of

conceptual modeling of Tuvan texts by the methods of the formal concepts analysis]. Certificate of state registration of computer programs no. 2018618907, issued by the Federal Service for Intellectual Property of the Russian Federation, 2018. (in Russian)

СВЕДЕНИЯ ОБ АВТОРАХ

АГИБАЛОВ Геннадий Петрович — доктор технических наук, профессор, главный научный сотрудник лаборатории компьютерной криптографии Национального исследовательского Томского государственного университета, г. Томск.
E-mail: **agibalov@mail.tsu.ru**

АЛЕХИНА Марина Анатольевна — доктор физико-математических наук, профессор, заведующая кафедрой математики и физики Пензенского государственного технологического университета, г. Пенза. E-mail: **alekhina.marina19@yandex.ru, alekhina@penzgtu.ru**

БАРСУКОВА Оксана Юрьевна — кандидат физико-математических наук, доцент кафедры компьютерных технологий Пензенского государственного университета, г. Пенза. E-mail: **oksana.barsukova.71@gmail.com**

БИЯШЕВ Рустем Гакашевич — доктор технических наук, профессов, заведующий лабораторией информационной безопасности Института информационных и вычислительных технологий МОН РК, г. Алматы. E-mail: **brg@ipic.kz**

БЫЗОВ Виктор Александрович — старший преподаватель кафедры прикладной математики и информатики Вятского государственного университета, г. Киров.
E-mail: **vbyzov@yandex.ru**

БЫКОВА Валентина Владимировна — доктор физико-математических наук, профессор Института математики и фундаментальной информатики Сибирского федерального университета, г. Красноярск. E-mail: **bykvalen@mail.ru**

ЗАГУМЕННОВ Денис Владимирович — аспирант Южного федерального университета, г. Ростов-на-Дону. E-mail: **zagumionnov.denis@yandex.ru**

КАЛИМОЛДАЕВ Максат Нурадилович — доктор физико-математических наук, акад. НАН РК, генеральный директор Института информационных и вычислительных технологий МОН РК, г. Алматы. E-mail: **mnk@ipic.kz**

ЛИТИЧЕВСКИЙ Дмитрий Владимирович — аспирант Челябинского государственного университета, г. Челябинск. E-mail: **litichevskiydv@gmail.com**

МКРТИЧЯН Вячеслав Витальевич — старший научный сотрудник ФГАНУ НИИ «Спецвузавтоматика», г. Ростов-на-Дону. E-mail: **mkrtichan@list.ru**

МОНГУШ Чодураа Михайловна — старший преподаватель Тувинского государственного университета, г. Кызыл. E-mail: **mongushchod91@yandex.ru**

ПАНКРАТОВА Ирина Анатольевна — кандидат физико-математических наук, доцент, заведующая лабораторией компьютерной криптографии Национального исследовательского Томского государственного университета, г. Томск.
E-mail: **pank@mail.tsu.ru**

ПУШКАРЕВ Игорь Александрович — кандидат физико-математических наук, доцент, доцент кафедры прикладной математики и информатики Вятского государственного университета, г. Киров. E-mail: **god_sha@mail.ru**

РОГ Ольга Алексеевна — научный сотрудник Института информационных и вычислительных технологий МОН РК, г. Алматы. E-mail: **olga@ipic.kz**

РЫБАЛОВ Александр Николаевич — кандидат физико-математических наук, старший научный сотрудник лаборатории комбинаторных и вычислительных методов алгебры и логики Института математики им. С. Л. Соболева СО РАН, г. Омск.
E-mail: **alexander.rybalov@gmail.com**