МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

ПРИКЛАДНАЯ ДИСКРЕТНАЯ МАТЕМАТИКА

Научный журнал

2018 № 40

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 14.06.2018. Формат $60 \times 84\frac{1}{8}$. Усл. п. л. 14,8. Тираж 300 экз.
Заказ № 3239. Цена свободная. Дата выхода в свет 29.06.2018.

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 512.13

КЛАСС СБАЛАНСИРОВАННЫХ АЛГЕБРАИЧЕСКИХ ПОРОГОВЫХ ФУНКЦИЙ

Д. А. Сошин

ФГУП «НИИ «Квант», г. Москва, Россия

Предложен подход к построению класса сбалансированных алгебраических пороговых функций (АПФ). Функция k-значной логики f называется АПФ, если существуют целочисленные наборы $\mathbf{c} = (c_0, c_1, \ldots, c_n)$, $\mathbf{b} = (b_0, b_1, \ldots, b_k)$ и натуральный модуль m, такие, что $f(x_1, x_2, \ldots, x_n) = \alpha$, если и только если $b_\alpha \leqslant r_m(c_0 + c_1x_1 + c_2x_2 + \cdots + c_nx_n) < b_{\alpha+1}$ для любого $\alpha \in \Omega_k$, где $r_m(x)$ — функция приведения числа x по модулю m. Тройку $(\mathbf{c}; \mathbf{b}; m)$ будем называть структурой функции f. Центральным результатом работы является построенный класс сбалансированных АПФ, а именно: если для АПФ f, заданной структурой $((c_0, c_1, c_2, \ldots, c_n);\ (0, p, 2p, \ldots, kp);\ kp) = (\mathbf{c}, \mathbf{b}, m)$, существует $c_i = pq$ и $(q, k) = 1$, то такая функция сбалансированная. Сбалансированные функции данного класса могут быть использованы в качестве координатных функций подстановок.

Ключевые слова: *алгебраические пороговые функции, сбалансированные функции.*

DOI 10.17223/20710410/40/1

THE CLASS OF BALANCED ALGEBRAIC THRESHOLD FUNCTIONS

D. A. Soshin

Technology Federal State Unitary Enterprise "Research Institute Kvant", Moscow, Russia

E-mail: danil_re@list.ru

The paper proposes an approach to the construction of a class of balanced algebraic threshold functions (ATF). The function f of k-valued logic is called ATF if there are sequences $\mathbf{c} = (c_0, c_1, \ldots, c_n)$, $\mathbf{b} = (b_0, b_1, \ldots, b_k)$ of integers and the natural modulus m such that $f(x_1, x_2, \ldots, x_n) = \alpha \quad \Leftrightarrow \quad b_\alpha \leqslant (c_0 + c_1x_1 + c_2x_2 + \ldots + c_nx_n) \bmod m < b_{\alpha+1}$ for any $\alpha \in \Omega_k = \{0, 1, \ldots, k-1\}$. The triple $(\mathbf{c}; \mathbf{b}; m)$ is called the structure of the function f. The central result of the paper is a class of balanced ATF constructed in the following way: if an ATF f has a structure $(\mathbf{c}, \mathbf{b}, m) = ((c_0, c_1, c_2, \ldots, c_n);\ (0, p, 2p, \ldots, kp);\ kp)$ where $c_i = pq$ and $(q, k) = 1$, then this function is balanced. Such functions can be used as coordinate functions of substitutions.

Keywords: *algebraic threshold functions, balanced functions.*

Работа посвящена развитию методов синтеза биективных преобразований, задаваемых регулярными системами функций. В [2, 1] в качестве координатных функций подстановок предлагается использовать системы однотипных пороговых функций и системы алгебраических пороговых функций. Согласно критерию Хаффмана, координатные функции должны быть сбалансированными, поэтому на первом этапе должен быть решён вопрос описания классов сбалансированных функций. В работе построен класс сбаланированных АПФ, обобщающий результаты, полученные в [3]. Использование АПФ представляет практический интерес в связи с возможностью их эффективной реализации непосредственно в среде — носителе сигнала, а именно в оптической среде.

Алгебраические пороговые функции k-значной логики от n переменных AT_n^k расширяют класс пороговых функций T_n^k за счёт добавления модульной операции.

Определение 1. Функцию k-значной логики $f : \Omega_k^n \to \Omega_k$ назовём алгебраической пороговой, если существуют целочисленные наборы $\mathbf{c} = (c_0, c_1, \ldots, c_n)$, $\mathbf{b} = (b_0, b_1, \ldots, b_k)$ и натуральный модуль m, такие, что для любого $\alpha \in \Omega_k$ выполняется

$$f(x_1, x_2, \ldots, x_n) = \alpha \quad \Leftrightarrow \quad b_\alpha \leqslant r_m(c_0 + c_1x_1 + c_2x_2 + \cdots + c_nx_n) < b_{\alpha+1}, \tag{1}$$

где $r_m(x)$ — функция приведения числа x по модулю m, $r_m(x) \in \{0, 1, \ldots, m-1\}$; $\Omega_k = \{0, 1, \ldots, k-1\}$. Тройку $(\mathbf{c}, \mathbf{b}, m)$ будем называть структурой функции f.

Двойное неравенство (1) можно записать равносильным способом

$$b_\alpha \leqslant c_1x_1 + c_2x_2 + \cdots + c_nx_n + mt < b_{\alpha+1} \tag{2}$$

для некоторого $t \in \mathbb{Z}$ [4].

Определение 2. Слоем $D_\alpha(f)$ (носителем значения α) пороговой функции f будем называть множество точек из Ω_k^n, в которых функция f принимает значение α, $\alpha \in \{0, \ldots, k-1\}$.

Пример 1. Зададим 3-значную АПФ $f(x_1, x_2)$ её структурой

$$(\mathbf{c}; \mathbf{b}; m) = ((0, 2, 3); (0, 3, 6, 9); 9)\,.$$

Выпишем неравенства, определяющие значения функции:

$$\begin{aligned} f(x_1, x_2) = 0 \quad &\Leftrightarrow \quad 0 \leqslant r_9(2x_1 + 3x_2) < 3; \\ f(x_1, x_2) = 1 \quad &\Leftrightarrow \quad 3 \leqslant r_9(2x_1 + 3x_2) < 6; \\ f(x_1, x_2) = 2 \quad &\Leftrightarrow \quad 6 \leqslant r_9(2x_1 + 3x_2) < 9. \end{aligned}$$

Представим функцию $f(x_1, x_2)$ в табличном виде:

x_2 \ x_1	0	1	2
0	0	0	1
1	1	1	2
2	2	2	0

Функция $f(x_1, x_2)$ принадлежит классу AT_2^3, но не классу T_2^3. Действительно, любая пороговая функция является полностью монотонной [5], а при фиксациях $x_1 = 1$ и $x_1 = 2$ у функции f нарушается условие 1-монотонности.

Следующие два утверждения описывают подкласс BAT_n^k класса AT_n^k сбалансированных k-значных функций, представляющих практический интерес.

Теорема 1. Пусть $c_i \neq 0$, $i \in \{1, \ldots, n\}$, тогда структура

$$((c_0, c_1, c_2, \ldots, c_n); (0, c_i, 2c_i, \ldots, kc_i); kc_i) \tag{3}$$

задаёт сбалансированную функцию, принадлежащую классу AT_n^k.

Доказательство. Рассмотрим функцию $f(x_1, x_2, \ldots, x_n)$, заданную структурой (3). Для множества Ω_k^n справедливо разложение

$$\Omega_k^n = \bigsqcup_{\alpha=0}^{k-1} D_\alpha(f), \quad D_i(f) \cap D_j(f) = \varnothing \text{ при } i \neq j.$$

Для доказательства теоремы достаточно проверить равномощность всех $D_\alpha(f)$. Зафиксируем произвольное значение $d \in \{0, \ldots, k-2\}$. Построим отображение

$$\varphi : D_d(f) \to D_{d+1}(f)$$

по следующему правилу. Для каждой точки $\mathbf{a} = (a_1, a_2, \ldots, a_n) \in D_d(f)$

$$\varphi(a_1, \ldots, a_{i-1}, a_i, a_{i+1}, \ldots, a_n) = (a_1, \ldots, a_{i-1}, a_i \oplus 1, a_{i+1}, \ldots, a_n), \tag{4}$$

где $\oplus$ — сложение по модулю k. Покажем корректность задания отображения φ. Запишем сравнения (2) для функции $f(x_1, x_2, \ldots, x_n)$, используя структуру (3):

$$\alpha c_i \leqslant c_0 + c_1 x_1 + c_2 x_2 + \ldots + c_n x_n + kc_i t < (\alpha + 1)c_i, \tag{5}$$

$kc_i = m$, $\alpha c_i = b_\alpha$, $(\alpha + 1)c_i = b_{\alpha+1}$. Для точки $\mathbf{a} \in D_d(f)$ преобразуем сравнение (5), подставив вместо значения α значение d:

$$dc_i \leqslant c_0 + c_1 a_1 + c_2 a_2 + \ldots + c_n a_n + kc_i t < (d+1)c_i.$$

Ко всем частям двойного неравенства прибавим c_i:

$$(c_i + dc_i) \leqslant c_0 + c_1 a_1 + \ldots + (c_i a_i + c_i) + \ldots + c_n a_n + kc_i t < c_i + (d+1)c_i.$$

Преобразовав последнее, получим

$$(d+1)c_i \leqslant c_0 + c_1 a_1 + \ldots + c_i(a_i + 1) + \ldots + c_n a_n + kc_i t < (d+2)c_i. \tag{6}$$

Сравнение (6) задаёт пределы, в которых лежит значение линейной формы функции $f(x_1, x_2, \ldots, x_n)$ в точках $(a_1, a_2, \ldots, a_{i-1}, a_i + 1, a_{i+1}, \ldots, a_n)$, $a_i \in \{0, \ldots, k-2\}$. Если координата a_i равна $k-1$, то (6) примет вид

$$(d+1)c_i \leqslant c_0 + c_1 a_1 + \ldots + c_i 0 + \cdots + c_n a_n + kc_i t' < (d+2)c_i, \tag{7}$$

где $t' = t + 1 \in \mathbb{Z}$. Объединяя (6) и (7), получим для любого $a_i \in \{0, \ldots, k-1\}$

$$(d+1)c_i \leqslant r_{kc_i}(c_0 + c_1 a_1 + \ldots + c_i(a_i \oplus 1) + \ldots + c_n a_n) < (d+2)c_i.$$

Из последнего видно, что для указанных значений d выполняется

$$f\left(\varphi(a_1, a_2, \ldots, a_n)\right) = d + 1,$$

откуда получаем, что вектор $\varphi(a_1, a_2, \ldots, a_n)$ лежит в множестве $D_{d+1}(f)$.

Докажем, что отображение φ биективное. Для этого построим обратное отображение $\varphi^{-1} : D_{d+1}(f) \to D_d(f)$, положив для каждой точки $\mathbf{a} = (a_1, a_2, \ldots, a_n)$, принадлежащей множеству $D_{d+1}(f)$,

$$\varphi(a_1, a_2, \ldots, a_n) = (a_1, a_2, \ldots, a_{i-1}, a_i \ominus 1, a_{i+1}, \ldots, a_n), \tag{8}$$

где $\ominus$ — вычитание по модулю k. Корректность определения φ^{-1} проверяется аналогично φ. Из (4) и (8) получим $\varphi^{-1}\varphi(a_1, a_2, \ldots, a_n) = (a_1, a_2, \ldots, a_n)$, что свидетельствует о биективности φ и равномощности множеств $D_\alpha(f)$, $\alpha \in \{0, \ldots, k-1\}$. ■

Обобщим результат, полученный в теореме 1. В доказательстве используется следствие из критерия о полноцикловости линейного конгруэнтного генератора. Напомним, что под последовательностью, порождённой линейным конгруэнтным генератором с параметрами X_0, a, q, m, понимается последовательность $\{X_i\}_{i \geqslant 0}$, удовлетворяющая следующему рекуррентному закону:

$$X_{i+1} = aX_i + q \pmod m, \qquad i \geqslant 0. \tag{9}$$

Критерий полноцикловости [6, с. 36] при $a = 1$ имеет следующий вид:

Линейный конгруэнтный генератор вида (9) при $a = 1$ является полноцикловым тогда и только тогда, когда $(q, m) = 1$.

Теорема 2. Если для АПФ f, заданной структурой

$$((c_0, c_1, c_2, \ldots, c_n);\ (0, p, 2p, \ldots, kp);\ kp)\,,$$

существует $c_i = pq$ и $(q, k) = 1$, то функция f сбалансированная.

Доказательство. Доказательство полностью повторяет логику доказательства теоремы 1 со следующими изменениями.

Отображение φ для любого d определяется условием

$$\varphi : D_d(f) \to D_{d \oplus q}(f).$$

Правило (4) при этом остаётся неизменным. Далее, в силу равенств $c_i = pq$ и $b_d = dp$, в (6) произведём соответствующие замены, в результате чего получим

$$(d+q)p \leqslant c_0 + c_1a_1 + \ldots + c_i(a_i + 1) + \ldots + c_na_n + kpt < (d+q+1)p. \tag{10}$$

Остальные преобразования (10) повторяют логику доказательства теоремы 1 с учётом приведения, при необходимости, порогов в (10) по модулю kp.

Заметим, что преобразование Φ, действующее на множестве $\{D_i(f) : i{=}0, \ldots, k-1\}$ и заданное по правилу

$$\Phi(D_i(f)) = D_{i \oplus q}(f),$$

удовлетворяет условиям критерия полноцикловости. Из биективности φ и того, что все множества $D_i(f)$ лежат на одном цикле преобразования Φ, следует их равномощность, при этом преобразование Φ можно записать $\Phi(D_i(f)) = \varphi(D_i(f))$. ■

При $q = 1$ утверждение теоремы 2 совпадает с теоремой 1.

Замечание 1. В соответствии с результатами [2] были найдены структуры 18 представителей сбалансированных геометрических типов булевых функций от четырёх переменных с номерами

8.1.1.1, 8.1.2.1, 8.1.5.1, 8.1.16.1, 8.1.29.1, 8.2.1.1,
8.2.9.2, 8.2.11.1, 8.2.13.1, 8.2.17.1, 8.2.20.1, 8.2.21.1,
8.3.4.1, 8.3.5.1, 8.4.2.1, 8.4.3.1, 8.5.1.1, 8.8.1.1.

Непосредственной проверкой условий теоремы 2 установлено, что геометрические типы с номерами

8.1.1.1, 8.2.1.1, 8.2.13.1, 8.2.17.1, 8.2.20.1, 8.2.21.1,
8.3.4.1, 8.3.5.1, 8.4.2.1, 8.4.3.1, 8.5.1.1, 8.8.1.1

принадлежат классу BAT_4^2.

ЛИТЕРАТУРА

1. *Сошин Д. А.* Построение подстановок на основе пороговых функций многозначной логики // Прикладная дискретная математика. 2016. № 2(32). С. 20–32.
2. *Сошин Д. А.* Задание подстановок алгоритмов блочного шифрования Магма и 2-ГОСТ с помощью алгебраических пороговых функций // Прикладная дискретная математика 2016. № 3(33). С. 53–66.
3. *Сошин Д. А.* Конструктивный метод синтеза сбалансированных k-значных алгебраических пороговых функций // Comp. Nanotechnol. 2015. No. 4. P. 31–36.
4. *Глухов М. М., Елизаров В. П., Нечаев А. А.* Алгебра. СПб.: Лань, 2015. 608 с.
5. *Никонов В. Г., Никонов Н. В.* Особенности пороговых представлений k-значных функций // Труды по дискретной математике. М.: Физматлит, 2008. Т. 11. № 1. С. 60–85.
6. *Кнут Д. Э.* Искусство программирования. Т. 2. Получисленные алгоритмы. 3-е изд. М.: Вильямс, 2007. 832 с.

REFERENCES

1. *Soshin D. A.* Postroenie podstanovok na osnove porogovykh funktsiy mnogoznachnoy logiki [Constructing substitutions on the basis of threshold functions of multivalued logic]. Prikladnaya Diskretnaya Matematika, 2016, no. 2(32), pp. 20–32. (in Russian)
2. *Soshin D. A.* Zadanie podstanovok algoritmov blochnogo shifrovania Magma i 2-GOST s pomochiu algebraicheskikh porogovykh funktsiy [The implementation of Magma and 2-GOST block cipher substitutions by algebraic threshold functions]. Prikladnaya Diskretnaya Matematika, 2016, no. 3(33), pp. 53–66.(in Russian)
3. *Soshin D. A.* Konstruktivnyy metod sinteza sbalansirovannykh k-znachnykh algebraicheskikh porogovykh funktsiy [The constructive method for synthesis of balanced k-valued algebraic threshold functions]. Comp. Nanotechnol., 2015, no. 4, pp. 31–36. (in Russian)
4. *Glukhov M. M., Elizarov V. P., and Nechaev A. A.* Algebra [Algebra]. St. Petersburg, Lan' Publ., 2015. 608 p. (in Russian)
5. *Nikonov V. G. and Nikonov N. V.* Osobennosti porogovykh predstavleniy k-znachnykh functions [Peculiarities of threshold representations of k-valued functions]. Proc. Discr. Math., Moscow, Fizmatlit Publ., 2008, vol. 11, no. 1, pp. 60–85. (in Russian)
6. *Knuth D. E.* The Art of Computer Programming. Vol. 2. Seminumerical Algorithms, 3rd Ed. Massachusetts, Addison-Wesley, 1997. 762 p.

УДК 519.719.325

О ЛИНЕЙНОЙ РАЗЛОЖИМОСТИ ДВОИЧНЫХ ФУНКЦИЙ

А. В. Черемушкин

ФГУП «НИИ «Квант», г. Москва, Россия

Рассматривается множество возможных разложений двоичной функции в сумму (произведение) функций от непересекающихся множеств переменных при различных линейных преобразованиях аргументов, полученных отбрасыванием одночленов малой степени в их многочленах Жегалкина. Каждому такому разложению соответствует разложение векторного пространства в прямую сумму подпространств. Приведены условия, при которых такое разложение определяется однозначно с точностью до перестановки слагаемых (сомножителей) и связанных с ними подпространств между собой.

Ключевые слова: *двоичные функции, бесповторная декомпозиция, разложение в прямую сумму, линейное преобразование.*

DOI 10.17223/20710410/40/2

LINEAR DECOMPOSITION OF BOOLEAN FUNCTIONS INTO A SUM OR A PRODUCT OF COMPONENTS

A. V. Cheremushkin

Technology Federal State Unitary Enterprise "Research Institute Kvant", Moscow, Russia

E-mail: avc238@mail.ru

Let $f : \mathrm{GF}(2)^n \to \mathrm{GF}(2)$ be a Boolean function, $n \geqslant 2$, and $\mathcal{U}_s$ be a set of Boolean functions f of degree $\deg f \leqslant s$. Here is a consideration of the disjunctive decomposition of f into sum and products modulo $\mathcal{U}_s$ of Boolean functions after a linear substitution on arguments. The main result is the following: if all arguments of the functions $f(xA)$ under linear substitutions A of the vector space $\mathrm{GF}(2)^n$ are essential modulo $\mathcal{U}_s$ and f may be represented as disjunctive sum $f \equiv f_1 \oplus \ldots \oplus f_m \pmod{\mathcal{U}_s}$, where m is maximal, then subsequent direct sum of subspaces $\mathrm{GF}(2)^n = V^{(1)} + \ldots + V^{(m)}$ is unique and invariant under stabilizer group of the function f in general linear group. The article contains analogous result describing sufficient uniqueness condition for disjunctive products $f \equiv f_1 \ldots f_m \pmod{\mathcal{U}_s}$, namely, every function f_i has no affine multipliers and the set $\{a \in V_i : f_i(x \oplus a) \oplus f_i(x)$ has affine multipliers$\}$ generates the whole subspace V_i, $i = 1, \ldots, m$. For instance, this class of functions contains a nondegenerated quadratic forms.

Keywords: *Boolean functions, disjunctive decomposition, disjunctive sum, disjunctive products, linear transformation.*

Пусть $\mathcal{F}_n = \{f : \mathrm{GF}(2)^n \to \mathrm{GF}(2)\}$ — множество двоичных функций от n переменных, $n \geqslant 1$, $\mathbf{H}_n$ — группа сдвигов. Для каждого целого $s \geqslant 0$ определим подпространство $\mathcal{U}_s = \{f : \deg f \leqslant s\}$ пространства функций $\mathcal{F}_n$, имеющих ограниченную

степень нелинейности. Заметим, что $\mathcal{U}_0 = \{0, 1\}$. При $s < 0$ положим $\mathcal{U}_s = \{0\}$ — нулевое подпространство. Обозначим $(\mathbf{H}_n)_f^{(s)}$ множество таких сдвигов $\begin{pmatrix} x \\ x \oplus a \end{pmatrix} \in \mathbf{H}_n$, что выполнено сравнение

$$f(x \oplus a) \equiv f(x) \,(\mathrm{mod}\, \mathcal{U}_s), \quad x \in \mathrm{GF}(2)^n.$$

При $s < 0$ группа $(\mathbf{H}_n)_f^{(s)}$ является обычной группой инерции $(\mathbf{H}_n)_f$ функции f.

Пусть $0 \leqslant t \leqslant n-1$, $1 \leqslant k \leqslant n$. Будем говорить, что переменные $x_{k+1}, \ldots, x_n$ функции $f(x_1, \ldots, x_n)$ являются несущественными по модулю $\mathcal{U}_s$, если найдётся функция $h(x_1, \ldots, x_k)$, такая, что $f \oplus h \in \mathcal{U}_s$. Нетрудно видеть, что переменная x_n является несущественной для функции f по модулю $\mathcal{U}_s$, если и только если

$$\begin{pmatrix} x \\ x \oplus e_n \end{pmatrix} \in (\mathbf{H}_n)_f^{(s-1)}$$

при $e^n = (0, \ldots, 0, 1)$.

Определение 1. Пусть $*$ — бинарная ассоциативная операция. Будем говорить, что функция $f \in \mathcal{F}_n$ *линейно разложима относительно* $*$ *по модулю* $\mathcal{U}_s$, если при некотором линейном преобразовании A пространства $\mathrm{GF}(2)^n$ и $1 \leqslant k < n$ найдутся функции f_1 и f_2, для которых выполнено сравнение

$$f(xA) \equiv f_1(x_1, \ldots, x_k) * f_2(x_{k+1}, \ldots, x_n) \,(\mathrm{mod}\, \mathcal{U}_s).$$

Заметим, что всего имеется четыре бинарных ассоциативных операции: $x \oplus y$, $x \cdot y$, $x \oplus y \oplus 1$ и $x \vee y$, однако при $s \geqslant 0$ достаточно ограничиться рассмотрением только двух операций: сложения и умножения.

1. Бесповторная сумма функций

Сначала исключим случай наличия слагаемых первой степени. Каждая функция, представимая в виде суммы функций с непересекающимися наборами аргументов, среди которых есть слагаемые, имеющие вид переменных в первой степени, линейно эквивалентна функции, у которой такое слагаемое только одно. Пусть, например, это x_n. Тогда вектор $e^n = (0, \ldots, 0, 1)$ порождает инвариантное подпространство, а описание групп инерции таких функций имеет следующий вид.

Теорема 1. Если имеет место равенство

$$f(x_1, \ldots, x_n) = h(x_1, \ldots, x_{n-1}) \oplus x_n$$

и $(\mathbf{H}_n)_f = 1$, то справедливы следующие изоморфизмы:

$$\mathbf{GL}\,(n, 2)_f \cong \mathbf{GL}\,(n-1, 2)_h^{(1)};$$
$$\mathbf{AGL}\,(n, 2)_f \cong \mathbf{AGL}\,(n-1, 2)_h^{(1)};$$
$$\mathbf{AGL}\,(n, 2)_f^{(0)} \cong \mathbf{AGL}\,(n-1, 2)_h^{(1)} \times \mathbf{H}_1.$$

Действительно, в этом случае $|(\mathbf{H}_n)_f^{(1)}| = 2$, причём $\begin{pmatrix} x \\ x \oplus e^n \end{pmatrix} \in (\mathbf{H}_n)_f^{(1)}$. Поэтому матрицы из группы $\mathrm{Pr}_{\mathbf{GL}\,(n,2)}\mathbf{AGL}\,(n, 2)_f^{(0)}$ имеют вид

$$Q = \begin{pmatrix} A & b \\ 0 \ldots 0 & 1 \end{pmatrix}.$$

Искомый изоморфизм задаётся соответствием $(Q,h)\mapsto((A,h^1),h_n)$, где $h=(h_1,\ldots,h_{n-1},h_n)=(h^1,h_n)$.

Заметим, что для сумм слагаемых второй степени и $s\leqslant 1$ ни об однозначности разложения, ни о сведении вычисления группы инерции всей функции к вычислению групп инерции слагаемых в принципе не может быть и речи, так как полученные функции имеют неприводимые группы инерции, в качестве которых выступают классические линейные симплектическая и ортогональная группы, являющиеся неприводимыми линейными группами.

В то же время для слагаемых степени три и выше, как правило, такое сведение уже может иметь место. Найдём достаточно общие условия, при которых можно показать однозначность разложения функции степени k в бесповторную сумму по модулю $\mathcal{U}_s$ при $-1\leqslant s\leqslant k-1$. Естественно, что для этого разложение должно иметь максимально возможное число слагаемых.

Сначала рассмотрим частный случай, когда разложение имеет вид, аналогичный каноническому представлению квадратичной формы.

Пусть $[G]\mathbf{S}_p$ обозначает экспоненцирование линейной группы $G\leqslant\mathbf{GL}(m,2)$ и симметрической группы $\mathbf{S}_p$ степени p, $[G]\mathbf{S}_p\leqslant\mathbf{GL}(mp,2)$. Эта группа состоит из матриц, полученных путем замещения ненулевых элементов подстановочных матриц произвольными матрицами из группы G. Воспользуемся следующей удобной конструкцией, введённой М. В. Лариным. Обозначим $\langle M_f\rangle$ подпространство, порождённое множеством $M_f\subset\mathrm{GF}(2)^n$.

Утверждение 1. Пусть множество $M_f=\{a\in\mathrm{GF}(2)^n : f(a)=1\}$ удовлетворяет условию$\langle M_f\rangle=\mathrm{GF}(2)^n$, причём его можно представить в виде такого нетривиального разбиения $M_f=M_1\cup\ldots\cup M_m$, что $\mathrm{GF}(2)^n=\langle M_1\rangle+\ldots+\langle M_m\rangle$ — разложение в прямую сумму и m — максимальное число с этим свойством. Тогда:

1) группа $\mathbf{GL}\,(n,2)_f$ сохраняет это разбиение и разложение;
2) если множество функций $\{f_i : M_{f_i}=M_i,\ i=1,\ldots,m\}$ разбивается на классы эквивалентности относительно $\mathbf{GL}\,(n,2)$ вида $\{f_{\mu_1},\ldots,f_{\mu_p}\},\ldots,\{f_{\nu_1},\ldots,f_{\nu_q}\}$, то

$$\mathbf{GL}\,(n,2)_f\cong[\mathbf{GL}\,(n_{\mu_1},2)_{f_{\mu_1}}]\mathbf{S}_p\times\cdots\times[\mathbf{GL}\,(n_{\nu_1},2)_{f_{\nu_1}}]\mathbf{S}_q.$$

Доказательство. Утверждение следует из того, что если два разбиения $M_f=$ $=M_1\cup\ldots\cup M_m$ и $M_f=L_1\cup\ldots\cup L_r$ порождают разложение пространства в прямую сумму, то их пересечение $M_f=\bigcup\limits_{i,j}M_i\cap L_j$ также порождает разложение пространства в прямую сумму подпространств. ■

Теорема 2 [1]. Если $n=mk$, $k\geqslant 3$, $m\geqslant 2$ и функция f имеет вид

$$f(x_1,\ldots,x_n)=\sum_{i=0}^{m-1}x_{ik+1}x_{ik+2}\cdots x_{ik+k},$$

то группа $\mathrm{Pr}_{\mathbf{GL}\,(n,2)}\mathbf{AGL}\,(n,2)_f^{(k-1)}$ изоморфно вложима в группу $[\mathbf{GL}\,(k,2)]\mathbf{S}_m$.

Доказательство. В данном случае множество векторов, для которых производные имеют линейные сомножители по модулю $\mathcal{U}_{k-2}$, совпадает с объединением подпространств $\langle e_{ik+1},e_{ik+2},\ldots,e_{ik+k}\rangle$, $i=1,\ldots,m$, где e_j — векторы стандартного базиса, $j=1,\ldots,n$. Это множество удовлетворяет условиям утверждения 1 и является инвариантным относительно группы $\mathrm{Pr}_{\mathbf{GL}\,(n,2)}\mathbf{AGL}\,(n,2)_f^{(k-1)}$. ■

Заметим, что в неявной форме в терминах полилинейных форм этот результат сформулирован в работе [2].

Перейдём теперь к рассмотрению общего случая (данный результат анонсирован в [3]). В дальнейшем будем использовать понятие подпространства существенных переменных (подробнее см. в [4]). Вместо двоичных функций на пространстве $\mathrm{GF}(2)^n$, являющемся множеством двоичных векторов, удобно рассматривать функции, заданные на произвольном пространстве V размерности n над полем $\mathrm{GF}(2)$. При фиксации базиса $e = (e^1, \ldots, e^n)$ этого пространства функция $f : V \to \mathrm{GF}(2)$ может быть записана в виде двоичной функции $f_e(x_1, \ldots, x_n)$, где $f_e : \mathrm{GF}(2)^n \to \mathrm{GF}(2)$, а каждой переменной $x_i = (x, e^{*i})$ соответствует вектор сопряжённого базиса e^{*i} из сопряжённого пространства V^*.

Если функция f зависит по модулю $\mathcal{U}_t$ существенно лишь от k, $1 \leqslant k < n$, переменных, т. е.

$$f(x) = f_e(x_e) \equiv h_e(x_1, \ldots, x_k) \pmod{\mathcal{U}_t},$$

причём k — минимальное с этим свойством по всем базисам (или, что то же самое, по всем линейным заменам переменных), то с этой функцией однозначно связаны два подпространства: подпространство $W = \langle e^{k+1}, \ldots, e^n \rangle \subseteq V$ векторов, сдвиги по которым лежат в группе $(\mathbf{H}_n)_f^{(t-1)}$, и двойственное ему подпространство

$$W^* = (W)^\perp = \{e^* : (x, e^*) = 0, x \in V_2\} = \langle e^{*1}, \ldots, e^{*k} \rangle \subseteq V^*,$$

называемое *подпространством существенных переменных по модулю* $\mathcal{U}_t$. В этом случае будем использовать обозначение $f = f(W^*)$.

Нетрудно видеть, что функция $f = f(V^*)$ линейно разложима в бесповторную сумму по модулю $\mathcal{U}_s$, если при некотором нетривиальном разложении пространства V^* в прямую сумму $V^* = V_1^* + V_2^*$ функция имеет вид $f \equiv f_1(V_1^*) \oplus f_2(V_2^*) \pmod{\mathcal{U}_s}$.

Лемма 1. Пусть имеется два разложения пространства V^* в прямую сумму ненулевых подпространств $V^* = V_1^* + V_2^* = U_1^* + U_2^*$. Если при $s \geqslant 2$ функция $f = f(x_1, \ldots, x_n) = f(V^*)$ имеет тривиальную группу инерции $(\mathbf{H}_n)_f^{(s-1)}$ и выполняется сравнение

$$f \equiv f_1(V_1^*) \oplus f_2(V_2^*) \equiv h_1(U_1^*) \oplus h_2(U_2^*) \pmod{\mathcal{U}_s}, \tag{1}$$

то функция f допускает разложение

$$f \equiv d_1(W_{11}^*) \oplus d_2(W_{12}^*) \oplus d_3(W_{21}^*) \oplus d_4(W_{22}^*) \pmod{\mathcal{U}_s},$$

где $W_{ij}^* = V_i^* \bigcap U_j^*$, $i, j = 1, 2$, удовлетворяют условиям

$$\begin{cases} V_1^* = W_{11}^* + W_{12}^*, \\ V_2^* = W_{21}^* + W_{22}^*, \\ U_1^* = W_{11}^* + W_{21}^*, \\ U_1^* = W_{12}^* + W_{22}^*, \end{cases}$$

d_i, $i = 1, \ldots, 4$, — некоторые функции, удовлетворяющие следующим сравнениям:

$$\begin{cases} f_1 \equiv d_1 \oplus d_2 \pmod{\mathcal{U}_s}, \\ f_2 \equiv d_3 \oplus d_4 \pmod{\mathcal{U}_s}, \\ h_1 \equiv d_1 \oplus d_3 \pmod{\mathcal{U}_s}, \\ h_2 \equiv d_2 \oplus d_4 \pmod{\mathcal{U}_s}. \end{cases}$$

Доказательство. При $n \leqslant 5$ таких разложений у функции f не существует. При $n = 6$ однозначность разложения (1) вытекает из теоремы 2. Пусть теорема верна для всех функций от $n-1 \geqslant 6$ переменных, докажем её для функции f от n переменных. Из условия $(\mathbf{H}_n)_f^{(s-1)} = 1$ следует, то все производные $\Delta_a f$, $0 \neq a \in V$, функции f имеют степень нелинейности не меньше s. Выберем вектор $0 \neq a \in V$ так, чтобы у производной $\Delta_a f$ было минимальное число существенных переменных по модулю $\mathcal{U}_s$. Так как производная суммы функций равна сумме соответствующих производных, то число существенных переменных по модулю $\mathcal{U}_s$ у производной $\Delta_a f$ суммы $f = f_1(V_1^*) \oplus f_2(V_2^*)$ может быть минимальным только в том случае, когда $a \in V_1$ или $a \in V_2$, где $V = V_1 + V_2$ — соответствующее разложение пространства V. Аналогично получаем, что $a \in U_1$ или $a \in U_2$. Пусть для определённости $a \in V_1 \cap U_1$.

Дополним вектор a до базиса пространства V так, что $e^1 = a$, $\langle e^1, \ldots, e^k\rangle = V_1$, $\langle e^{k+1}, \ldots, e^n\rangle = V_2$. Пусть $\langle e^{*1}, \ldots, e^{*n}\rangle$ — сопряжённый базис пространства V^*, такой, что $\langle e^{*1}, \ldots, e^{*k}\rangle = V_1^*$, $\langle e^{*k+1}, \ldots, e^{*n}\rangle = V_2^*$. Обозначим $U^* = (\langle a\rangle)^\perp = \langle e^{*2}, \ldots, e^{*n}\rangle \subset V^*$.

Выберем еще один базис $\langle u^1, \ldots, u^n\rangle$ пространства V так, чтобы $u^1 = a$ и $\langle u^1 \ldots, u^k\rangle = U_1$, $\langle u^{t+1}, \ldots, u^n\rangle = U_2$. Обозначим сопряжённый базис через $u^{*1}, \ldots, u^{*n}$, $\langle u^{*1}, \ldots, u^{*t}\rangle = U_1^*$, $\langle u^{*t+1}, \ldots, u^{*n}\rangle = U_2^*$. Имеем $(a, e^{*1}) = (a, u^{*1}) = 1$, $(a, e^{*i}) = (a, u^{*i}) = 0$ при $2 \leqslant i \leqslant n$.

Нетрудно видеть, что $U^* = \langle e^{*2}, \ldots, e^{*n}\rangle = \langle u^{*2}, \ldots, u^{*n}\rangle$.

Рассмотрим два случая:

1) $u^{*1} \in V_1^*$;

2) $u^{*1} \notin V_1^*$.

В п е р в о м с л у ч а е можно предполагать, что $u^{*1} = e^{*1}$. Тогда сравнение (1) принимает вид

$$f = f_1(x_1, \widetilde{V_1^*}) \oplus f_2(V_2^*) \equiv h_1(x_1, \widetilde{U_1^*}) \oplus h_2(U_2^*) \pmod{\mathcal{U}_s}, \tag{2}$$

где $x_1 = (x, e^{*1}) = (x, u^{*1})$, $\widetilde{V_1^*} = \langle e^{*2}, \ldots, e^{*k}\rangle$, $\widetilde{U_1^*} = \langle u^{*2}, \ldots, u^{*t}\rangle$, причём выполнены равенства $U^* = \widetilde{V_1^*} + V_2^* = \widetilde{U_1^*} + U_2^*$. Разложим функции f_1 и h_1 по первой переменной:

$$\begin{aligned} f_{1e}(x_1, \widetilde{V_1^*}) &\equiv x_1 f'_{1e}(\widetilde{V_1^*}') \oplus f_{1e}^{(0)}(\widetilde{V_1^*}^{(0)}) \pmod{\mathcal{U}_s}, \\ h_{1u}(x_1, \widetilde{U_1^*}) &\equiv x_1 h'_{1u}(\widetilde{U_1^*}') \oplus h_{1u}^{(0)}(\widetilde{U_1^*}^{(0)}) \pmod{\mathcal{U}_s}, \end{aligned}$$

где

$$\begin{aligned} &f'_{1e}(\widetilde{V_1^*}') = f_{1u}(1, \widetilde{U_1^*}) \oplus f_{1u}(0, \widetilde{U_1^*}), \\ &f_{1e}^{(0)}(\widetilde{V_1^*}^{(0)}) = f_{1u}(0, \widetilde{U_1^*}), \\ &h'_{1u}(\widetilde{U_1^*}') = h_{1u}(1, \widetilde{U_1^*}) \oplus h_{1u}(0, \widetilde{U_1^*}), \\ &h_{1u}^{(0)}(\widetilde{U_1^*}^{(0)}) = h_{1u}(0, \widetilde{U_1^*}), \end{aligned}$$

$\widetilde{V_1^*}'$, $\widetilde{V_1^*}^{(0)} \subseteq V_1^*$ и $\widetilde{U_1^*}'$, $\widetilde{U_1^*}^{(0)} \subseteq U_1^*$ — соответствующие пространства существенных переменных по модулю $\mathcal{U}_s$. Подставляя в сравнение (2) значения $x_1 = 0$ и $x_1 = 1$, получаем, что оно равносильно системе

$$\begin{cases} f_{1e}^{(0)}(\widetilde{V_1^*}^{(0)}) \oplus f_{2e}(V_2^*) \equiv h_{1u}^{(0)}(\widetilde{U_1^*}^{(0)}) \oplus h_{2u}(U_2^*) \pmod{\mathcal{U}_s}, \\ f'_{1e}(\widetilde{V_1^*}') \equiv h'_{1u}(\widetilde{U_1^*}') \pmod{\mathcal{U}_{s-1}}. \end{cases}$$

Так как в этих сравнениях стоят функции, у которых все переменные существенны по модулю $\mathcal{U}_s$, то, в частности, получаем $\widetilde{V_1^*} + V_2^* = \widetilde{U_1^*} + U_2^*$ и $\widetilde{V_1^*}' = \widetilde{U_1^*}'$.

Введём обозначения для подпространств пространства U^*:

$$\begin{cases} \widetilde{W}_{11}^{*(0)} &= \widetilde{V_1^*}^{(0)} \bigcap \widetilde{U_1^*}^{(0)}, \\ W_{12}^{*(0)} &= \widetilde{V_1^*}^{(0)} \bigcap U_2^* = V_1^* \bigcap U_2^*, \\ W_{21}^{*(0)} &= V_2^* \bigcap \widetilde{U_1^*}^{(0)} = V_2^* \bigcap U_1^*, \\ W_{22}^* &= V_2^* \bigcap U_2^*. \end{cases}$$

По предположению индукции для пространства U^*, размерность которого меньше размерности пространства V^*, найдутся функции $d_i^{(0)}$, такие, что функция $f^{(0)}$ допускает разложение

$$f^{(0)} \equiv d_1^{(0)}(\widetilde{W}_{11}^{*(0)}) \oplus d_2^{(0)}(W_{12}^{*(0)}) \oplus d_3^{(0)}(W_{21}^{*(0)}) \oplus d_4^{(0)}(W_{22}^*) \pmod{\mathcal{U}_s}.$$

При этом должны выполняться следующие сравнения:

$$\begin{cases} f_1^{(0)} &\equiv d_1^{(0)} \oplus d_2^{(0)} \pmod{\mathcal{U}_s}, \\ f_2 &\equiv d_3^{(0)} \oplus d_4^{(0)} \pmod{\mathcal{U}_s}, \\ h_1^{(0)} &\equiv d_1^{(0)} \oplus d_3^{(0)} \pmod{\mathcal{U}_s}, \\ h_2 &\equiv d_2^{(0)} \oplus d_4^{(0)} \pmod{\mathcal{U}_s}. \end{cases}$$

Полагая $d'_{1e} \equiv f'_{1e} \equiv h'_{1e} \pmod{\mathcal{U}_{s-1}}$, $d_1 = x_1 d'_1 \oplus d_1^{(0)}$, $W_{11}^* = \langle e^{*1} \rangle + \widetilde{V_1^*}' + \widetilde{W}_{11}^{*(0)}$ и $d_i = d_i^{(0)}$, $i = 1, 2, 3$, получаем требуемое утверждение.

В о в т о р о м с л у ч а е, не уменьшая общности, можно предполагать, что вектор u^{*1} имеет вид $u^{*1} = e^{*1} + e^{*k+1}$. Поэтому сравнение (1) принимает вид

$$f \equiv f_1(x_1, \widetilde{V}_1^*) \oplus f_2(V_2^*) \equiv h_1(x_1 \oplus x_{t+1}, \widetilde{U}_1^*) \oplus h_2(U_2^*) \pmod{\mathcal{U}_s}, \tag{3}$$

где $U_1^* = \langle e^{*1} + e^{*t+1} \rangle + \widetilde{U_1^*}$ и $U^* = \widetilde{V_1^*} + V_2^* = \widetilde{U_1^*} + U_2^*$.

Рассуждая аналогично, получаем, что (3) равносильно системе

$$\begin{cases} f_{1e}^{(0)}(\widetilde{V_1^*}^{(0)}) \oplus f_{2e}(V_2^*) \equiv x_{t+1} h'_{1u}(\widetilde{U_1^*}') \oplus h_{1u}^{(0)}(\widetilde{U_1^*}^{(0)}) \oplus h_{2u}(U_2^*) \pmod{\mathcal{U}_s}, \\ f'_{1e}(\widetilde{V_1^*}') \equiv h'_{1u}(\widetilde{U_1^*}') \pmod{\mathcal{U}_{s-1}}, \end{cases}$$

где $h_{1u}(y, \widetilde{U_1^*}) = y h'_{1u}(\widetilde{U_1^*}') \oplus h_{1u}^{(0)}(\widetilde{U_1^*}^{(0)})$, $h_{1u}^{(0)}(\widetilde{U_1^*}^{(0)}) = h_{1u}(0, \widetilde{U_1^*})$.

Теперь из второго сравнения получаем равенство подпространств $\widetilde{V_1^*}' = \widetilde{U_1^*}'$. С другой стороны, в первом равенстве в правой части содержится произведение переменной x_{t+1} на функцию h'_{1u}, зависящую от переменных из множества $\widetilde{U_1^*}'$, а в левой части переменная x_{t+1} может входить только во второе слагаемое, зависящее от переменных $x_{t+1}, \ldots, x_n$. Но это может быть только в том случае, когда $\widetilde{U_1^*}'$ — пустое множество, а переменная x_1 является линейным слагаемым. Получаем противоречие с тем, что по условию x_1 является существенной переменной по модулю $\mathcal{U}_s$ при $s \geqslant 2$. ■

Теорема 3. Если при $s \geqslant 2$ функция $f = f(x_1, \ldots, x_n)$ имеет тривиальную группу инерции $(\mathbf{H}_n)_f^{(s-1)}$ и линейно разложима в бесповторную сумму по модулю $\mathcal{U}_s$, то для этой функции найдётся однозначно определённое линейное разложение по модулю $\mathcal{U}_s$ в бесповторную сумму линейно неразложимых (в бесповторную сумму) слагаемых в том смысле, что любое другое такое разложение соответствует тому же самому разложению пространства в сумму подпространств, а соответствующие функции сравнимы по модулю $\mathcal{U}_s$.

Доказательство. Предположим, что имеется два разложения

$$f(V^*) \equiv f_1(V_1^*) \oplus f_2(V_2^*) \oplus \ldots \oplus f_m(V_m^*) \pmod{\mathcal{U}_s}, \quad m \geqslant 2,$$
$$f(V^*) \equiv h_1(U_1^*) \oplus h_2(U_2^*) \oplus \ldots \oplus h_l(U_l^*) \pmod{\mathcal{U}_s}, \quad l \geqslant 2,$$

в которых функции f_i и h_j линейно неразложимы в бесповторную сумму по модулю $\mathcal{U}_s$, $i = 1, \ldots, m$, $j = 1, \ldots, l$. Если разложения пространства

$$V^* = V_1^* + V_2^* + \ldots + V_m^* = U_1^* + U_2^* + \ldots + U_l^*$$

различны и не получаются одно из другого перенумерацией подпространств, то найдётся нетривиальное пересечение $V_i^* \cap U_j^*$ при некоторых i, j, и с помощью леммы 1 получаем противоречие с линейной неразложимостью функций f_i и h_j в бесповторную сумму по модулю $\mathcal{U}_s$, $i = 1, \ldots, m$, $j = 1, \ldots, l$. Поэтому разбиения пространства совпадают, а соответствующие функции сравнимы по модулю $\mathcal{U}_s$. ■

В качестве следствия получаем описание группы инерции таких функций в полной аффинной группе.

Следствие 1. Если в условиях теоремы 3 функция f представлена в виде суммы линейно неразложимых в бесповторную сумму по модулю $\mathcal{U}_s$ функций

$$f \equiv f_1 \oplus \ldots \oplus f_m \pmod{\mathcal{U}_s},$$

причём множество функций $\{f_1, \ldots, f_m\}$ разбивается на t классов аффинной эквивалентности по модулю $\mathcal{U}_s$: $\{f_{\mu_1}, \ldots, f_{\mu_p}\} \in \mathcal{F}_{n_1}, \ldots, \{f_{\nu_1}, \ldots, f_{\nu_q}\} \in \mathcal{F}_{n_t}$, то для группы инерции бесповторной суммы этих функций справедлив изоморфизм

$$\mathbf{AGL}\,(n,2)^{(s)}_{f_1 \oplus \ldots \oplus f_m} \cong [\mathbf{AGL}\,(n_1,2)^{(s)}_{f_{\mu_1}}]\mathbf{S}_p \times \ldots \times [\mathbf{AGL}\,(n_t,2)^{(s)}_{f_{\nu_1}}]\mathbf{S}_q.$$

Аналогичное описание справедливо для группы $\mathbf{GL}\,(n,2)$.

2. Бесповторное произведение функций

Как и в случае слагаемых первой степени, у функции необходимо сначала выделить аффинные сомножители. Говорят, что *функция f имеет аффинный сомножитель по модулю $\mathcal{U}_s$*, $-1 \leqslant s \leqslant n-1$, если найдутся такие функция $l(x) = (x, a^*) \oplus b$, $0 \neq a^* \in V^*$, $b \in V$, и функция h, что $f \equiv lh \pmod{\mathcal{U}_s}$. Приведём простейшие свойства таких функций.

Лемма 2. Пусть аффинная функция $l(x) = (x, a^*) \oplus b$ отлична от константы. Следующие условия равносильны:

(а) f имеет аффинный сомножитель l по модулю $\mathcal{U}_s$;
(б) $lf \equiv f \pmod{\mathcal{U}_{s+1}}$;
(в) $\overline{l}f \equiv 0 \pmod{\mathcal{U}_{s+1}}$.

Доказательство. Импликации (а) $\Rightarrow$ (б) и (б) $\Leftrightarrow$ (в) очевидны. Докажем (б) $\Rightarrow$ (а). Пусть $l(x) = (x, e^*) \oplus b$, $b \in \{0, 1\}$. Вектор e^* должен принадлежать пространству существенных переменных функции f, иначе $\deg lf = \deg f + 1$. Поэтому достаточно рассмотреть случай $l(x_1, \ldots, x_n) = x_1 \oplus b = x_1^{1-b}$. Разложим функцию f по первой переменной:

$$f(x_1, x_2, \ldots, x_n) = \overline{x}_1 f^{(0)}(x_2, \ldots, x_n) \oplus x_1 f^{(1)}(x_2, \ldots, x_n).$$

Условие $lf \overset{s+1}{\equiv} f$ имеет вид $x_1^{1-b} f^{(1-b)} \overset{s+1}{\equiv} \overline{x}_1 f^{(0)} \oplus x_1 f^{(1)}$, что эквивалентно $f^{(b)} \in \mathcal{U}_s$, откуда

$$f = \overline{x}_1 f^{(0)} \oplus x_1 f^{(1)} = x_1^{1-b} f^{(1-b)} \oplus x_1^b f^{(b)} =$$
$$= x_1^{1-b}(f^{(0)} \oplus f^{(1)}) \oplus f^{(b)} \equiv x_1^{1-b}(f^{(0)} \oplus f^{(1)}) \pmod{\mathcal{U}_s}.$$

Следствие доказано. ■

Следствие 2. Если $\deg f = k$, то f не имеет аффинных сомножителей по модулю $\mathcal{U}_{k-1}$ в том и только в том случае, когда $(x, e^*)f \not\equiv 0 (\bmod \mathcal{U}_k)$ при всех $0 \neq e^* \in V^*$, или, что то же самое, $\deg (x, e^*)f = k + 1$.

Доказательство. Если $(x, e^*) \oplus b$ — аффинный сомножитель по модулю $\mathcal{U}_{k-1}$, то $[(x, e^*) \oplus b]f \equiv f \pmod{\mathcal{U}_k}$, откуда $(x, e^*)f \equiv 0 \pmod{\mathcal{U}_k}$. С другой стороны, если $(x, e^*)f \equiv 0 \pmod{\mathcal{U}_k}$ при некотором $0 \neq e^* \in V^*$, то $[(x, e^*) \oplus 1]f \equiv f \pmod{\mathcal{U}_k}$ и поэтому $(x, e^*) \oplus 1$ — аффинный сомножитель по модулю $\mathcal{U}_{k-1}$. ■

Лемма 3. Пусть разложение функции f по первой переменной имеет вид

$$f(x_1, x_2, \ldots, x_n) = \overline{x}_1 f^{(0)}(x_2, \ldots, x_n) \oplus x_1 f^{(1)}(x_2, \ldots, x_n)$$

и $-1 \leqslant s \leqslant \deg f - 2$. Тогда f не имеет аффинных сомножителей по модулю $\mathcal{U}_s$ в том и только в том случае, когда функции $f^{(0)}, f^{(1)}$ не принадлежат $\mathcal{U}_{s-1}$ и не имеют аффинных сомножителей по модулю $\mathcal{U}_{s-1}$ с одинаковой линейной частью.

Доказательство. Пусть $e^1, \ldots, e^n$ — базис пространства V и $e^{*1}, \ldots, e^{*n}$ — сопряжённый базис. Рассмотрим разложение функции f по первой переменной $f = \overline{x}_1 f^{(0)} \oplus$ $\oplus\, x_1 f^{(1)}$, где $f_e^{(0)}(x_2, \ldots, x_n) = f_e(0, x_2, \ldots, x_n)$, $f_e^{(1)}(x_2, \ldots, x_n) = f_e(1, x_2, \ldots, x_n)$. Можно считать, что функции $f^{(0)}$ и $f^{(1)}$ заданы на пространстве $U = \langle e^2, \ldots, e^n \rangle = \langle e^{*1} \rangle^{\perp}$, поэтому

$$f = \overline{x}_1 f^{(0)}(U^*) \oplus x_1 f^{(1)}(U^*),$$

где $U^* = \langle e^{*2}, \ldots, e^{*n} \rangle$.

Пусть $(x, e^*) \oplus b \neq 0$ — аффинный сомножитель функции f по модулю $\mathcal{U}_s$, $e^* \in V^*$, $b \in \{0, 1\}$. Если векторы e^{*1} и e^* совпадают, то $((x, e^*) \oplus b) = x_1^{1-b}$ и

$$((x, e^*) \oplus b)f = x_1^{1-b}(\overline{x}_1 f^{(0)} \oplus x_1 f^{(1)}) = x_1^{1-b} f^{(1-b)} \equiv f \pmod{\mathcal{U}_{s+1}},$$

откуда $f^{(b)} \in \mathcal{U}_s$. Если же векторы e^{*1} и e^* линейно независимы, то либо $e^* = e^{*1} + u^*$, $u^* \in U^*$, либо $e^* = u^* \in U^*$. Можно полагать, что $u^* = e^{*2}$. В первом случае имеем $(x, e^*) = x_1 \oplus x_2$ и $(x_1 \oplus x_2 \oplus b)f = \overline{x}_1 x_2^{1-b} f^{(0)}(U^*) \oplus x_1 x_2^b f^{(1)}(U^*) \equiv f \pmod{\mathcal{U}_{s+1}}$, откуда $x_2^{1-b} f^{(0)} = f^{(0)} \pmod{\mathcal{U}_s}$ и $x^b f^{(1)} = f^{(1)} \pmod{\mathcal{U}_s}$.

Во втором случае $((x, e^*) \oplus b) = x_2^{1-b}$ и

$$((x, e^*) \oplus b)f = x_2^{1-b}(\overline{x}_1 f^{(0)} \oplus x_1 f^{(1)}) = \overline{x}_1 x_2^{1-b} f^{(0)} \oplus x_1 x_2^{1-b} f^{(1)} \equiv f \pmod{\mathcal{U}_{s+1}}.$$

Отсюда $x_2^{1-b} f^{(i)} \equiv f^{(i)} \pmod{\mathcal{U}_s}$, $i = 1, 2$. Поэтому $(x, e^*) \oplus b$ — аффинный сомножитель обеих функций $f^{(0)}$ и $f^{(1)}$ по модулю $\mathcal{U}_{s-1}$.

Обратное утверждение очевидно. ■

Пусть $\mathcal{N}L_s \subset \mathcal{F}_n$ — подмножество функций, не имеющих аффинных сомножителей по модулю $\mathcal{U}_s$, $s \geqslant -1$. Легко видеть, что справедливы включения

$$\mathcal{N}L_{-1} \supset \mathcal{N}L_0 \supset \mathcal{N}L_1 \supset \cdots \supset \mathcal{N}L_{n-2} = \varnothing.$$

Следствие 3. Если в условиях леммы 3 $f^{(0)}, f^{(1)} \notin \mathcal{U}_{s-1}$ и $f^{(0)}, f^{(1)} \in \mathcal{N}L_{s-1}^{(n-1)}$, то $f \in \mathcal{N}L_s^{(n)}$.

Если функция f имеет k аффинных сомножителей по модулю $\mathcal{U}_s$

$$l_i(x) = (x, a^{*i}) \oplus b_i,$$

где $a^{*i} \in V_n^*$ линейно независимы, $b_i \in \{0,1\}$, $i = 1, \ldots, k$, $k \geqslant 1$, но не имеет $k+1$ таких сомножителей, то будем говорить, что она *имеет ровно k аффинных сомножителей по модулю $\mathcal{U}_s$*. Доказательство следующей леммы очевидно.

Лемма 4. Пусть $k \geqslant 1$ и $s \leqslant \deg f - 1$. Тогда следующие условия эквивалентны:

а) f имеет ровно k аффинных сомножителей по модулю $\mathcal{U}_s$;

б) при некоторой линейной замене переменных с матрицей A функция f удовлетворяет условию $f(xA) \equiv x_1^{b_1} \ldots x_k^{b_k} h(x_{k+1}, \ldots, x_n) (\mathrm{mod}\ \mathcal{U}_s)$, где $b_i \in \{0,1\}$, $i = 1, \ldots, k$, и h не имеет аффинных сомножителей по модулю $\mathcal{U}_{s-k}$.

В условиях леммы 4 с функцией f однозначно связаны также два подпространства: *подпространство аффинных сомножителей* $L_1^* = \langle e^{*1}, \ldots, e^{*k} \rangle \leqslant V^*$ и двойственное к нему подпространство $(L_1^*)^\perp \subseteq V$, первое из которых является инвариантным относительно линейных преобразований из группы $G = \mathrm{Pr}_{\mathbf{GL}(n,2)} \mathbf{AGL}(n,2)_f^{(s)}$, а второе — относительно группы $G^* = \{A : (A^t)^{-1} \in G\}$. Описание групп инерции таких функций в полной линейной и аффинной группах приведено в [1]. Поэтому далее будем предполагать, что функция раскладывается в произведение нелинейных сомножителей.

Заметим, что для случая точного равенства функций (случай $s = -1$) разложение двоичной функции в бесповторное произведение нелинейных неразложимых сомножителей изучалось в работе [5], где показана однозначность такого разложения с точностью до перестановки эквивалентных сомножителей.

Приведём результаты, позволяющие в некоторых случаях описывать группы инерции функций для случая сравнения функций по модулю $\mathcal{U}_s$ при $s \geqslant 0$ для произведения функций без аффинных сомножителей. С их помощью можно, в частности, описывать группы инерции в обобщённых группах для бесповторных произведений квадратичных форм. В основе применяемого подхода лежит изучение структуры множества векторов, производные по направлению по которым имеют аффинные сомножители.

Лемма 5. Пусть $f(x) \equiv f_1(x^1) f_2(x^2) \ (\mathrm{mod}\, \mathcal{U}_{k-1})$, $\deg f = k$, $\deg f_i = k_i > 1$, $x^i = (x_{i_1}, \ldots, x_{i_{n_i}})$, $x = (x^1, x^2)$, $i = 1, 2$. Обозначим через V_i пространство $V_i = \{e^{i_1}, \ldots, e^{i_{n_i}}\}$, $i = 1, 2$, $V = V_1 \oplus V_2$, $V_1 \cup V_2 = \{0\}$. Тогда

1) для всех $a = (a^1, a^2)$, $a^i \in V_i$, $i = 1, 2$, выполняется сравнение

$$\Delta_a f \equiv (\Delta_{a^1} f_1) f_2 \oplus (\Delta_{a^2} f_2) f_1 \ (\mathrm{mod}\, \mathcal{U}_{k-2});$$

2) f имеет линейные сомножители по модулю $\mathcal{U}_{k-1}$ в том и только в том случае, когда при некотором i, $i \in \{1, 2\}$, функция f_i имеет линейные сомножители по модулю $\mathcal{U}_{k_i - 1}$.

Доказательство. Первое утверждение леммы вытекает из равенства

$$\Delta_a f(x) = (\Delta_{a^1} f_1)(x^1) f_2(x^2) \oplus (\Delta_{a^2} f_2)(x^2) f_1(x^1 \oplus a^1).$$

Докажем второе утверждение. Пусть функция f имеет линейный сомножитель по модулю $\mathcal{U}_{k-1}$: $(x, l^{*1} \oplus l^{*2}) f(x) \in \mathcal{U}_k$, где $l^{*i} \in V_i^*$, $i = 1, 2$. Тогда

$$(x, l^{*1}) f_1(x^1) f_2(x^2) \oplus f_1(x^1)(x, l^{*2}) f_2(x^2) \in \mathcal{U}_k.$$

Отсюда $(x, l^{*i}) f_i(x^i) \in \mathcal{U}_{k_i}$, $i = 1, 2$. ■

Пусть $L_s(f)$ — множество векторов $a \in V$, таких, что $\Delta_a f$ имеет аффинные сомножители по модулю $\mathcal{U}_s$, $-1 \leqslant s \leqslant \deg f - 1$. Очевидны включения

$$V = L_{\deg f-1} \supset L_{\deg f-2} \supset \cdots \supset L_0 \supset L_{-1}.$$

Лемма 6. Пусть выполняется условие леммы 5. Если, кроме того, при $i = 1, 2$ $f_i \in \mathcal{N}L_{k_i-1}$ и $|(\mathbf{H}_n)_{f_i}^{(k_i-2)}| = 1$, то $L_{k-2}(f) = L_{k_1-2}(f_1) \cup L_{k_2-2}(f_2)$.

Доказательство. Достаточно проверить только включение

$$L_{k-2}(f) \subseteq L_{k_1-2}(f_1) \cup L_{k_2-2}(f_2).$$

Предположим, что при некотором $a \in V$ функция $\Delta_a f$ имеет аффинный сомножитель по модулю $\mathcal{U}_{k-2}$. Тогда по следствию 2 при некотором $0 \neq e^* \in V^*$

$$(x, e^*)\Delta_a f(x) \equiv 0 \pmod{\mathcal{U}_{k-1}}.$$

Пусть $a = a^1 \oplus a^2$, $a^i \in V_{n_i}$, $a^1 \neq 0$, $i = 1, 2$. Если $e^* = e^{*1} \oplus e^{*2}$, $e^{*i} \in V_{n_i}$, $i = 1, 2$, то предыдущее сравнение принимает вид

$$(x, e^{*1} \oplus e^{*2})(\Delta_{a^1} f_1 \cdot f_2 \oplus \Delta_{a_2} f_2 \cdot f_1) \equiv 0 \pmod{\mathcal{U}_{k-1}}.$$

Раскрывая скобки, получаем

$$(x, e^{*1})\Delta_{a^1} f_1 \cdot f_2 \oplus (x, e^{*1}) f_1 \cdot \Delta_{a^2} f_2 \oplus \Delta_{a^1} f_1 \cdot (x, e^{*2}) f_2 \oplus f_1 \cdot (x, e^{*2})\Delta_{a^2} f_2 \equiv 0 \pmod{\mathcal{U}_{k-1}}.$$

Одночлены степени $k = k_1 + k_2$, в которых имеется $k_1 - 1$ переменных из первого множества и $k_2 + 1$ переменных из второго множества, могут появиться только из слагаемого $\Delta_{a^1} f_1 \cdot (x, e^{*2}) f_2$. Так как $|(\mathbf{H}_{n_1})_{f_1}^{(k_1-2)}| = 1$, то $\deg \Delta_{a^1} f_1 = k_1 - 1$, откуда $(x, e^{*2}) f_2 \in \mathcal{U}_{k_2}$. С другой стороны, функция f_2 не имеет линейных сомножителей по модулю $\mathcal{U}_{k_2-1}$. Поэтому в силу следствия 2 должно быть $e^{*2} = 0$ и $e^* = e^{*1} \neq 0$. Аналогично из второго слагаемого получаем $a^2 = 0$ и $a = a^1 \in L_{k_1-2}(f_1)$. ■

Теорема 4. Пусть выполнено сравнение

$$f(x) \equiv f_1(x^1) \cdots f_m(x^m) \pmod{\mathcal{U}_{k-1}},$$

где $\deg f = k$, $\deg f_i = k_i > 1$, $x^i = (x_{i_1}, \ldots, x_{i_{n_i}})$, $x = (x^1, \ldots, x^m)$, $|(\mathbf{H}_n)_{f_i}^{(k_i-2)}| = 1$, $i = 1, \ldots, m$, причём m — максимальное число с таким свойством среди всех функций, получающихся из f линейной заменой переменных. Обозначим через V_i подпространство $V_i = \{e^{i_1}, \ldots, e^{i_{n_i}}\}$, $i = 1, \ldots, m$, $V = V_1 \oplus \ldots \oplus V_m$. Тогда если $f_i \in \mathcal{N}L_{k_i-1}$ и $\langle L_{k_i-2}(f_i)\rangle = V_i$, $i = 1, \ldots, m$, то группа $\mathrm{Pr}_{\mathbf{GL}(n,2)}(\mathbf{AGL}(n, 2))_f^{(k-1)}$ сохраняет разложение $V = V_1 \oplus \ldots \oplus V_m$ в прямую сумму подпространств.

Доказательство. Из леммы 6 следует, что справедливо разложение

$$L_{k-2}(f) = L_{k_1-2}(f_1) \cup \ldots \cup L_{k_m-2}(f_m).$$

Пусть $L_{k_i-2}(f_i) = M_{i,1} \cup \ldots \cup M_{i,t_i}$ — максимальное разбиение, удовлетворяющее условию $\langle L_{k_i-2}(f_i)\rangle = \langle M_{i,1}\rangle \oplus \ldots \oplus \langle M_{i,t_i}\rangle$, $t_i \geqslant 1$, $i = 1, \ldots, m$. Тогда разбиение

$$L_{k-2}(f) = \bigcup_{i=1}^{m} \bigcup_{j=1}^{t_i} M_{i,j}$$

также удовлетворяет условию утверждения 1. Поэтому группа $\mathrm{Pr}_{\mathbf{GL}\,(n,2)}(\mathbf{AGL}\,(n,2))_f^{(k-1)}$ должна сохранять разложение $V = \bigcup\limits_{i=1}^{m}\bigcup\limits_{j=1}^{t_i} V_{i,j}$, где $V_{i,j} = \langle M_{i,j}\rangle$, $i = 1,\ldots,m$, $j = 1,\ldots,t_i$.

Составим базис пространства V из базисов подпространств $V_{i,j}$, $i = 1,\ldots,m$, $j = 1,\ldots,t_i$. Тогда, сравнивая старшие члены в многочленах Жегалкина у функций $h(x) = f(xC)$, где C — матрица линейного преобразования, соответствующего переходу к этому базису, и $h(xQ)$ при $Q \in \mathbf{AGL}\,(n,2)_h^{(k-1)}$, получим, что $\mathrm{Pr}_{\mathbf{GL}\,(n,2)}\mathbf{AGL}\,(n,2)_h^{(k-1)}$, а следовательно, и группа $\mathrm{Pr}_{\mathbf{GL}(n,2)}\mathbf{AGL}\,(n,2)_f^{(k-1)}$ должны сохранять разложение в прямую сумму подпространств $V = V_1 \oplus \ldots \oplus V_m$, что и доказывает теорему. ■

Следствие 4. Если в условиях теоремы 4 множество функций $\{f_1,\ldots,f_m\}$ при некотором $s \geqslant 1$ разбивается на классы $\mathbf{AGL}\,(n,2)\mathcal{U}_{k_i-s}$-эквивалентности $\{f_{\mu_1},\ldots,f_{\mu_p}\}$, …, $\{f_{\nu_1},\ldots,f_{\nu_q}\}$, то

$$\mathbf{AGL}\,(n,2)_f^{(k-s)} \cong \left[\mathbf{AGL}\,(n_{\mu_1},2)_{f_{\mu_1}}^{(k_{\mu_1}-s)}\right]\mathbf{S}_p \times \cdots \times \left[\mathbf{AGL}\,(n_{\nu_1},2)_{f_{\nu_1}}^{(k_{\nu_1}-s)}\right]\mathbf{S}_q.$$

При $s = 1$ утверждение теоремы вытекает из теоремы 4. При $s > 1$ оно вытекает из того факта, что группа $\mathbf{AGL}\,(n,2)_f^{(k-s)}$ является подгруппой группы $\mathbf{AGL}\,(n,2)_f^{(k-1)}$.

Следствие 5. Если в условиях теоремы 4 функции f_i являются невырожденными квадратичными формами ранга $2r_i \geqslant 4$, $i = 1,\ldots,m$, причём при некотором s, $1 \leqslant s \leqslant 3$, множество функций $\{f_1,\ldots,f_m\}$ разбивается на классы $\mathbf{AGL}\,(n,2)\mathcal{U}_{2-s}$-эквивалентности $\{f_{\mu_1},\ldots,f_{\mu_p}\},\ldots,\{f_{\nu_1},\ldots,f_{\nu_q}\}$, то

$$\mathbf{AGL}\,(n,2)_f^{(2m-s)} \cong \left[\mathbf{AGL}\,(n_{\mu_1},2)_{f_{\mu_1}}^{(2-s)}\right]\mathbf{S}_p \times \cdots \times \left[\mathbf{AGL}\,(n_{\nu_1},2)_{f_{\nu_1}}^{(2-s)}\right]\mathbf{S}_q.$$

Доказательство. Квадратичные формы невырождены, если число переменных совпадает со значением ранга. Поскольку невырожденные квадратичные формы f_i ранга $2r_i \geqslant 4$, $i = 1,\ldots,m$, удовлетворяют условию теоремы 4, то для доказательства достаточно показать, что m — максимальное число сомножителей среди всех разложений функции f, полученных при различных линейных заменах переменных.

Предположим, что это не так. Тогда у функции f есть линейные сомножители по модулю $\mathcal{U}_{2m-1}$. Но в силу леммы 5 из условия $f_i \in \mathcal{N}L_1$, $i = 1,\ldots,m$, вытекает $f \in \mathcal{N}L_{2m-1}$. ■

Следствие 6. Пусть выполняются условия теоремы 4 и функции f_i являются квадратичными формами ранга $2r_i \geqslant 4$, $i = 1,\ldots,m$, причём все они имеют тривиальные группы инерции в группе $\mathbf{H}_{n_i}$, где n_i — число переменных формы f_i, $i = 1,\ldots,m$. Тогда если при некотором s, $2 \leqslant s \leqslant 3$, множество форм $\{f_1,\ldots,f_m\}$ разбивается на классы $\mathbf{AGL}\,(n,2)\mathcal{U}_{2-s}$-эквивалентности $\{f_{\mu_1},\ldots,f_{\mu_p}\},\ldots,\{f_{\nu_1},\ldots,f_{\nu_q}\}$, то

$$\mathbf{AGL}\,(n,2)_f^{(2m-s)} \cong \left[\mathbf{AGL}\,(n_{\mu_1},2)_{f_{\mu_1}}^{(2-s)}\right]\mathbf{S}_p \times \cdots \times \left[\mathbf{AGL}\,(n_{\nu_1},2)_{f_{\nu_1}}^{(2-s)}\right]\mathbf{S}_q.$$

Доказательство. Если все квадратичные формы невырождены, то утверждение вытекает из предыдущего следствия.

Рассмотрим теперь случай, когда некоторые квадратичные формы могут быть вырожденными. Пусть это $f_{t+1},\ldots,f_m$. С учётом классификации квадратичных форм

достаточно рассмотреть случай, когда ранг квадратичной формы на единицу меньше числа переменных, т. е. можно полагать, что

$$f_i(x_{i,1},\dots,x_{i,n_i}) = x_{i,1}x_{i,2} \oplus \dots \oplus x_{i,n_i-2}x_{i,n_i-1} \oplus x_{i,n_i},$$

$n_i = 2r_i + 1 = m_i + 1$, $i = t+1,\dots,m$. Если ввести обозначение

$$f_i(x_{i,1},\dots,x_{i,n_i}) = h_i(x_{i,1},\dots,x_{i,n_i-1}) \oplus x_{i,n_i},$$

то функции h_i — невырожденные квадратичные формы, $i = t+1,\dots,m$. Пусть

$$f(x) = f_1(x^1)\dots f_m(x^m), \qquad f'(x) = f_1\dots f_t h_t\dots h_m.$$

Так как $f \equiv f' \pmod{\mathcal{U}_{2m-1}}$, переменные $x_{i,n_{t+1}},\dots,x_{i,n_m}$ являются несущественными по модулю $\mathcal{U}_{2m-1}$. Поэтому у всякого аффинного преобразования (Q,b), такого, что

$$(Q,b) \in \mathbf{AGL}\,(n,2)_f^{(2m-s)} \subseteq \mathbf{AGL}\,(n,2)_f^{(2m-1)},$$

матрица Q после перенумерации переменных может быть приведена к виду

$$Q = \begin{pmatrix} \mathrm{A} & \mathrm{B} \\ 0 & \mathrm{C} \end{pmatrix},$$

где C — $n\times n$-матрица, $(A,b^1) \in \mathbf{AGL}\,(n-(m-t),2)_{f'}^{(2m-1)}$, $b = (b^1,b^2)$. В силу теоремы 4 получаем, что преобразование (A,b^1) должно переставлять между собой функции $f_1,\dots,f_t,h_{t+1},\dots,h_m$.

Далее, не уменьшая общности, можно полагать, что преобразование (A,b^1) не изменяет порядка следования функций $f_1,\dots,f_t$, $h_{t+1},\dots,h_m$. Покажем, что преобразование (Q,b) также оставляет на месте функции $f_1,\dots,f_m$. Имеем

$$f(x) \equiv f_1\dots f_t h_{t+1}\dots h_m \oplus \sum_{i=t+1}^{n} x_{i,n_i} f_1\dots f_t h_{t+1}\dots h_{i-1}h_{i+1}\dots h_m \pmod{\mathcal{U}_{2m-2}}.$$

Тогда из сравнения $f(x) \equiv f(xQ\oplus b)) \pmod{\mathcal{U}_s}$ вытекает, что

$$\sum_{i=t+1}^{n} (l_{i,n_i}(x) \oplus x_{i,n_i}) f_1\dots f_t h_{t+1}\dots h_{i-1}h_{i+1}\dots h_m \equiv 0 \pmod{\mathcal{U}_{2m-2}}, \tag{4}$$

где $l_{i,n_i}(x)$ — линейная функция, описывающая координату с номером n_i вектора $xQ\oplus b$, $i = t+1,\dots,m$.

Пусть $l_{i,n_i}(x) = l'_{i,n_i}(x)\oplus l''_{i,n_i}(x)$, где $l'_{i,n_i}(x)$ — часть слагаемых функции $l_{i,n_i}(x)$, которая зависит от переменных $x_{i,j}$, $1 \leqslant j \leqslant n_i$, относящихся к функции h_i, $i = t+1,\dots,m$, а $l''_{i,n_i}(x)$ — от оставшихся переменных. Из вида слагаемых в правой части сравнения (4) следует, что при каждом $i = t+1,\dots,m$ после умножения $l''_{i,n_i}(x)$ на произведение $f_1\dots f_t h_{t+1}\dots h_{i-1}h_{i+1}\dots h_m$ в качестве ненулевых слагаемых в сравнении получаются только одночлены, не содержащие ни одной переменной $x_{i,j}$, $1 \leqslant j \leqslant n_i$, но содержащие произведения более трёх переменных других функций. Такие одночлены могут встретиться только в i-м слагаемом суммы из сравнения (4) и, следовательно, не могут ни с чем сократиться. Поэтому при каждом $i = t+1,\dots,m$ должно выполняться сравнение

$$l''_{i,n_i}(x) f_1\dots f_t h_{t+1}\dots h_{i-1}h_{i+1}\dots h_m \equiv 0 \pmod{\mathcal{U}_{2m-2}}.$$

Поскольку функции $f_1,\dots,f_t,h_{t+1},\dots,h_m$ по условию не имеют линейных сомножителей, должны выполняться равенства $l''_{i,n_i}(x) = 0$, $i = t+1,\dots,m$, что означает равенство нулю элементов подматрицы B матрицы Q. ■

ЛИТЕРАТУРА

1. *Черемушкин А. В.* Методы аффинной и линейной классификации двоичных функций // Труды по дискретной математике. М.: Физматлит, 2001. Т. 4. С. 273–314.
2. *Dixon L. E.* Linear Groups with Exposition Galois Field Theory. Leipzig, 1901; 2nd ed.: N.Y.: Dover Publications, 1958.
3. *Черемушкин А. В.* Условие однозначности разложения двоичной функции в бесповторную сумму функций при линейной замене переменных // Прикладная дискретная математика. Приложение. 2017. № 10. С. 55–56.
4. *Черемушкин А. В.* К вопросу о линейной декомпозиции двоичных функций // Прикладная дискретная математика. 2016. № 1(31). С. 46–56.
5. *Черемушкин А. В.* Однозначность разложения двоичной функции в бесповторное произведение нелинейных неприводимых сомножителей // Вестник Московского государственного университета леса «Лесной вестник». 2004. № 4(35). С. 86–90.

REFERENCES

1. *Cheremushkin A. V.* Metody affinnoy i lineynoy klassifikatsii dvoichnykh funkiy [Methods of affine and linear classification of binary functions]. Tr. Diskr. Mat., 2001, vol. 4, pp. 273–314. (in Russian)
2. *Dixon L. E.* Linear Groups with Exposition Galois Field Theory. Leipzig, 1901; 2nd ed.: N.Y., Dover Publications, 1958.
3. *Cheremushkin A. V.* Uslovie odnoznachnosti razlozheniya dvoichnoy funktsii v bespovtornuyu summu funktsiy pri lineynoy zamene peremennykh [A condition for uniqueness of linear decomposition of a Boolean function into disjunctive sum of indecomposable functions]. Prikladnaya Diskretnaya Matematika. Prilozhenie, 2017, no. 10, pp. 55–56. (in Russian)
4. *Cheremushkin A. V.* K voprosu o lineynoy dekompozitsii dvoichnykh funktsiy [On linear decomposition of Boolean functions]. Prikladnaya Diskretnaya Matematika, 2016, no. 1(31), pp. 46–56. (in Russian)
5. *Cheremushkin A. V.* Odnoznachnost' razlozheniya dvoichnoy funktsii v bespovtornoe proizvedenie nelineynykh neprivodimykh somnozhiteley [The uniqueness of the binary function decomposition in a unrepeated product of non-linear irreducible factors]. Lesnoy vestnik, 2004, no. 4(35), pp. 86–90. (in Russian)

МАТЕМАТИЧЕСКИЕ МЕТОДЫ КРИПТОГРАФИИ

UDC 519.7 DOI 10.17223/20710410/40/3

ASYMMETRIC CRYPTOSYSTEMS ON BOOLEAN FUNCTIONS[1]

G. P. Agibalov, I. A. Pankratova

National Research Tomsk State University, Tomsk, Russia

Here, we define an asymmetric substitution cryptosystem combining both a public key cipher and a signature scheme with the functional keys. A public key in the cryptosystem is a vector Boolean function $f(x_1, \ldots, x_n)$ of a dimension n. This function is obtained by permutation and negation operations on variables and coordinate functions of a bijective vector Boolean function $g(x_1, \ldots, x_n) = (g_1(x_1, \ldots, x_n), \ldots, g_n(x_1, \ldots, x_n))$. The function g is called a generating function of the cryptosystem. For each $i \in \{1, \ldots, n\}$, its coordinate function $g_i(x_1, \ldots, x_n)$ is assumed to be specified in a constructive way and to have a polynomial (in n) complexity. A private key of the cryptosystem is the function f^{-1}, that is, the inverse of f. The existence of f^{-1} follows from the bijectiveness of g and preserving this property by permutation and negation operations. Function g and its coordinates $g_1, \ldots, g_n$ are public parameters of the cryptosystem. (A variant of the cryptosystem allows to include them into the private key). Of course, the permutation and negation operations by which a public key is computed from the generating function must be secret as private exponents in RSA and ElGamal cryptosystems. A block P of a plaintext is encrypted to a block C of a ciphertext by the rule $C = f(P)$, and C is decrypted to P by the rule $P = f^{-1}(C)$. A signature on a message M is computed as $S = f^{-1}(P)$, and its validation is proved by verifying the equality $M = f(S)$. This cryptosystem is believed to resist classical and quantum computers attacks. Its security is based on the difficulty of inverting large bijective vector Boolean functions. Cryptanalysis of the cryptosystem shows that its computational complexity can reach the value $\mathrm{O}(n!2^n)$.

Keywords: *vector Boolean functions, invertibility, asymmetric substitution cryptosystem, cryptanalysis.*

Introduction

Public-key cryptosystems are usually constructed on the base of number theory or algebraic structures and are very susceptible to quantum attacks. Perhaps the only exception to this rule are finite automaton public key cryptosystems [1]. In this paper, we suggest a public-key cryptosystem based on an invertible system of n Boolean functions which is variable like a cryptographic key by the permutation and negation operations on system's arguments and coordinates. We call it ACBF — Asymmetric Cryptosystem on Boolean Functions. The cryptosystem typically consists of two parts — a public-key cipher and a signature scheme. A general cryptanalysis scheme is described for both of them. According to this scheme, some particular known playntext (and known message) attacks are proposed for the universal ACBF and for its derivatives with some permutation and negation operations being identities. Estimates for computational complexity of these

[1]The authors were supported by the RFBR-grant no. 17-01-00354.

attacks are given too. The most of them is $O(n!2^n)$. For each of fifteen ACBF we considered, the proposed attacks on its cipher and signature scheme happened to have the same estimate of computational complexity.

1. Definition

In [2], we have defined a symmetric block substitution cipher with the functional keys. Here, by using the same construction, we define an asymmetric substitution cryptosystem including both a public key cipher and a signature scheme with the functional keys. To give a formal definition of this cryptosystem, we first define the permutation and negation operations. Let n be an integer, $n \geqslant 2$, and $\mathbb{S}_n$ be the set of all permutations of the row $(1\,2\,\ldots\,n)$, that is, $\mathbb{S}_n = \{(i_1 i_2 \ldots i_n) : i_j \in \{1, 2, \ldots, n\}, j \neq r \Rightarrow i_j \neq i_r; j, r \in \{1, \ldots, n\}\}$. A permutation $\pi = (i_1 i_2 \ldots i_n) \in \mathbb{S}_n$ is called a *permutation operation* if the result of its application to any word $w = w_1 w_2 \ldots w_n$ is the word $\pi(w) = w_{i_1} w_{i_2} \ldots w_{i_n}$. A Boolean vector $\sigma = b_1 b_2 \ldots b_n \in \mathbb{F}_2^n$ is called a *negation operation* if the result of its application to a string $\alpha = a_1 a_2 \ldots a_n$ of Boolean values (constants, variables or functions) $a_1, \ldots, a_n$ is the string $\alpha^\sigma = a_1^{b_1} a_2^{b_2} \ldots a_n^{b_n}$ where for a and b in $\mathbb{F}_2$, $a^b = a$ if $b = 1$ and $a^b = \neg a$ if $b = 0$. The permutation and negation operations π and σ are called *identity* and denoted by 1 if $\pi = (1\,2\,\ldots\,n)$ and $\sigma = 11\ldots1$ respectively.

Formally, our *asymmetric cryptosystem on Boolean functions* is a three-tuple $\mathcal{C} = (X, K, Y)$ where X is the set of plaintexts, or messages, $X \subseteq \mathbb{F}_2^n$, Y is the set of ciphertexts or signatures, $Y \subseteq \mathbb{F}_2^n$, and $K = K_1 \times K_2$ is the set of keys, K_1 — the set of public keys, $K_1 \subseteq K_n(g) = \{f(x) : f(x) = \pi_2(g^{\sigma_2}(\pi_1(x^{\sigma_1}))); \sigma_1, \sigma_2 \in \mathbb{F}_2^n; \pi_1, \pi_2 \in \mathbb{S}_n\}$; $x = (x_1, \ldots, x_n)$ is a string of different Boolean variables, $g : \mathbb{F}_2^n \to \mathbb{F}_2^n$ is a bijective vector Boolean function $g(x) = g_1(x)g_2(x)\ldots g_n(x)$ (we call it *generating function* of $\mathcal{C}$) with all its coordinate functions $g_1(x), \ldots, g_n(x)$ specified in a constructive way and computed with a polynomial (in n) time complexity; π_1, π_2 and σ_1, σ_2 are, respectively, permutation and negation operations (we call them *key parameters* of $\mathcal{C}$); and $K_2 = \{f^{-1} : f \in K_1\}$ — the set of private keys. In the case of $X = Y = \mathbb{F}_2^n$ and $K_1 = K_n(g)$, we call $\mathcal{C}$ a *universal* ACBF.

In $\mathcal{C}$, as in any asymmetric cipher, a public key f is used to encrypt a plaintext x and the private key f^{-1} — to decrypt the corresponding ciphertext y, namely: $y = f(x)$ and $x = f^{-1}(y)$ for $x \in X$, $y \in Y$, $f \in K_1$, $f^{-1} \in K_2$. Also, in $\mathcal{C}$, as in any digital signature scheme with appendix (the signed message), a private key f^{-1} is used to sign a message x and the public key f — to verify signatures, namely: the signature for a message x is $s = f^{-1}(x)$ and a signature s on a message x is valid iff $f(s) = x$.

To provide the necessary property of ease (polynomial time complexity) of computing the functions f and f^{-1}, the generating function g itself and its inverse g^{-1} should have this property too. In this case, the values $y = f(x) = \pi_2(g^{\sigma_2}(\pi_1(x^{\sigma_1})))$ and $x = f^{-1}(y) = [\pi_1^{-1}(g^{-1}((\pi_2^{-1}(y))^{\sigma_2}))]^{\sigma_1}$ would be computed with a polynomial complexity. The polynomial computational complexity of each coordinate in generating function g guarantees a polynomial complexity of computing g itself. This is true if, for example, every coordinate function $g_i(x)$ essentially depends on some $s_i \leqslant s_0$ variables $x_{i_1}, \ldots, x_{i_{s_i}}$ from the string x, that is, $g_i(x) = h_i(x_{i_1}, \ldots, x_{i_{s_i}})$ for a function $h_i : \mathbb{F}_2^{s_i} \to \mathbb{F}_2$ and s_0 is a small enough integer, $1 \leqslant s_0 \leqslant n$. As for providing a polynomial complexity of computing the function g^{-1}, there are many ways to choose g in $\mathcal{C}$ preserving its polynomial complexity in g^{-1}. One of them is the following: $g(x) = g^{(1)}(x)\ldots g^{(r)}(x)$, $1 \leqslant r \leqslant n$, $g^{(i)}(x) = g_{i_1}(x)\ldots g_{i_{s_i}}(x) = h_{i_1}(x^{(i)})\ldots h_{i_{s_i}}(x^{(i)}) = h^{(i)}(x^{(i)})$, $x^{(i)} = x_{i_1}\ldots x_{i_{s_i}}$, $h^{(i)} : \mathbb{F}_2^{s_i} \to \mathbb{F}_2^{s_i}$ is a bijection, $s_1 + s_2 + \ldots + s_r = n$, $i \neq j \Rightarrow \{i_1, \ldots, i_{s_i}\} \cap \{j_1, \ldots, j_{s_j}\} = \varnothing$, $i, j \in \{1, 2, \ldots, r\}$. In this case, $g(x) = h^{(1)}(x^{(1)})\ldots h^{(r)}(x^{(r)})$ and if $y = g(x)$, then

$y^{(i)} = y_{i_1} \dots y_{i_{s_i}} = h^{(i)}(x^{(i)})$, $h^{(i)^{-1}}(y^{(i)}) = x^{(i)}$, $g^{-1}(y) = h^{(1)^{-1}}(y^{(1)}) \dots h^{(r)^{-1}}(y^{(r)})$. That is, $g^{-1}(y)$ is computed with a polynomial complexity.

The security of ACBF $\mathcal{C}$ is based on the difficulty of inverting large bijective vector Boolean functions, that is, of computing $x = f^{-1}(y)$ for $y = f(x)$. For an opponent or cryptanalyst, who (this is believed) doesn't know the values of key parameters π_1, π_2, σ_1 and σ_2 in f, this problem is really difficult one with an exponential time complexity of decision algorithm.

2. Cryptanalysis problem

The cryptanalysis problem that we study for ACBF $\mathcal{C}$ in the paper is the secret key recovery, assuming some plaintexts or messages and the corresponding ciphertexts or signatures are known. If $\mathcal{C}$ is a cipher, the problem is set as follows: given $f(x) \in K_1$, $P_l \in X$, and $C_l = f(P_l)$, $l = 1, 2, \dots, m$, compute $f^{-1}(y)$. Otherwise, that is, if $\mathcal{C}$ is a signature scheme, the problem looks like the following: given $f(x) \in K_1$, $M_l \in X$, and $S_l = f^{-1}(M_l)$, $l = 1, 2, \dots, m$, compute $f^{-1}(x)$. Further, the problem in the first case is called the cipher cryptanalysis, in the second case — the signature cryptanalysis. Any methods solving them are called attacks on cipher and on signature scheme respectively. Aiming to recover the secret key, they are total break attacks. Besides, when we say that the public key $f(x)$ is given, we suppose that everybody has a possibility to compute its value at any point x for a polynomial time, but no cryptanalyst (opponent) knows the parameters $\pi_1, \pi_2, \sigma_1, \sigma_2$ of $f(x)$ (compare, for example, with g^a over $\mathbb{Z}_p$ in ElGamal cipher).

Below we describe some attacks on ciphers and on signature schemes of a universal ACBF $\mathcal{C}$ and of its particular derivatives which are ACBF obtained from $\mathcal{C}$ by replacing some key parameters with the identity operation 1. Here is a more correct definition of this concept. Let $I = \{\pi_1, \pi_2, \sigma_1, \sigma_2\}$, $J \subset I$, and $\mathcal{C}(J) = (X, K(J), Y)$ where $X = Y = \mathbb{F}_2^n$, $K(J) = K_1(J) \times K_2(J)$, $K_2(J) = \{f^{-1}(x) : f(x) \in K_1(J)\}$, $K_1(J) = K_n(g, J)$, and $K_n(g, J)$ is the set of all functions $f(x) = \pi_2(g^{\sigma_2}(\pi_1(x^{\sigma_1})))$ such that $\pi_1, \pi_2 \in \mathbb{S}_n$, $\sigma_1, \sigma_2 \in \mathbb{F}_2^n$, and each key parameter from $I \setminus J$ is the identity. By the definition, $K_n(g, J) \subseteq K_n(g)$, therefore $\mathcal{C}(J)$ is really an ACBF; $K_n(g, I) = K_n(g)$, therefore $\mathcal{C}(I) = \mathcal{C}$; $K_n(g, \varnothing) = \{g(x)\}$, therefore $\mathcal{C}(\varnothing)$ is a trivial cryptosystem. We call $\mathcal{C}(J)$ a *particular derivative* of the $\mathcal{C}$ if $\varnothing \neq J \neq I$. So, for any universal ACBF $\mathcal{C}$, we have 14 particular derivatives $\mathcal{C}(J)$ in total.

Note that for any vector-columns a, σ in $\mathbb{F}_2^n$ and a permutation $\pi = (i_1 i_2 \dots i_n) \in \mathbb{S}_n$, if $c = \neg\sigma$, $\mathrm{w}(a)$ is the weight of a, that is the number of 1's in a, and $T = (t_{kj})$ is a permutation matrix of order n over $\mathbb{F}_2$ where $t_{kj} = 1 \Leftrightarrow j = i_k$ for all $k, j \in \{1, 2, \dots, n\}$, then $a^\sigma = a \oplus c$, $\pi(a) = Ta$, and $\mathrm{w}(Ta) = \mathrm{w}(a)$. Further, we use these facts without additional explanations and call T the matrix of the permutation π. Moreover, we introduce the following notation: A and D are the matrices of permutations π_1 and π_2 respectively and b and d are the vector-columns $\neg\sigma_1$ and $\neg\sigma_2$ respectively. This allows us to apply the symbols of variables A, D, b, d instead of symbols of operations $\pi_1, \pi_2, \sigma_1, \sigma_2$ respectively in the sets I, J as well as in the formulas for $f(x), f^{-1}(x)$ and so on. The fact of such replacement is denoted by the sign $\simeq$. For example, $\{\pi_1, \sigma_2\} \simeq \{A, d\}$.

3. General scheme of attack

Here is the general scheme of an attack on the cipher in an ACBF $\mathcal{C}(J)$.

1. Express the function $f^{-1}(y)$ by a formula in the set of variables and operations

$$J \cup \{y, g, \oplus, \cdot, ^{-1}\}.$$

2. Record the system of equations E expressing the dependence of variables from J on the values P_l, C_l, $1 \leqslant l \leqslant n$, by means of operations $\oplus, \cdot, ^{-1}$ and function g.
3. Solve the system E in unknowns from J using a proper method [3].
4. Substitute the variables from J in formula for $f^{-1}(y)$ by their values from the solution of the system E. The resulting formula is the result of the attack.
5. Estimate the computational complexity of the attack as a time complexity of solving the system of equations E.

The description of the general scheme of an attack on the signature scheme in an ACBF $\mathcal{C}(J)$ is obtained from this scheme for ciphers by substitution $f^{-1}(x)$, M_l, and S_l for $f^{-1}(y)$, C_l, and P_l respectively. The attacks, described below, on universal ACBF $\mathcal{C} = \mathcal{C}(I)$ and on its particular derivatives $\mathcal{C}(J)$ are constructed according to the general scheme. In the case of nonlinear equations in the system E, this system in them is solved by the method of linearization set [3, 4] (further we call it briefly method of LS). The vector weight invariance related to multiplying by a permutation matrix is used to decrease the computational complexity of some of these attacks in practice.

4. Attacks on universal ACBF

4.1. Attack on cipher

We have $y = f(x) = \pi_2(g^{\sigma_2}(\pi_1(x^{\sigma_1}))) = D(g(A(x \oplus b)) \oplus d)$; $D^{-1}y \oplus d = g(A(x \oplus b))$, $A^{-1}(g^{-1}(D^{-1}y \oplus d)) \oplus b = x$, $f^{-1}(y) = x$, and $C_l = f(P_l)$, $l = 1, 2, \ldots, m$. Hence,

$$f^{-1}(y) = A^{-1}g^{-1}(D^{-1}y \oplus d) \oplus b$$

where (D^{-1}, d, b, A) is a solution of the system of equations

$$D^{-1}C_l \oplus d = g(A(P_l \oplus b)), \quad l = 1, 2, \ldots, m,$$

which is solved by the method of LS, namely by assigning in turn the different values to the variables A, b and solving the resulting system of linear equations with unknowns in D^{-1} and d. So, the computational complexity of the attack is $\mathrm{O}(n!2^n)$.

4.2. Attack on signature scheme

We have $S_l = f^{-1}(M_l) = A^{-1}g^{-1}(D^{-1}M_l \oplus d) \oplus b$, $l = 1, 2, \ldots, m$, and

$$f^{-1}(x) = A^{-1}g^{-1}(D^{-1}x \oplus d) \oplus b$$

where (D^{-1}, d, b, A) is a solution of the system of equations

$$D^{-1}M_l \oplus d = g(A(S_l \oplus b)), \quad l = 1, 2, \ldots, m,$$

which is solved by the method of LS, that is, by assigning in turn the different values to the variables A, b and solving the resulting system of linear equations with unknowns D^{-1} and d. So, the computational complexity of the attack is $\mathrm{O}(n!2^n)$.

5. Attacks on particular derivatives of universal ACBF

5.1. Attacks on ciphers

Given $J \subset \{\pi_1, \pi_2, \sigma_1, \sigma_2\} \simeq \{A, D, b, d\}$, $f(x) \in K_n(g, J)$, $P_l \in \mathbb{F}_2^n$, and $C_l = f(P_l)$, $l = 1, 2, \ldots, m$, compute $f^{-1}(y)$. Consider the possible cases.

1. $J = \{\sigma_1\} \simeq \{b\}$.

We have $y = f(x) = g(x^{\sigma_1}) = g(x \oplus b)$ and $b = \neg\sigma_1$, therefore $g^{-1}(y) = x \oplus b$ and $f^{-1}(y) = x$. Hence,

$$f^{-1}(y) = g^{-1}(y) \oplus b$$

where

$$b = P_l \oplus g^{-1}(C_l), \quad l = 1, 2, \ldots, m.$$

So, $b = P_1 \oplus g^{-1}(C_1)$. Computational complexity of this attack is a polynomial in n.

2. $J = \{\pi_1\} \simeq \{A\}$.

Here, $y = f(x) = g(\pi_1(x)) = g(Ax)$ and A is the matrix of π_1; $Ax = g^{-1}(y)$, $x = A^{-1}g^{-1}(y)$, $f^{-1}(y) = x$. Hence,

$$f^{-1}(y) = A^{-1}g^{-1}(y),$$

and the matrix A is a solution of the system of linear equations

$$AP_l = g^{-1}(C_l), \quad l = 1, 2, \ldots, m,$$

which is solved for a polynomial time.

In case $m = 1$, this system has $r = w!(n - w)!$ of solutions where $w = \mathrm{w}(P_1)$ and r is the product of permutation quantities for ones and for zeros in P_1. The computational complexity of this attack coincides the time complexity of solving the system of linear equations problem and does not exceed a polynomial in n.

3. $J = \{\pi_1, \sigma_1\} \simeq \{A, b\}$.

In this case, $y = f(x) = g(\pi_1(x^{\sigma_1})) = g(A(x \oplus b))$; $g^{-1}(y) = A(x \oplus b)$, $\mathrm{w}(x \oplus b) = \mathrm{w}(g^{-1}(y))$, $A^{-1}g^{-1}(y) \oplus b = x$, $f^{-1}(y) = x$.

Define $B_l \subseteq \mathbb{F}_2^n$ by induction on $l = 1, 2, \ldots, m$, namely let $B_1 = \{\underline{b} : \mathrm{w}(P_1 \oplus \underline{b}) = \mathrm{w}(g^{-1}(C_1))\}$ and $B_l = \{\underline{b} : \underline{b} \in B_{l-1}, \mathrm{w}(P_l \oplus \underline{b}) = \mathrm{w}(g^{-1}(C_l))\}$, $2 \leqslant l \leqslant m$. Then

$$f^{-1}(y) = A^{-1}g^{-1}(y) \oplus b$$

where (A, b) is a solution of the following system of equations

$$A(P_l \oplus b) = g^{-1}(C_l), \quad l = 1, 2, \ldots, m,$$

which is solved, using the method of LS, by assigning in turn the different values from the set B_m to the variable b and solving the resulting system of linear equations with unknowns in A. The computational complexity of the attack is $\mathrm{O}(\mathrm{C}_n^{n/2})$.

4. $J = \{\sigma_2\} \simeq \{d\}$.

$y = f(x) = g^{\sigma_2}(x) = g(x) \oplus d$ and $d = \neg\sigma_2$; $g^{-1}(y \oplus d) = x$, $f^{-1}(y) = x$. Hence,

$$f^{-1}(y) = g^{-1}(y \oplus d)$$

where

$$d = C_l \oplus g(P_l), \quad l = 1, 2, \ldots, m.$$

So, $d = C_1 \oplus g(P_1)$. Computational complexity of the attack is a polynomial in n.

5. $J = \{\sigma_1, \sigma_2\} \simeq \{b, d\}$.

$y = f(x) = g^{\sigma_2}(x^{\sigma_1}) = g(x \oplus b) \oplus d$ where $b = \neg\sigma_1$, $d = \neg\sigma_2$; $x \oplus b = g^{-1}(y \oplus d)$, $g^{-1}(y \oplus d) \oplus b = x$, $f^{-1}(y) = x$. Hence,

$$f^{-1}(y) = g^{-1}(y \oplus d) \oplus b$$

where (b, d) is a solution of the following system of equations

$$g(P_l \oplus b) \oplus d = C_l, \quad l = 1, 2, \ldots, m,$$

and can be computed by the method of LS, that is, by assigning in turn the different values to b and solving the resulting system of linear equations with unknowns in d. The complexity of this attack is $\mathrm{O}(2^n)$.

6. $J = \{\pi_1, \sigma_2\} \simeq \{A, d\}$.

$y = f(x) = g^{\sigma_2}(\pi_1(x)) = g(Ax) \oplus d$; $Ax = g^{-1}(y \oplus d)$, $\mathrm{w}(x) = \mathrm{w}(g^{-1}(y \oplus d))$, $A^{-1}g^{-1}(y \oplus \oplus d) = x$, $f^{-1}(y) = x$.

Define $D_l \subseteq \mathbb{F}_2^n$ by induction on $l = 1, 2, \ldots, m$, namely let $D_1 = \{\underline{d} : \mathrm{w}(P_1) = = \mathrm{w}(g^{-1}(C_1 \oplus \underline{d}))\}$ and $D_l = \{\underline{d} : \underline{d} \in D_{l-1}, \mathrm{w}(P_l) = \mathrm{w}(g^{-1}(C_l \oplus \underline{d}))\}$, $2 \leqslant l \leqslant m$. Then

$$f^{-1}(y) = A^{-1}g^{-1}(y \oplus d)$$

where (A, d) is a solution of the following system of equations

$$AP_l = g^{-1}(C_l \oplus d), \quad l = 1, 2, \ldots, m.$$

This system is solved, using the method of LS, by assigning in turn the different values from the set D_m to the variable d and solving the resulting system of linear equations in unknowns in A. The computational complexity of the attack is $\mathrm{O}(\mathrm{C}_n^{n/2})$.

7. $J = \{\pi_1, \sigma_1, \sigma_2\} \simeq \{A, b, d\}$.

$y = f(x) = g^{\sigma_2}(\pi_1(x^{\sigma_1})) = g(A(x \oplus b)) \oplus d$; $A(x \oplus b) = g^{-1}(y \oplus d)$, $\mathrm{w}(x \oplus b) = \mathrm{w}(g^{-1}(y \oplus d))$, $A^{-1}g^{-1}(y \oplus d) = x \oplus b$, $f^{-1}(y) = x$.

For each $\underline{d} \in \mathbb{F}_2^n$, define $B_l(\underline{d}) \subseteq \mathbb{F}_2^n$ by induction on $l = 1, 2, \ldots, m$, namely let $B_1(\underline{d}) = = \{\underline{b} : \mathrm{w}(P_1 \oplus \underline{b}) = \mathrm{w}(g^{-1}(C_1 \oplus \underline{d}))\}$ and $B_l(\underline{d}) = \{\underline{b} : \underline{b} \in B_{l-1}(\underline{d}), \mathrm{w}(P_l \oplus \underline{b}) = \mathrm{w}(g^{-1}(C_l \oplus \underline{d}))\}$, $2 \leqslant l \leqslant m$. Then

$$f^{-1}(y) = A^{-1}g^{-1}(y \oplus d) \oplus b$$

where (A, b, d) is a solution of the following system of equations

$$A(P_l \oplus b) = g^{-1}(C_l \oplus d), \quad l = 1, 2, \ldots, m.$$

This system is solved, using the method of LS, by assigning in turn the different values $(\underline{b}, \underline{d})$ with $\underline{d}$ from $\mathbb{F}_2^n$ and $\underline{b}$ from $B_m(\underline{d})$ to the pair of variables (b, d) and solving the resulting system of linear equations in unknowns in A. The computational complexity of the attack is $\mathrm{O}(2^n \mathrm{C}_n^{n/2})$.

8. $J = \{\pi_2\} \simeq \{D\}$.

$y = f(x) = \pi_2(g(x)) = Dg(x)$ where D is the matrix of the permutation π_2; $D^{-1}y = = g(x)$, $g^{-1}(D^{-1}y) = x$, $f^{-1}(y) = x$. Hence,

$$f^{-1}(y) = g^{-1}(D^{-1}y)$$

where D^{-1} is a solution of the system of linear equations

$$D^{-1}C_l = g(P_l), \quad l = 1, 2, \ldots, m.$$

Computational complexity of this attack is a polynomial in n.

9. $J = \{\pi_2, \sigma_1\} \simeq \{D, b\}$.

$y = f(x) = \pi_2(g(x^{\sigma_1})) = Dg(x \oplus b)$; $D^{-1}y = g(x \oplus b)$, $\mathrm{w}(y) = \mathrm{w}(g(x \oplus b))$, $g^{-1}(D^{-1}y) \oplus \oplus b = x$, $f^{-1}(y) = x$.

Define $B_l \subseteq \mathbb{F}_2^n$ by induction on $l = 1, 2, \ldots, m$, namely let $B_1 = \{\underline{b} : \mathrm{w}(C_1) = \mathrm{w}(g(P_1 \oplus \oplus \underline{b}))\}$ and $B_l = \{\underline{b} : \underline{b} \in B_{l-1}, \mathrm{w}(C_l) = \mathrm{w}(g(P_l \oplus \underline{b}))\}$, $2 \leqslant l \leqslant m$. Then

$$f^{-1}(y) = g^{-1}(D^{-1}y) \oplus b$$

where (D^{-1}, b) is a solution of the system of equations

$$D^{-1}C_l = g(P_l \oplus b), \quad l = 1, 2, \ldots, m,$$

which is solved, using the method of LS, by assigning in turn the different values from B_m to the variable b and solving the resulting system of linear equations with unknowns in D^{-1}. So, the computational complexity of the attack is $\mathrm{O}(\mathrm{C}_n^{n/2})$.

10. $J = \{\pi_1, \pi_2\} \simeq \{A, D\}$.

$y = f(x) = \pi_2(g(\pi_1(x))) = Dg(Ax)$; $D^{-1}y = g(Ax)$, $A^{-1}g^{-1}(D^{-1}y) = x$, $f^{-1}(y) = x$. Hence,

$$f^{-1}(y) = A^{-1}g^{-1}(D^{-1}y)$$

where (A, D^{-1}) is a solution of the system of equations

$$D^{-1}C_l = g(AP_l), \quad l = 1, 2, \ldots, m,$$

which is solved by the method of LS, that is, by assigning in turn the different values to the variable A and solving the resulting system of linear equations with unknowns in D^{-1}. So, the computational complexity of the attack is $\mathrm{O}(n!)$.

11. $J = \{\pi_1, \pi_2, \sigma_1\} \simeq \{A, D, b\}$.

$y = f(x) = \pi_2(g(\pi_1(x^{\sigma_1}))) = Dg(A(x \oplus b))$; $D^{-1}y = g(A(x \oplus b))$, $A^{-1}g^{-1}(D^{-1}y) = x \oplus b$, $f^{-1}(y) = x$. Hence,

$$f^{-1}(y) = A^{-1}g^{-1}(D^{-1}y) \oplus b$$

where (A, D^{-1}, b) is a solution of the system of equations

$$A^{-1}g^{-1}(D^{-1}C_l) \oplus b = P_l, \quad l = 1, 2, \ldots, m,$$

which is solved by the method of LS, that is, by assigning in turn the different values to the variable D^{-1} and solving the resulting system of linear equations with unknowns in A, b. The computational complexity of the attack is $\mathrm{O}(n!)$.

12. $J = \{\pi_2, \sigma_2\} \simeq \{D, d\}$.

$y = f(x) = \pi_2(g^{\sigma_2}(x)) = D(g(x) \oplus d)$; $D^{-1}y = g(x) \oplus d$, $g^{-1}(D^{-1}y \oplus d) = x$, $f^{-1}(y) = x$. Hence,

$$f^{-1}(y) = g^{-1}(D^{-1}y \oplus d)$$

where (D^{-1}, d) is a solution of the system of linear equations

$$D^{-1}C_l \oplus d = g(P_l), \quad l = 1, 2, \ldots, m,$$

which is solved with a polynomial complexity.

13. $J = \{\pi_2, \sigma_1, \sigma_2\} \simeq \{D, b, d\}$.

$y = f(x) = \pi_2(g^{\sigma_2}(x^{\sigma_1})) = D(g(x \oplus b) \oplus d)$; $D^{-1}y = g(x \oplus b) \oplus d$, $g^{-1}(D^{-1}y \oplus d) = x \oplus b$, $f^{-1}(y) = x$. Hence,

$$f^{-1}(y) = g^{-1}(D^{-1}y \oplus d) \oplus b$$

where (D^{-1}, b, d) is a solution of the system of equations

$$D^{-1}C_l \oplus d = g(P_l \oplus b), \quad l = 1, 2, \ldots, m,$$

which is solved by the method of LS, that is, by assigning in turn the different values to the variable b and solving the resulting system of linear equations with unknowns in D^{-1} and d. So, the computational complexity of the attack is $\mathrm{O}(2^n)$.

14. $J = \{\pi_1, \pi_2, \sigma_2\} \simeq \{A, D, d\}$.

$y = f(x) = \pi_2(g^{\sigma_2}(\pi_1(x))) = D(g(Ax) \oplus d)$; $D^{-1}y = g(Ax) \oplus d$, $A^{-1}g^{-1}(D^{-1}y \oplus d) = x$, $f^{-1}(y) = x$. Hence,

$$f^{-1}(y) = A^{-1}g^{-1}(D^{-1}y \oplus d)$$

where (D^{-1}, d, A) is a solution of the system of equations

$$D^{-1}C_l \oplus d = g(AP_l), \quad l = 1, 2, \ldots, m,$$

which is solved by the method of LS, that is, by assigning in turn the different values to the variable A and solving the resulting system of linear equations with unknowns in D^{-1} and d. So, the computational complexity of the attack is $\mathrm{O}(n!)$.

5.2. Attacks on signature schemes

Given $J \subset \{\pi_1, \pi_2, \sigma_1, \sigma_2\} \simeq \{A, D, b, d\}$, $f(x) \in K_n(g, j)$, $M_l \in \mathbb{F}_2^n$, and $S_l = f^{-1}(M_l)$, $l = 1, 2, \ldots, m$, compute $f^{-1}(x)$. Consider the possible cases. As mentioned above, in every case the attack on a signature scheme differs from the attack on a cipher just given by only using variables x, M_l, and S_l instead of y, C_l, and P_l respectively. Taking into account that the attacks on the signature schemes have the great and distinctive significance for cryptography, we describe them without abbreviations.

1. $J = \{\sigma_1\} \simeq \{b\}$.

In this case, $S_l = g^{-1}(M_l) \oplus b$, $l = 1, 2, \ldots, m$, and

$$f^{-1}(x) = g^{-1}(x) \oplus b$$

where

$$b = S_l \oplus g^{-1}(M_l), \quad l = 1, 2, \ldots, m.$$

The computational complexity of the attack is a polynomial in n.

2. $J = \{\pi_1\} \simeq \{A\}$.

$S_l = A^{-1}g^{-1}(M_l)$, $l = 1, 2, \ldots, m$, and

$$f^{-1}(x) = A^{-1}g^{-1}(x)$$

where A is a solution of the system of linear equations

$$AS_l = g^{-1}(M_l), \quad l = 1, 2, \ldots, m,$$

which is solved for a polynomial time.

3. $J = \{\pi_1, \sigma_1\} \simeq \{A, b\}$.

$S_l = A^{-1}g^{-1}(M_l) \oplus b$, $A(S_l \oplus b) = g^{-1}(M_l)$, $\mathrm{w}(S_l \oplus b) = \mathrm{w}(g^{-1}(M_l))$, $l = 1, 2, \ldots, m$, $B_1 = \{\underline{b} : \mathrm{w}(S_1 \oplus \underline{b}) = \mathrm{w}(g^{-1}(M_1))\}$, and $B_l = \{\underline{b} : \underline{b} \in B_{l-1}, \mathrm{w}(S_l \oplus \underline{b}) = \mathrm{w}(g^{-1}(M_l))\}$, $2 \leqslant l \leqslant m$. So,

$$f^{-1}(x) = A^{-1}g^{-1}(x) \oplus b$$

where (A, b) is a solution of the system of equations

$$A(S_l \oplus b) = g^{-1}(M_l), \quad l = 1, 2, \ldots, m,$$

which is solved by the method of LS, that is, by assigning in turn the different values from B_m to the variable b and solving the resulting system of linear equations with unknowns in A. The computational complexity of the attack is $\mathrm{O}(\mathrm{C}_n^{n/2})$.

4. $J = \{\sigma_2\} \simeq \{d\}$.

$S_l = g^{-1}(M_l \oplus d)$, $l = 1, 2, \ldots, m$, and

$$f^{-1}(x) = g^{-1}(x \oplus d)$$

where

$$d = M_l \oplus g(S_l), \quad l = 1, 2, \ldots, m.$$

Computational complexity of the attack is a polynomial in n.

5. $J = \{\sigma_1, \sigma_2\} \simeq \{b, d\}$.

$S_l = g^{-1}(M_l \oplus d) \oplus b$, $l = 1, 2, \ldots, m$, and

$$f^{-1}(x) = g^{-1}(x \oplus d) \oplus b$$

where (b, d) is a solution of the following system of equations

$$g(S_l \oplus b) = M_l \oplus d, \quad l = 1, 2, \ldots, m,$$

which is solved, using the method of LS, that is, by assigning in turn the different values to the variable b and solving the resulting system of linear equations with unknowns in d. The computational complexity of the attack is $\mathrm{O}(2^n)$.

6. $J = \{\pi_1, \sigma_2\} \simeq \{A, d\}$.

$S_l = A^{-1}g^{-1}(M_l \oplus d)$, $A(S_l \oplus d) = g^{-1}(M_l)$, $\mathrm{w}(S_l \oplus d) = \mathrm{w}(g^{-1}(M_l))$, $l = 1, 2, \ldots, m$, $D_1 = \{\underline{d} : \mathrm{w}(S_1 \oplus \underline{d}) = \mathrm{w}(g^{-1}(M_1))\}$, and $D_l = \{\underline{d} : \underline{d} \in D_{l-1}, \mathrm{w}(S_l \oplus \underline{d}) = \mathrm{w}(g^{-1}(M_l))\}$, $2 \leqslant l \leqslant m$. So,

$$f^{-1}(x) = A^{-1}g^{-1}(x \oplus d)$$

where (A, d) is a solution of the system of equations

$$AS_l = g^{-1}(M_l \oplus d), \quad l = 1, 2, \ldots, m,$$

which is solved, using the method of LS, that is, by assigning in turn the different values from D_m to the variable d and solving the resulting system of linear equations with unknowns in A. The computational complexity of the attack is $\mathrm{O}(\mathrm{C}_n^{n/2})$.

7. $J = \{\pi_1, \sigma_1, \sigma_2\} \simeq \{A, b, d\}$.

$S_l = A^{-1}g^{-1}(M_l \oplus d) \oplus b$, $A(S_l \oplus b) = g^{-1}(M_l \oplus d)$, $\mathrm{w}(S_l \oplus b) = \mathrm{w}(g^{-1}(M_l \oplus d))$, $l = 1, 2, \ldots, m$, $B_1(\underline{d}) = \{\underline{b} : \mathrm{w}(S_1 \oplus \underline{b}) = \mathrm{w}(g^{-1}(M_1 \oplus \underline{d}))\}$, and $B_l(\underline{d}) = \{\underline{b} : \underline{b} \in B_{l-1}(\underline{d}), \mathrm{w}(S_l \oplus \underline{b}) = \mathrm{w}(g^{-1}(M_l \oplus \underline{d}))\}$, $\underline{d} \in \mathbb{F}_2^n$, $2 \leqslant l \leqslant m$. So,

$$f^{-1}(x) = A^{-1}g^{-1}(x \oplus d) \oplus b,$$

where (A, b, d) is a solution of the system of equations

$$A(S_l \oplus b) = g^{-1}(M_l \oplus d), \quad l = 1, 2, \ldots, m,$$

which is solved, using the method of LS, that is, by assigning in turn the different values $(\underline{b}, \underline{d})$ with $\underline{d}$ from $\mathbb{F}_2^n$ and $\underline{b}$ from $B_m(\underline{d})$ to the pair of variables (b, d) and solving the resulting system of linear equations with unknowns in A. The computational complexity of the attack is $\mathrm{O}(2^n \mathrm{C}_n^{n/2})$.

8. $J = \{\pi_2\} \simeq \{D\}$.

$S_l = g^{-1}(D^{-1}M_l)$, $l = 1, 2, \ldots, m$, and

$$f^{-1}(x) = g^{-1}(D^{-1}x)$$

where D^{-1} is a solution of the system of linear equations

$$D^{-1}M_l = g(S_l), \quad l = 1, 2, \ldots, m.$$

Computational complexity of this attack is a polynomial in n.

9. $J = \{\pi_2, \sigma_1\} \simeq \{D, b\}$.

$S_l = g^{-1}(D^{-1}M_l) \oplus b$, $g(S_l \oplus b) = D^{-1}M_l$, $\mathrm{w}(M_l) = \mathrm{w}(g(S_l \oplus b))$, $l = 1, 2, \ldots, m$, $B_1 = \{\underline{b} : \mathrm{w}(M_1) = \mathrm{w}(g(S_1 \oplus \underline{b}))\}$, and $B_l = \{\underline{b} : \underline{b} \in B_{l-1}, \mathrm{w}(M_l) = \mathrm{w}(g(S_l \oplus \underline{b}))\}$, $2 \leqslant l \leqslant m$. So,

$$f^{-1}(x) = g^{-1}(D^{-1}x) \oplus b$$

where (D^{-1}, b) is a solution of the system of equations

$$D^{-1}M_l = g(S_l \oplus b), \quad l = 1, 2, \ldots, m,$$

which is solved, using the method of LS, that is, by assigning in turn the different values from B_m to the variable b and solving the resulting system of linear equations with unknowns in D^{-1}. The computational complexity of the attack is $\mathrm{O}(\mathrm{C}_n^{n/2})$.

10. $J = \{\pi_1, \pi_2\} \simeq \{A, D\}$.

$S_l = A^{-1}g^{-1}(D^{-1}M_l)$, $l = 1, 2, \ldots, m$, and

$$f^{-1}(x) = A^{-1}g^{-1}(D^{-1}x)$$

where (A, D^{-1}) is a solution of the system of equations

$$D^{-1}M_l = g(AS_l), \quad l = 1, 2, \ldots, m,$$

which is solved, using the method of LS, that is, by assigning in turn the different values to the variable A and solving the resulting system of linear equations with unknowns in D^{-1}. So, the computational complexity of the attack is $\mathrm{O}(n!)$.

11. $J = \{\pi_1, \pi_2, \sigma_1\} \simeq \{A, D, b\}$.

$S_l = A^{-1}g^{-1}(D^{-1}M_l) \oplus b$, $l = 1, 2, \ldots, m$, and

$$f^{-1}(x) = A^{-1}g^{-1}(D^{-1}x) \oplus b$$

where (A, D^{-1}, b) is a solution of the system of equations

$$A^{-1}g^{-1}(D^{-1}M_l) \oplus b = S_l, \quad l = 1, 2, \ldots, m,$$

which is solved, using the method of LS, that is, by assigning in turn the different values to the variables D^{-1} and solving the resulting system of linear equations with unknowns in A, b. The computational complexity of the attack is $\mathrm{O}(n!)$.

12. $J = \{\pi_2, \sigma_2\} \simeq \{D, d\}$.
$S_l = g^{-1}(D^{-1}M_l \oplus d)$, $l = 1, 2, \ldots, m$, and

$$f^{-1}(x) = g^{-1}(D^{-1}x \oplus d)$$

where (D^{-1}, d) is a solution of the system of equations

$$D^{-1}M_l \oplus d = g(S_l), \quad l = 1, 2, \ldots, m,$$

which is solved with a polynomial complexity.

13. $J = \{\pi_2, \sigma_1, \sigma_2\} \simeq \{D, b, d\}$.
$S_l = g^{-1}(D^{-1}M_l \oplus d) \oplus b$, $l = 1, 2, \ldots, m$, and

$$f^{-1}(x) = g^{-1}(D^{-1}x \oplus d) \oplus b$$

where (D^{-1}, d, b) is a solution of the system of equations

$$D^{-1}M_l \oplus d = g(S_l \oplus b), \quad l = 1, 2, \ldots, m,$$

which is solved, using the method of LS, by assigning in turn the different values to the variable b and solving the resulting system of linear equations with unknowns in D^{-1} and d. So, the computational complexity of the attack is $\mathrm{O}(2^n)$.

14. $J = \{\pi_1, \pi_2, \sigma_2\} \simeq \{A, D, d\}$. $S_l = A^{-1}g^{-1}(D^{-1}M_l \oplus d)$, $l = 1, 2, \ldots, m$, and

$$f^{-1}(x) = A^{-1}g^{-1}(D^{-1}x \oplus d)$$

where (D^{-1}, d, A) is a solution of the system of equations

$$D^{-1}M_l \oplus d = g(AS_l), \quad l = 1, 2, \ldots, m,$$

which is solved, using the method of LS, by assigning in turn the different values to the variable A and solving the resulting system of linear equations with unknowns in D^{-1} and d. So, the computational complexity of the attack is $\mathrm{O}(n!)$.

Conclusion

What we have discussed above in the paper should be considered as a step in the process of developing the theory of public key cryptosystems based on the bijective systems of Boolean functions. There are many problems we need solve on this way. Some of them are the following: 1) generating pseudorandom invertible systems of Boolean functions depending on a covered parameter such that the system is computed and inverted with a polynomial time complexity iff the value of the parameter is known; 2) necessary and sufficient conditions for uniqueness of a private key, with which the given blocks of a ciphertext are decrypted to the given blocks of a plaintext; 3) lower and upper bounds for the number of blocks with this property of the private key.

REFERENCES

1. *Tao R.* Finite Automata and Application to Cryptography. Berlin; Heidelberg, Springer, 2009. 411 p.
2. *Agibalov G. P.* Substitution block ciphers with functional keys. Prikladnaya Diskretnaya Matematika, 2017, no. 38, pp. 57–65.
3. *Agibalov G. P.* Metody resheniya sistem polinomial'nykh uravneniy nad konechnym polem [Methods for solving systems of polynomial equations over a finite field]. Vestnik TSU. Prilozhenie, 2006, no. 17, pp. 4–9. (in Russian)
4. *Agibalov G. P.* Logicheskie uravneniya v kriptoanalize generatorov klyuchevogo potoka [Logical equations in cryptanalysis of key stream generators]. Vestnik TSU. Prilozhenie, 2003, no. 6, pp. 31–41. (in Russian)

UDC 519.7

MATHEMATICAL METHODS IN SOLUTIONS OF THE PROBLEMS PRESENTED AT THE THIRD INTERNATIONAL STUDENTS' OLYMPIAD IN CRYPTOGRAPHY[1]

N. Tokareva[*,**], A. Gorodilova[*,**], S. Agievich[***], V. Idrisova[*,**], N. Kolomeec[*,**], A. Kutsenko[*], A. Oblaukhov[*], G. Shushuev[**]

[*] *Novosibirsk State University, Novosibirsk, Russia,*
[**] *Sobolev Institute of Mathematics, Novosibirsk, Russia,*
[***] *Belarusian State University, Minsk, Belarus*

The mathematical problems, presented at the Third International Students' Olympiad in Cryptography NSUCRYPTO'2016, and their solutions are considered. They are related to the construction of algebraic immune vectorial Boolean functions and big Fermat numbers, the secrete sharing schemes and pseudorandom binary sequences, biometric cryptosystems and the blockchain technology, etc. Two open problems in mathematical cryptography are also discussed and a solution for one of them proposed by a participant during the Olympiad is described. It was the first time in the Olympiad history. The problem is the following: construct $F : \mathbb{F}_2^5 \to \mathbb{F}_2^5$ with maximum possible component algebraic immunity 3 or prove that it does not exist. Alexey Udovenko from University of Luxembourg has found such a function.

Keywords: *cryptography, ciphers, Boolean functions, biometry, blockchain, Olympiad, NSUCRYPTO.*

DOI 10.17223/20710410/40/4

Introduction

The Third International Students' Olympiad in Cryptography — NSUCRYPTO'2016 was held on November 13–21, 2016. NSUCRYPTO is the unique cryptographic Olympiad containing scientific mathematical problems for students and professionals from any country. Its aim is to involve young researchers in solving curious and tough scientific problems of modern cryptography. From the very beginning, the concept of the Olympiad was not to focus on solving olympic tasks but on including unsolved research problems at the intersection of mathematics and cryptography.

The Olympiad consisted of two independent Internet rounds. The First round (duration 4 hours 30 minutes) was individual. It was divided into two sections: A and B. Theoretical problems in mathematics of cryptography were offered to participants. The Second round (duration 1 week) was devoted to research and programming problems of cryptography solved in teams. Anyone who wanted to try his/her hand in solving cryptographic problems were able to become a participant. During the registration every participant had to choose corresponding category: "School Student" (for junior researchers: pupils and school students), "University Student" (for participants who were currently studying at universities) or "Professional" (for participants who had already completed education, are PhD students, or just wanted to be in the restriction-free category). The winners

[1]The work was supported by Russian Ministry of Science and Education under the 5-100 Excellence Programme, RMC NSU, and by the Russian Foundation for Basic Research (projects no. 15-07-01328, 17-41-543364).

were awarded in each category separately. The language of the Olympiad is English. All information about organization and rules of the Olympiad can be found on the official website at `www.nsucrypto.nsu.ru`.

In 2016 the geography of participants has expanded significantly. There were 420 participants from 24 countries: Russia (Novosibirsk, Moscow, Saint Petersburg, Yekaterinburg, Kazan, Saratov, Taganrog, Krasnoyarsk, Petrozavodsk, Perm, Chelyabinsk, Zelenograd, Tomsk, Korolev, Omsk, Ramenskoye, Yaroslavl, Novokuznetsk), Belarus (Minsk), Ukraine (Kiev, Kharkov, Zaporozhye), Kazakhstan (Astana, Almaty, Kaskelen), Kirghizia (Bishkek), Great Britain (Bristol), Bulgaria (Sofia), Germany (Berlin, Munich, Bochum, Witten), France (Paris), Luxembourg (Luxembourg), Hungary (Szeged), Sweden (Gothenburg), Switzerland (Zurich, Bern), Italy (Padova), Czech Republic (Prague), Estonia (Tartu), Spain (Barcelona) Canada (Edmonton), Iran (Tehran), South Africa (Cape Town), China (Beijing), Vietnam (Ho Chi Minh City, Saigon), Indonesia (Bandung), India (Kollam, Haydebarad).

Organizers of the Olympiad are: Novosibirsk State University, Sobolev Institute of Mathematics (Novosibirsk), Tomsk State University, Belarusian State University and University of Leuven (KU Leuven, Belgium).

In section 1 we describe problem structure of the Olympiad according to sections and rounds. Section 2 is devoted to unsolved problems formulated in NSUCRYPTO for all years since 2014, with attention to solutions proposed for two of them. Section 3 contains the conditions of all 16 mathematical problems of NSUCRYPTO'2016. Among them, there are both some amusing tasks based on historical ciphers as well as hard mathematical problems. We consider mathematical problems related to the construction of algebraically immune vectorial Boolean functions and big Fermat numbers, problems about secrete sharing schemes and pseudorandom binary sequences, biometric cryptosystems and the blockchain technology, etc. Some unsolved problems are also discussed. In section 4 we present solutions of all the problems with paying attention to solutions proposed by the participants. Information about the winners is given in section 5.

Mathematical problems of the previous International Olympiads NSUCRYPTO'2015 and NSUCRYPTO'2016 can be found in [1] and [2] respectively.

1. Problem structure of the Olympiad

There were 16 problems on the Olympiad. Some of them were included in both rounds (Tables 1, 2). Thus, section A (school section) of the first round consisted of 6 problems, whereas section B (student section) contained 7 problems. Three problems were common for both sections. In the Table 1 you can see the highest scores one could get in case of solving the problem.

The second round was composed of 12 problems; they were common for all the participants. Two of the problems presented on the second round were marked as unsolved (awarded special prizes from the Program Committee).

Table 1

Problems of the first round

A — school section

N	Problem title	Maximum scores
1	Cipher from the pieces	4
2	Get an access	4
3	Find the key	4
4	Labyrinth	4
5	System of equations	4
6	Biometric pin-code	4

B — student section

N	Problem title	Maximum scores
1	Cipher from the pieces	4
2	Labyrinth	4
3	Quadratic functions	8
4	System of equations	4
5	Biometric key	6
6	Secret sharing	6
7	Protocol	6

Table 2

Problems of the second round

N	Problem title	Maximum scores
1	Algebraic immunity	Unsolved
2	Zerosum at AES	8
3	Latin square	6
4	Nsucoin	10
5	Metrical cryptosystem	6
6	Quadratic functions	8
7	Secret sharing	6
8	Biometric key	6
9	Protocol	6
10	Find the key	4
11	Labyrinth	4
12	Big Fermat numbers	Unsolved

2. Unsolved problems of NSUCRYPTO since the origin

In this section we shortly present all unsolved problems stated in the history of NSUCRYPTO (Table 3), where we mention also the current status of all the problems. The formulations of the problems can be found in [1] (year 2014), [2] (year 2015) as well as the current paper (year 2016).

Table 3

Unsolved Problems of NSUCRYPTO

N	Year	Problem title	Status
1	2014	Watermarking cipher	Unsolved
2	2014	APN permutation	Unsolved
3	2014	Super S-box	Unsolved
4	2015	A secret sharing	Partially SOLVED in [3]
5	2015	Hypothesis	Unsolved
6	2016	Algebraic immunity	SOLVED during the Olympiad
7	2016	Big Fermat numbers	Unsolved

The Olympiad NSUCRYPTO-2016 became particular since there was the first time when an unsolved problem stated was successfully solved by a participant during the Olympiad. Alexey Udovenko (University of Luxembourg) was able to find a solution to the problem “Algebraic immunity” (see section 4.15).

Moreover, we are also very pleased to say that the unsolved problem “A secret sharing” of NSUCRYPTO-2015 was also partially solved! In [3] Kristina Geut, Konstantin

Kirienko, Prokhor Kirienko, Roman Taskin, Sergey Titov (Ural State University of Railway Transport, Yekaterinburg) found a solution for the problem in the case of even dimension (the problem remains open for odd dimension).

The current status of unsolved problems from NSUCRYPTO of all years can be found at `http://nsucrypto.nsu.ru/unsolved-problems/`. Everyone is welcome to propose a solution to any problem stated. Please, send you ideas to `nsucrypto@nsu.ru`.

3. Problems

In this section we formulate all the problems of the Olympiad.

3.1. Problem "Cipher from the pieces"

Recover the original message, splitting the figure (Fig. 1) into equal pieces such that each color occurs once in every piece.

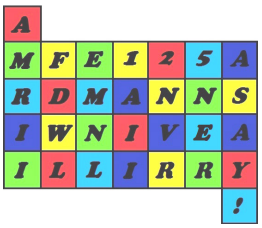

Fig. 1.

3.2. Problem "Get an access"

To get an access to the safe one should put 20 non-negative integers in the following cells (Fig. 2). The safe will be opened if and only if the sum of any two numbers is even number k, such that $4 \leqslant k \leqslant 8$, and each possible sum occurs at least once. Find the sum of all these numbers.

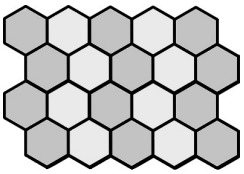

Fig. 2.

3.3. Problem "Find the key"

The key of a cipher is the set of positive integers a, b, c, d, e, f, g, such that the following relation holds:

$$a^3 + b^3 + c^3 + d^3 + e^3 + f^3 + g^3 = 2016^{2017}.$$

Find the key!

3.4. Problem "Labyrinth"

Read the message hidden in the labyrinth (Fig. 3)!

Fig. 3.

3.5. Problem "System of equations"

Analyzing a cipher Caroline gets the following system of equations in binary variables $x_1, x_2, \dots, x_{16} \in \{0, 1\}$ that represent the unknown bits of the secrete key:

$$\begin{cases} x_1x_3 \oplus x_2x_4 = x_5 - x_6, \\ x_{14} \oplus x_{11} = x_{12} \oplus x_{13} \oplus x_{14} \oplus x_{15} \oplus x_{16}, \\ (x_8 + x_9 + x_7)^2 = 2(x_6 + x_{11} + x_{10}), \\ x_{13}x_{11} \oplus x_{12}x_{14} = -(x_{16} - x_{15}), \\ x_5x_1x_6 = x_4x_2x_3, \\ x_{11} \oplus x_8 \oplus x_7 = x_{10} \oplus x_6, \\ x_6x_{11}x_{10} \oplus x_7x_9x_8 = 0, \\ \left(\dfrac{x_{12} + x_{14} + x_{13}}{\sqrt{2}}\right)^2 - x_{15} = x_{16} + x_{11}, \\ x_1 \oplus x_6 = x_5 \oplus x_3 \oplus x_2, \\ x_6x_8 \oplus x_9x_7 = x_{10} - x_{11}, \\ 2(x_5 + x_1 + x_6) = (x_4 + x_3 + x_2)^2, \\ x_{11}x_{13}x_{12} = x_{15}x_{14}x_{16}. \end{cases}$$

Help Caroline to find the all possible keys!

Remark. If you do it in analytic way (without computer calculations) you get twice more scores.

3.6. Problem "Biometric pin-code"

Iris is one of the most reliable biometric characteristics of a human. While measuring let us take 16-bit vector from the biometric image of an iris. As in reality, we suppose that two 16-bit biometric images of *the same human* can differ not more than by 10–20 %, while biometric images of *different people* have differences at least 40–60 %.

Let a key k be an arbitrary 5-bit vector. We suppose that the key is a pin-code that should be used in order to get an access to the bank account of a client.

To avoid situation when malefactor can steal the key of a some client and then be able to get an access to his account, the bank decided to combine usage of the key with biometric authentication of a client by iris-code. The following scheme of covering the key with biometric data was proposed:

1) on registration of a client take 16-bit biometric image b_{template} of his iris;
2) extend 5-bit key k to 16-bit string s using Hadamard encoding, i.e. if $k = (k_1, \ldots, k_5)$, where $k_i \in \{0, 1\}$, then s is the vector of values of the Boolean function $f(x_1, \ldots, x_4) = k_1 x_1 \oplus \ldots \oplus k_4 x_4 \oplus k_5$;
3) save the vector $c = b_{\text{template}} \oplus s$ on the smart-card and give it to the client. A vector c is called *biometrically encrypted key.*

To get an access to his account a client should

1) take a new 16-bit biometric image b of his iris;
2) using information from the smart-card count 16-bit vector s' as $s' = b \oplus c$;
3) decode s' to the 5-bit vector k' using Hadamard decoding procedure.

Then the bank system checks: if $k' = k$ then the client is authenticated and the key is correct; hence bank provides an access to the account of this client. Otherwise, if $k' \neq k$ then bank signals about an attempt to get illegal access to the bank account.

The problem. Find the 5-bit k of Alice if you know her smart-card data c and a new biometric image b (both are given on the picture (Fig. 4)).

Fig. 4.

Remark. Vector of values of a Boolean function f in 4 variables is a binary vector $(f(x^0), f(x^1), \ldots, f(x^{15}))$ of length 16, where $x^0 = (0, 0, 0, 0)$, $x^1 = (0, 0, 0, 1)$, …, $x^{15} = (1, 1, 1, 1)$, ordered by lexicographical order; for example, vector of values of the function $f(x_1, x_2, x_3, x_4) = x_3 \oplus x_4 \oplus 1$ is equal to (1010101010101010).

3.7. Problem "Quadratic functions"

Alice and Bob are going to use the following pseudorandom binary sequence $u = \{u_i\}$, $u_i \in \mathbb{F}_2$:

— $u_1, \ldots, u_n$ are initial values;
— $u_{i+n} = f(u_i, u_{i+1}, \ldots, u_{i+n-1})$, where

$$f \in Q_n = \{a_0 \oplus \bigoplus_{i=1}^{n} a_i x_i \oplus \bigoplus_{1 \leqslant i < j \leqslant n} a_{ij} x_i x_j : a_0, a_i, a_{ij} \in \mathbb{F}_2\}.$$

Suppose that you have intercepted the elements $u_t, u_{t+1}, \ldots, u_{t+k-1}$ of a sequence for some t. Is it possible to uniquely reconstruct the elements $u_{t+k}, u_{t+k+1}, u_{t+k+2}, \ldots$ provided $k \leqslant cn$, where c is a constant independent on n?

3.8. Problem "Biometric key"

Iris is one of the most reliable biometric characteristics of a human. While measuring let us take 128-bit biometric image of an iris. As in reality, we suppose that two 128-bit biometric images of *the same human* can differ not more than by 10–20 %, while biometric images of *different people* have differences at least 40–60 %.

Let a key k be an arbitrary 8-bit vector. It can be represented in hexadecimal notation. For example, $\texttt{e2} = 11100010$. We suppose that the key is a pin-code that should be used in order to get access to the bank account of a client.

To avoid situation when malefactor can steal the key of a some client and then be able to get an access to his account, the bank decided to combine usage of the key with biometric authentication of a client by iris-code. The following scheme of covering the key with biometric data was proposed:

1) on registration of a client take 128-bit biometric image b_{template} of his iris;
2) extend 8-bit key k to 128-bit string s using Hadamard encoding, i.e. if $k = (k_1, \dots, k_8)$, where $k_i \in \mathbb{F}_2$, then s is the vector of values of the Boolean function $f(x_1, \dots, x_7) = k_1x_1 \oplus \dots \oplus k_7x_7 \oplus k_8$;
3) save the vector $c = b_{\text{template}} \oplus s$ on the smart-card and give it to the client. A vector c is called *biometrically encrypted key*.

To get an access to his account a client should

1) take a new 128-bit biometric image b of his iris;
2) using information from the smart-card count 128-bit vector s' as $s' = b \oplus c$;
3) decode s' to 8-bit vector k' using Hadamard decoding procedure.

Then the bank system checks: if $k' = k$ then the client is authenticated and the key is correct; hence bank provides an access to the account of this client. Otherwise, if $k' \neq k$ then bank signals about an attempt to get illegal access to the bank account.

The problem. One day a person, say X, came to the bank and tried to get an access to the bank account of Alice using the smart-card. This may be noticed that person X was in hurry and may be a little bit nervous. Suddenly, another person, say Y, appeared in the bank and declared loudly: "Please stop any operation! I am Alice! My smart-card was stolen."

Bank clerk, say Claude, stopped all operations. In order to solve the situation he took new biometric images b^X and b^Y of persons X and Y respectively, and with smart-card containing vector c leaved his post for consultations with bank specialists.

When Claude came back, he already knew who was Alice. He wanted to stop the other person and call to police but that person has already disappeared. So, can you solve this problem too? Who was real Alice? Determine her 8-bit key k. You can use the data b^X, b^Y and c presented on the picture (Fig. 5). It is known also that the key of Alice contains odd number of ones.

Fig. 5.

3.9. Problem "Secret sharing"

Alena, Boris and Sergey developed the following secret sharing scheme to share a password $P \in \mathbb{F}_2^{32}$ into three parts to collectively manage money through online banking.

— Vectors $v_i^a, v_i^b, v_i^s \in \mathbb{F}_2^{32}$ and values $c_i^a, c_i^b, c_i^s \in \mathbb{F}_2$ are randomly generated for all $i = 1, \dots, 32$.
— Vectors v_i^a, v_i^b, v_i^s are known to all participants of the scheme.
— Values $c_i^a, c_i^b, c_i^s \in \mathbb{F}_2$ are known only to Alena, Boris and Sergey respectively.

— Then the secret password P is calculated by the rule

$$P = \bigoplus_{i=1}^{32} c_i^a v_i^a \oplus \bigoplus_{i=1}^{32} c_i^b v_i^b \oplus \bigoplus_{i=1}^{32} c_i^s v_i^s.$$

What is the probability that Alena and Boris together can not get any information about the password P? What is the probability that they are able without Sergey to get a guaranteed access to online banking using not more than 23 attempts?

3.10. Problem "Protocol"

Alena and Boris developed a new protocol for establishing session keys. It consists of the following three steps:

1) The system has a common prime modulus p and a generator g. Alena and Boris have their own private keys $\alpha_a \in \mathbb{Z}_{p-1}$, $\alpha_b \in \mathbb{Z}_{p-1}$ and corresponding public keys $P_a = g^{\alpha_a} \mod p$, $P_b = g^{\alpha_b} \mod p$.
2) To establish a session key Alena generates a random number $R_a \in \mathbb{Z}_{p-1}$, computes $X_a = (\alpha_a + R_a) \mod (p-1)$ and sends it to Boris. Then Boris generates his random number R_b, computes X_b in the same way as Alena and sends it back to her.
3) Alena computes the session key in the following way:

$$K_{a,b} = (g^{X_b} P_b^{-1})^{R_a} \mod p.$$

Bob computes the session key in the following way:

$$K_{b,a} = (g^{X_a} P_a^{-1})^{R_b} \mod p.$$

How can an attacker Evgeniy compute any future session key between Alena and Boris, if he steals the only one session key $K_{a,b}$?

3.11. Problem "Zerosum at AES"

Let $\mathtt{AES}_0$ be a mapping that represents the algorithm AES-256 with the all-zero key. Let $X_1, \ldots, X_{128} \in \mathbb{F}_2^{128}$ be pairwise different vectors such that

$$\bigoplus_{i=1}^{128} X_i = \bigoplus_{i=1}^{128} \mathtt{AES}_0(X_i).$$

1) Propose an effective algorithm to find an example of such vectors $X_1, \ldots, X_{128}$.
2) Provide an example of $X_1, \ldots, X_{128}$.

3.12. Problem "Latin square"

Alice has registered on Bob's server. During the registration Alice got the secret key that is represented as a latin square of order 10. A latin square is a 10×10 matrix filled with integers $0, 1, \ldots, 9$, each occurring exactly once in each row and exactly once in each column.

To get an access to Bob's resources Alice authenticates by the following algorithm:

1) Bob sends to Alice a decimal number $abcd$, where $a, b, c, d \in \{0, 1, \ldots, 9\}$ and $a \neq b$, $b \neq c$, $c \neq d$.
2) Alice performs three actions.
   — At first she finds the integer t_1 standing at the intersection of the row $(a+1)$ and the column $(b+1)$.
   — Then she finds t_2 standing at the intersection of the row $(t_1 + 1)$ and the column $(c+1)$.

— Finally, Alice finds t_3 standing at the intersection of the row (t_2+1) and the column $(d+1)$.

3) Alice sends to Bob the integer t_3.
4) Bob performs the same actions and verifies Alice's answer.
5) Steps 1–4 are repeated several times. In case of success Bob recognizes that Alice knows the secret latin square.

Find Alice's secret key if you can get the answer t_3 for any your correct input request *abcd* here: `http://nsucrypto.nsu.ru/archive/2016r2/task3`.

3.13. Problem "*nsucoin*"

Alice, Bob, Caroline and Daniel are using a digital payment system ***nsucoin*** (Fig. 6) to buy from each other different sorts of flowers. Alice sells only chamomiles, Bob — only tulips, Caroline — only gerberas and Daniel — only roses. At the beginning each person has 5 flowers. The cost of each flower is 2 coins.

Fig. 6.

Transactions are used to make purchases by transferring coins in the system *nsucoin*. Each transaction involves two different users (the seller A and the buyer B) and distributes a certain amount of coins S between A and B, say $S = S_A + S_B$. The value S is equal to the sum of all the coins received by the buyer in the indicated k transactions, $1 \leqslant k \leqslant 2$. We will say that the current transaction is *based* on these k transactions. The value S_A is the amount of coins that the buyer pays the seller for his product, $S_A > 0$; the value S_B is the rest of available amount of coins S that returns to buyer (in further transactions B can spend these coins). At the same time, coins received by users in each transaction can not be distributed more than once in other transactions.

In order for transactions to be valid they must be verified. To do this **block chain** is used. Each block verifies from 1 to 4 transactions. Each transaction to be verified can be based on already verified transactions and transactions based on verified transactions.

There are 4 *special* transactions. Each of them brings 10 coins to one user. These transactions do not based on other transactions. The first block verifies all special transactions.

Define what bouquet Alice can make from the flowers she has if the last block in chain is the following string (hash of this block in `00004558`):

```
height:2;prevHash:0000593b;ctxHash:8fef76cb;nonce:17052
```

Technical description of *nsucoin*

• **Transactions.** Transaction is given by the string `transaction` of the following format:

```
transaction = ‘‘txHash:{hashValue};{transactionInfo}’’
hashValue = Hash({transactionInfo})
transactionInfo = ‘‘inputTx:{Tx};{sellerInfo};{buyerInfo}’’
Tx = ‘‘{Tx1}’’ or ‘‘{Tx1,Tx2}’’
sellerInfo = ‘‘value1:{V1};pubKey1:{PK1};sign1:{S1}’’
buyerInfo = ‘‘value2:{V2};pubKey2:{PK2};sign2:{S2}’’
```

Here `Tx1, Tx2` are values of the field `txHash` of transactions which the current transaction based on; `V`i is a non-negative integer that is equal to the amount of coins received by the user with public key `PK`i, $0 \leqslant$ `V`$i \leqslant 10$, `V1`$\neq 0$. Digital signature

`S`i = `DecToHexStr`(`Signature`(`Key2`,`StrToByteDec`(`Hash`(`Tx1+Tx2+PK`i)))),

where `+` is concatenation operation of strings. `Key2` is private key of buyer.

In the special transactions fields `inputTx`, `sign1` are empty and there is no `buyerInfo`. For example, one of the special transactions is the following:

`txHash:1a497b59;inputTx:;value1:10;pubKey1:11;sign1:`

• **Block chain.** Each block is given by the string `block` of the following format:

`block = ‘‘height:{Height};prevHash:{PrHash};ctxHash:{CTxHash};nonce:{Nonce}’’`

Here `Height` is the block number in a chain, the first block has number 0. `PrHash` is hash of block with number `Height` $-$ 1. `CTxHash` is hash of concatenation of all the `TxHash` of transactions verified by this block. `Nonce` is the minimal number from 0 to 40000 such that block has hash of the form `0000####`.

Let `PrHash` = `00000000` for the first block.

• **Hash function.** `Hash` is calculated as reduced MD5: the result of hashing is the first 4 bytes of standard MD5 represented as a string. For example, `Hash`(‘‘`teststring`’’)= = ‘‘`d67c5cbf`’’, `Hash`(‘‘`1a497b5917`’’) = ‘‘`e0b9e4a8`’’.

• **Digital signature.** `Signature`($key, message$) is RSA digital signature with n of order 64 bits, n = 9101050456842973679. Public exponents `PK` of users are the following:

User	Alice	Bob	Caroline	Daniel
`PK`	11	17	199	5

For example,
`Signature`(2482104668331363539, 7291435795363422520) = 7538508415239841520.

• **Additional functions.** `StrToByteDec` decodes a string to bytes that are considered as a number. Given a number `DecToHexStr` returns a string that is equal to the hexadecimal representation of this number. For example, `StrToByteDec`(‘‘`e0b9e4a8`’’) = = 7291435795363422520 and `DecToHexStr`(7538508415239841520) = ‘‘`689e297682a9e6f0`’’. Strings are given in UTF-8.

Examples of a transaction and a block

• Suppose that Alice are buying from Bob 2 tulips. So, she must pay him 4 coins. The transaction of this operation, provided that Alice gets 10 coin in the transaction with hash `1a497b59`, is

`txHash:98e93fd5;inputTx:1a497b59;value1:4;pubKey1:17;sign1:689e297682a9e6f0;`
`value2:6;pubKey2:11;sign2:fec9245898b829c`

• The block on height 2 verifies transactions with hash values (values of `txHash`) `98e93fd5`, `c16d8b22`, `b782c145` and `e1e2c554`, provided that hash of the block on height 1 is `00003cc3`, is the following:

`height:2;prevHash:00003cc3;ctxHash:9f8333d4;nonce:25181`

Hash of this block is `0000642a`.

3.14. Problem "Metrical cryptosystem"

Alice and Bob exchange messages using the following cryptosystem. Let $\mathbb{F}_2^n$ be an n-dimensional vector space over the field $\mathbb{F}_2 = \{0,1\}$. Alice has a set $A \subseteq \mathbb{F}_2^n$ and Bob has a set $B \subseteq \mathbb{F}_2^n$ such that both A and B are metrically regular sets and they are metrical complements of each other. Let d be the Hamming distance between A and B. To send some number a ($0 \leqslant a \leqslant d$) Alice chooses some vector $x \in \mathbb{F}_2^n$ at distance a from the set A and sends this vector to Bob. To obtain the number that Alice has sent Bob calculates the distance b from x to the set B and concludes that the initial number a is equal to $d - b$.

Is this cryptosystem correct? In other words, does Bob correctly decrypt all sent messages, regardless of initial sets A, B satisfying given conditions and of the choice of vector x?

Remark 1. Recall several definitions and notions. The *Hamming distance* $\mathrm{d}(x,y)$ between vectors x and y is the number of coordinates in which these vectors differ. Distance from vector $y \in \mathbb{F}_2^n$ to the set $X \subseteq \mathbb{F}_2^n$ is defined as $\mathrm{d}(y,X) = \min\limits_{x \in X} \mathrm{d}(y,x)$. The *metrical complement* of a set $X \subseteq \mathbb{F}_2^n$ (denoted by $\widehat{X}$) is the set of all vectors $y \in \mathbb{F}_2^n$ at maximum possible distance from X (this maximum distance is also known as *covering radius* of a set). A set $X \subseteq \mathbb{F}_2^n$ is called *metrically regular*, if its second metrical complement $\widehat{\widehat{X}}$ coincides with X.

Remark 2. Let us consider several examples:

— Let X consist of a single vector $x \in \mathbb{F}_2^n$. It is easy to see that $\widehat{X} = \{x \oplus \mathbf{1}\}$, where $\mathbf{1}$ is the all-ones vector, and therefore $\widehat{\widehat{X}} = \{x \oplus \mathbf{1} \oplus \mathbf{1}\} = \{x\} = X$, so X is a metrically regular set; it is also easy to see that cryptosystem based on $A = \{x\}$, $B = \{x \oplus \mathbf{1}\}$ is correct.

— Let Y be a ball of radius $r > 0$ centered at x: $Y = B(r,x) = \{y \in \mathbb{F}_2^n : \mathrm{d}(x,y) \leqslant r\}$. You can verify that $\widehat{Y} = \{x \oplus \mathbf{1}\}$, but $\widehat{\widehat{Y}} = \{x\} \neq Y$, and Y is not metrically regular.

— Let X be an arbitrary subset of $\mathbb{F}_2^n$. Then, if we denote $X_0 := X$, $X_{k+1} = \widehat{X_k}$ for $k \geqslant 0$, there exists a number M such that X_m is a metrically regular set for all $m > M$. You can prove this fact as a small exercise, or simply use it in your solution.

3.15. Problem "Algebraic immunity" (Unsolved)

A mapping F from $\mathbb{F}_2^n$ to $\mathbb{F}_2^m$ is called a *vectorial Boolean function*. If $m = 1$ then F is a *Boolean function* in n variables. A *component function* F_v of F is a Boolean function defined by a vector $v \in \mathbb{F}_2^m$ as follows $F_v = \langle v, F\rangle = v_1 f_1 \oplus \ldots \oplus v_m f_m$, where $f_1, \ldots, f_m$ are coordinate functions of F. A function F has its unique *algebraic normal form* (ANF)

$$F(x) = \bigoplus_{I \in \mathcal{P}(N)} a_I \prod_{i \in I} x_i,$$

where $\mathcal{P}(N)$ is the power set of $N = \{1, \ldots, n\}$ and a_I belongs to $\mathbb{F}_2^m$. Here $\oplus$ denotes the coordinate-wise sum of vectors modulo 2. The *algebraic degree* of F is the degree of its ANF: $\deg(F) = \max\{|I| :\ a_I \neq (0,\ldots,0),\ I \in \mathcal{P}(N)\}$.

Algebraic immunity $\mathrm{AI}(f)$ of a Boolean function f is the minimal algebraic degree of a Boolean function g, $g \not\equiv 0$, such that $fg \equiv 0$ or $(f \oplus 1)g \equiv 0$. The notion was introduced by W. Meier, E. Pasalic, C. Carlet in 2004.

It is wellknown that $\mathrm{AI}(f) \leqslant \lceil n/2 \rceil$, where $\lceil x \rceil$ is the ceiling function of number x. There exist functions with $AI(f) = \lceil n/2 \rceil$ for any n.

Component algebraic immunity $\mathrm{AI}_{\mathrm{comp}}(F)$ of a function from $\mathbb{F}_2^n$ to $\mathbb{F}_2^m$ is defined as the minimal algebraic immunity of its component functions F_v, $v \neq (0,\ldots,0)$. Component

algebraic immunity was considered by C. Carlet in 2009. It is easy to see that $\mathrm{AI}_{\mathrm{comp}}(F)$ is also upper bounded by $\lceil n/2 \rceil$.

The problem. What is the tight upper bound of component algebraic immunity? For all possible combination of n and m, $m \leqslant n \leqslant 4$, vectorial Boolean functions with $\mathrm{AI}_{\mathrm{comp}}(F) = \lceil n/2 \rceil$ exist.

Construct $F : \mathbb{F}_2^5 \to \mathbb{F}_2^5$ with maximum possible component algebraic immunity 3 or prove that it does not exist.

3.16. Problem "Big Fermat numbers" (Unsolved)

It is known that constructing big prime numbers is very actual and complicated problem interesting for cryptographic applications. One of the popular way to find them is... to guess! For example to guess them between numbers of some special form. For checking there are Mersenne numbers $2^k - 1$, Fermat numbers $F_k = 2^{2^k} + 1$ for nonnegative integer k, etc.

Let us concentrate our attention on Fermat's numbers.

It is known that Fermat numbers $F_0 = 3$, $F_1 = 5$, $F_2 = 17$, $F_3 = 257$, $F_4 = 65537$ are prime. But the number $F_5 = 4\,284\,967\,297 = 641 \cdot 6\,700\,417$ is already composite as was proven by L. Euler in XVIII.

For now it is known that all Fermat numbers, where $k = 5, \ldots, 32$, are composite and there is the hypothesis that every Fermat number F_k, where $k \geqslant 5$ is composite.

Could you prove that for any big number N there exists a composite Fermat number F_k such that $F_k > N$?

4. Solutions of the problems

In this section we present solutions of the problems with paying attention to solutions proposed by the participants (right/wrong and beautiful).

4.1. Problem "Cipher from the pieces"

Solution. The only way to split this figure is the following (Fig. 7).

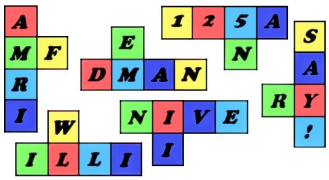

Fig. 7.

Then we need to arrange these pieces and read letters horizontally. In this way we can obtain the text «`WILLI AMFRI EDMAN 125AN NIVER SARY!`». So, the answer is «`WILLIAM FRIEDMAN 125 ANNIVERSARY!`».

The problem was devoted to the anniversary of a mathematician and cryptographer William Friedman known as «The Father of American Cryptology» (September 24, 1891 — November 12, 1969).

This problem was solved completely by 68 participants from all categories. They all acted in the same way. We could mentioned the best solutions of school students Alexander Grebennikov and Alexander Dorokhin (Presidential Physics and Mathematics Lyceum 239, Saint Petersburg), Arina Prostakova (Gymnasium 94, Yekaterinburg).

4.2. Problem "Get an access"

Solution. Here we want to describe a very compact and full solution by Vladimir Schavelev (Presidential Physics and Mathematics Lyceum 239, Saint Petersburg).

Let us prove that the sum can be any even number from 44 up to 76. It is obvious that the whole sum is even number since sums of any two numbers is even number (4, 6 or 8). Let us consider the set where the sum takes its minimal possible value. Due to condition of the problem we have two numbers a and b with the sum equal to 8, so split all the numbers in pairs, such that one of these pairs is a, b. So, the minimal sum of such pairs is $8 + 9 \cdot 4 = 44$. In the same way we obtain that the maximal possible sum is $4 + 9 \cdot 8 = 76$.

If we fill cells with two 2 and the rest of the numbers are 4, we obtain the sum $4 \cdot 18 + 4 = 76$, and this filling satisfies the condition. If we substitute any "4" by "2" we obtain the sum 74. We can continue this process, obtaining all possible variants of the sum, up to minimal, that is 44.

This problem was solved by 9 school students in the first round.

4.3. Problem "Find the key"

Solution. The answer for this problem is any set of positive integers a, b, c, d, e, f, g such that the following relation holds:

$$a^3 + b^3 + c^3 + d^3 + e^3 + f^3 + g^3 = 2016^{2017}.$$

For example, such a set can be found in the following way:

$$a^3 + b^3 + c^3 + d^3 + e^3 + f^3 + g^3 = 2016^{2017} = 2016 \cdot 2016^{2016} = 2016 \cdot (2016^{672})^3.$$

Let us divide both sides on 2016^{2016} and assume that there exist a', b', c', d', e', f', g' such that $a'^3 + b'^3 + c'^3 + d'^3 + e'^3 + f'^3 + g'^3 = 2016$. Then we can find easily these numbers, for instance, one of such sets is 3, 4, 5, 6, 7, 8, 9.

Then the original solution has the form $x = x' \cdot 2016^{672}$. So, we have

$$(3 \cdot 2016^{672})^3 + (4 \cdot 2016^{672})^3 + (5 \cdot 2016^{672})^3 + (6 \cdot 2016^{672})^3 + (7 \cdot 2016^{672})^3 + \\ + (8 \cdot 2016^{672})^3 + (9 \cdot 2016^{672})^3 = 2016^{2017}.$$

The most elegant and beautiful solution was send by Alexey Udovenko (University of Luxembourg). There also were great solutions from Alexandr Grebennikov (Presidential Physics and Mathematics Lyceum 239, Saint Petersburg), Vadzim Marchuk, Anna Gusakova, and Yuliya Yarashenia team (Research Institute for Applied Problems of Mathematics and Informatics, Institut of Mathematics, Belarusian State University), that contains a lot of such keys and even the solutions by Alexey Ripinen, Oleg Smirnov, and Peter Razumovsky team (Saratov State University), Henning Seidler and Katja Stumpp team (Technical University of Berlin) that describes all possible keys. These solutions were awarded by additional scores.

4.4. Problem "Labyrinth"

Solution. Since there is a labyrinth, one can conjecture that the ciphertext is hidden in the right path from a mouse to cheese. Such a path is unique and contains the following string: `ONFIWQHWJJFLHZAOAXWESPPNGRCTPXGJXFWUDTOXYMCWJKML`.

The first hint given tells us that the secret message begins with "`ONE...`". By comparing the first three letters "`ONF`" of the ciphertext with "`ONE`" one can suppose that a polyalphabetic cipher is used. The hint about turns gives us an idea that each simple cipher is a substitution Ceasar cipher with some shift, where each turn in the labyrinth increase shift in the alphabet (Table 4).

Table 4

Path	O	N	F	I	W	Q	H	W	J	J	F	L	H	Z	A	O	A	X	W	E	S	P	P	N
Shift	0	0	1	1	2	3	4	5	5	6	7	7	7	8	8	9	9	9	10	11	11	11	12	13
Message	O	N	E	H	U	N	D	R	E	D	Y	E	A	R	S	F	R	O	M	T	H	E	D	A
Path	G	R	C	T	P	X	G	J	X	F	W	U	D	T	O	X	Y	M	C	W	J	K	M	L
Shift	13	13	14	14	14	15	15	16	16	17	17	18	18	19	20	20	20	20	21	22	22	23	24	24
Message	T	E	O	F	B	I	R	T	H	O	F	C	L	A	U	D	E	S	H	A	N	N	O	N

Thus, the secrete message can be read as

«One hundred years from the date of birth of Claude Shannon».

It was devoted to the anniversary of an outstanding mathematician, electrical engineer, and cryptographer Claude Elwood Shannon (April 30, 1916 — February 24, 2001).

The problem was completely solved by 46 participants of all categories in both rounds. The most beautiful solutions were presented by Arina Prostakova (Gymnasium 94, Yekaterinburg), Maria Tarabarina (Lomonosov Moscow State University), Dragos Alin Rotaru, Marco Martinoli, and Tim Wood team (University of Bristol, United Kingdom), Nguyen Duc, Bui Minh Tien Dat, and Quan Doan team (University of Information Technology, Vietnam), Henning Seidler and Katja Stumpp team (Technical University of Berlin).

4.5. Problem "System of equations"

Solution. One can notice these equations can be grouped under the three subsystems:

$$1)\begin{cases} x_1 \oplus x_2 \oplus x_3 \oplus x_5 \oplus x_6 = 0, \\ x_1x_3 \oplus x_2x_4 = x_5 - x_6, \\ (x_2 + x_3 + x_4)^2 = 2(x_1 + x_5 + x_6), \\ x_2x_3x_4 \oplus x_1x_5x_6 = 0; \end{cases} \qquad 2)\begin{cases} x_6 \oplus x_7 \oplus x_8 \oplus x_{10} \oplus x_{11} = 0, \\ x_6x_8 \oplus x_7x_9 = x_{10} - x_{11}, \\ (x_7 + x_8 + x_9)^2 = 2(x_6 + x_{10} + x_{11}), \\ x_7x_8x_9 \oplus x_6x_{10}x_{11} = 0; \end{cases}$$

$$3)\begin{cases} x_{11} \oplus x_{12} \oplus x_{13} \oplus x_{15} \oplus x_{16} = 0, \\ x_{11}x_{13} \oplus x_{12}x_{14} = x_{15} - x_{16}, \\ (x_{12} + x_{13} + x_{14})^2 = 2(x_{11} + x_{15} + x_{16}), \\ x_{11}x_{12}x_{13} \oplus x_{14}x_{15}x_{16} = 0. \end{cases}$$

Note that the first and the second subsystems have a common variable x_6, just like the second and the third ones both involve x_{11}.

The first two are the subsystems having the template

$$\begin{cases} y_1 \oplus y_2 \oplus y_3 \oplus y_5 \oplus y_6 = 0, \\ y_1y_3 \oplus y_2y_4 = y_5 - y_6, \\ (y_2 + y_3 + y_4)^2 = 2(y_1 + y_5 + y_6), \\ y_2y_3y_4 \oplus y_1y_5y_6 = 0, \end{cases}$$

but the third subsystem has the following one

$$\begin{cases} y_1 \oplus y_2 \oplus y_3 \oplus y_5 \oplus y_6 = 0, \\ y_1y_3 \oplus y_2y_4 = y_5 - y_6, \\ (y_2 + y_3 + y_4)^2 = 2(y_1 + y_5 + y_6), \\ y_1y_2y_3 \oplus y_4y_5y_6 = 0. \end{cases}$$

In both of them variables are $(y_1, y_2, y_3, y_4, y_5, y_6) = \mathbf{y}$, where $y_i \in \{0, 1\}$, $i = 1, 2, \ldots, 6$. Obviously, one of the solutions of both templates is equal to $\mathbf{y} = (0, 0, 0, 0, 0, 0)$; we denote it by $\mathbf{y}^1$.

Let us consider the first template and find all its solutions.

In the case $\underline{y_5 = y_6} = y \in \{0, 1\}$ we have

$$\begin{cases} y_1 \oplus y_2 \oplus y_3 = 0, \\ y_1 y_3 \oplus y_2 y_4 = 0, \\ (y_2 + y_3 + y_4)^2 = 2(y_1 + 2y), \\ y_2 y_3 y_4 \oplus y_1 y = 0. \end{cases}$$

Evidently, the third equation holds if

a) $y_2 + y_3 + y_4 = 0$ and $y_1 + 2y = 0$; or b) $y_2 + y_3 + y_4 = 2$ and $y_1 + 2y = 2$.

From a we obtain $\mathbf{y}^1$. Using the case b and the equation $y_1 \oplus y_2 \oplus y_3 = 0$ we receive $y = 1$, $y_1 = 0$, $y_2 \oplus y_3 = 0$ (i.e. $y_2 = y_3 = y' \in \{0, 1\}$) and $2y' + y_4 = 2$. Thus, we have $\mathbf{y} = (0, 1, 1, 0, 1, 1)$, which evidently satisfies the other equations of the template. So, the second solution is $(0, 1, 1, 0, 1, 1)$; we denote it by $\mathbf{y}^2$.

In the case $\underline{y_5 \neq y_6}$ (i.e. $y_5 = 1$, $y_6 = 0$ since it must hold $y_5 \geqslant y_6$ by virtue of the second equation) we have

$$\begin{cases} y_1 \oplus y_2 \oplus y_3 \oplus 1 = 0, \\ y_1 y_3 \oplus y_2 y_4 = 1, \\ (y_2 + y_3 + y_4)^2 = 2(y_1 + 1), \\ y_2 y_3 y_4 = 0. \end{cases}$$

Again, the third equation holds if

a) $y_2 + y_3 + y_4 = 0$ and $y_1 + 1 = 0$; or b) $y_2 + y_3 + y_4 = 2$ and $y_1 + 1 = 2$.

The case a is impossible for binary y_1. Using the case b and the equation $y_1 \oplus y_2 \oplus y_3 = 1$ we receive $y_1 = 1$, $y_2 \oplus y_3 = 0$ (i.e. $y_2 = y_3 = y'' \in \{0, 1\}$) and $2y'' + y_4 = 2$. In this way we have $\mathbf{y} = (1, 1, 1, 0, 1, 0)$, which evidently satisfies the other equations of the template. So, the third solution is $(1, 1, 1, 0, 1, 0)$; we denote it by $\mathbf{y}^3$.

Let us consider the second template. It only differs from the first template in the fourth equation. But at the same time, this equation has not been used while obtaining the solutions besides checking. So, it is enough to check whether these solutions satisfy the equation $y_1 y_2 y_3 \oplus y_4 y_5 y_6 = 0$. Obviously, $\mathbf{y}^1$ and $\mathbf{y}^2$ are suitable, but $\mathbf{y}^3$ does not satisfy it.

Thus, after considering the links between corresponding subsystems that involve common variables, we get that all solutions of the initial system are the following:

1. (00000**0**0000**0**00000); 3. (01101**1**1101**0**00000); 5. (11101**0**0000**0**00000);
2. (00000**0**0000**0**11011); 4. (01101**1**1101**0**11011); 6. (11101**0**0000**0**11011).

The problem was completely solved by 26 participants, 12 of whom used computer calculations. Many participants have noticed it is possible to solve the system separately by taking into account the structure of the whole system. But at the same time, some of them have not considered the difference between the templates, in such cases extra solutions have been included in the problem's answer. The right solutions with good explanation were made by Alexandr Grebennikov (Presidential Physics and Mathematics Lyceum 239, Saint Petersburg), Dmitry Morozov (Novosibirsk State University), Anna Gusakova (Institute of Mathematics of National Academy of Sciences of Belarus).

4.6. Problem "Biometric pin-code"

Solution. First we compute $s' = b \oplus c$, as described in the algorithm, and obtain $s' = (1000\,0111\,1101\,0000)$. Note that, since a new biometric image b of Alice can be different from the image that was taken during creation of her biometrically encrypted key (by not more than 10–20 %), s' can also be different from Hadamard code codeword s corresponding to real Alice's key k, but by 10–20 % at most.

So, in order to find Alice's key, we need to find a codeword of Hadamard code of length 16, which differs from s' in not more than 20 % of bits. Since all codewords of Hadamard code differ from each other in at least 50 % of bits, such a codeword is unique (in case it exists). Thus, we can simply search for the codeword closest to s'.

Since Hadamard code of length 16 has $2^5 = 32$ codewords, we can easily (with or without use of computer) find the closest codeword: it is $s = (0000111111110000)$ and it corresponds to the key $k = (11000)$. We see that s and s' differ in only 3 bits, which is 18.75 % of all 16 bits. So the key k that we have found fits all conditions. Therefore, it is the real key of Alice.

Unfortunately, there were no complete solutions of the problem by school students.

4.7. Problem "Quadratic functions"

Solution. Here we would like to describe the solution by George Beloshapko, Stepan Gatilov, and Anna Taranenko team (Novosibirsk State University, Ledas, Sobolev Institute of Mathematics) as the simplest.

Let us consider the sequences $\{u_i\}$ and $\{v_i\}$ generated by the functions $f_u(x_1, \ldots, x_n) = x_1x_2$ and $f_v(x_1, \ldots, x_n) = x_2x_n \oplus x_1 \oplus x_n$ respectively with the initial value $\underbrace{1\ldots10}_{n}$. Let us describe the first $n^2 - n + 1$ elements of the sequences:

$$\{u_i\} = \underbrace{1\ldots1}_{n-1}\,\underbrace{0}_{1}\,\underbrace{1\ldots1}_{n-2}\,\underbrace{00}_{2}\ldots\underbrace{11}_{2}\,\underbrace{0\ldots0}_{n-2}\,\underbrace{1}_{1}\,\underbrace{0\ldots0}_{n-1}0\ldots$$

$$\{v_i\} = \underbrace{1\ldots1}_{n-1}\,\underbrace{0}_{1}\,\underbrace{1\ldots1}_{n-2}\,\underbrace{00}_{2}\ldots\underbrace{11}_{2}\,\underbrace{0\ldots0}_{n-2}\,\underbrace{1}_{1}\,\underbrace{0\ldots0}_{n-1}1\ldots$$

The first $n^2 - n$ elements of the sequences are the same, but the next elements differ. So, it is impossible to uniquely reconstruct the sequence by the segment of length cn, where c is a constant.

There were no complete solutions of the problem in the first round. In the second round, in addition to the given solution, the problem was completely solved by Alexey Udovenko (University of Luxembourg), Alexey Ripinen, Oleg Smirnov, and Peter Razumovsky team (Saratov State University), Maxim Plushkin, Ivan Lozinskiy, and Alexey Solovev team (Lomonosov Moscow State University).

4.8. Problem "Biometric key"

Solution. To solve the problem, we compute values s' for both persons X and Y and check if any of them is close enough to some Hadamard code codeword, corresponding to the key with odd number of 1's. Here "close enough" means difference in not more than 20 % bits, since two biometric images of the same person can not be more different from each other.

Let us denote these values s^X and s^Y for person X and Y respectively. Now, with the use of a computer program, we can go through all 256 codewords of Hadamard code of length 128, and find those which are closest to s^X and s^Y.

For person Y, the closest Hadamard codeword differs from s^Y by 49 bits (around 38.28 %). This means that there exists no key k, using which person Y could make himself biometrically encrypted key c.

For person X, the closest Hadamard codeword is obtained from (11011010) and it differs from s^X by 25 bits (around 19.53 %). This means that person X is Alice and her 8-bit key is $k = (11011010)$ (it also has an odd number of 1's, which solidifies our confidence).

Note that there are many interesting approaches for combining biometrics with cryptography (for example, see [4]).

The problem was completely solved by Irina Slonkina (Novosibirsk State University of Economics and Management) and George Beloshapko (Novosibirsk State University) in the first round, and by 10 teams in the second round, Evgeniya Ishchukova, Ekaterina Maro, and Dmitry Alekseev team (Southern Federal University, Taganrog) was among them.

4.9. Problem "Secret sharing"

Solution. Without Sergey, Alena and Boris know the following vector:

$$P' = P \oplus \bigoplus_{i=1}^{32} c_i^s v_i^s.$$

So, they do not know any information about P if and only if the dimension of the linear span $\langle v_1^s, \dots, v_{32}^s \rangle$ is equal to 32, i.e. $v_1^s, \dots, v_{32}^s$ form basis of $\mathbb{F}_2^{32}$. Otherwise, $\langle v_1^s, \dots, v_{32}^s \rangle \subset \mathbb{F}_2^{32}$ and $P \in P' \oplus \langle v_1^s, \dots, v_{32}^s \rangle$, it means that there are not more than $2^{\mathrm{rk}\langle v_1^s, \dots, v_{32}^s \rangle}$ ways for Alena and Boris to get P.

Let V be a 32×32 matrix with columns $v_1^s, \dots, v_{32}^s$. Note that $\mathrm{rk}V = \mathrm{rk}\langle v_1^s, \dots, v_{32}^s \rangle$. Since all bits of $v_1^s, \dots, v_{32}^s$ are randomly generated (and independent of each other), all elements of V are randomly generated too.

Hence, the probability p_1 that Alena and Boris can not get any information about P is equal to $N_1/2^{32 \cdot 32}$, where N_1 is the number of matrices V of rank 32:

$$p_1 = \frac{(2^{32} - 2^0)(2^{32} - 2^1) \cdot \ldots \cdot (2^{32} - 2^{31})}{2^{32 \cdot 32}} \approx 0.288788.$$

In order for Alena and Boris to get a guaranteed access to online banking without Sergey using not more than 23 attempts, it should hold $2^{\mathrm{rk}\langle v_1^s, \dots, v_{32}^s \rangle} \leqslant 23$, i.e. $\mathrm{rk}\langle v_1^s, \dots, v_{32}^s \rangle \leqslant 4$. So, the probability p_2 of that is equal to $N_2/2^{32 \cdot 32}$, where N_2 is the number of matrices V of rank not more than 4. Note that the number of $n \times n$ matrices of rank k is equal to

$$R_n^k = \frac{(2^n - 2^0)^2 (2^n - 2^1)^2 \cdot \ldots \cdot (2^n - 2^{k-1})^2}{(2^k - 2^0)(2^k - 2^1) \cdot \ldots \cdot (2^k - 2^{k-1})}.$$

Therefore,

$$p_2 = \frac{R_{32}^0 + R_{32}^1 + R_{32}^2 + R_{32}^3 + R_{32}^4}{2^{32 \cdot 32}} \approx 1.625 \cdot 2^{-783}.$$

In the first round, the problem was completely solved by Alexey Udovenko (University of Luxembourg) and Igor Fedorov (Novosibirsk State University); almost completely solved by Mohammadjavad Hajialikhani (Sharif University of Technology, Iran), Pavel Hvoryh (Omsk State Technical University), George Beloshapko (Novosibirsk State University) and Ekaterina Kulikova (Munich, Germany). In the second round (in addition to the first round) the problem was completely solved by Aliaksei Ivanin, Oleg Volodko, and Konstantin Pavlov team (Belarusian State University).

4.10. Problem "Protocol"

Solution. Here we would like to describe the solution proposed by Alexey Udovenko (University of Luxembourg), that is similar to the author's one, but is more elegant and compact.

The session key is equal to

$$K_{a,b} = g^{R_a R_b} = g^{(X_a-\alpha_a)(X_b-\alpha_b)} = g^{X_aX_b-\alpha_aX_b-\alpha_bX_a+\alpha_a\alpha_b} \bmod p.$$

Evgeniy observes X_a and X_b and he also knows g, $P_a = g^{\alpha_a} \bmod p$ and $P_b = g^{\alpha_b} \bmod p$. From g, P_a, P_b and one exposured key he can compute

$$s = g^{\alpha_a\alpha_b} = K_{a,b}/(g^{X_aX_b}P_a^{-X_b}P_b^{-X_a}) \bmod p.$$

Then for new sessions he can intercept new X_a and X_b and easily compute new

$$K'_{a,b} = g^{X_aX_b}P_a^{-X_b}P_b^{-X_a}s \mod p.$$

It worth noting that recovering of the next keys would be impossible if the protocol has a property of so-called *forward secrecy*. More details can be found in [5].

This problem was completely solved by 7 participants in the first round, Robert Spencer (University of Cape Town, South Africa) was among them, and by 15 teams in the second round, Roman Lebedev, Ilia Koriakin, and Vlad Kuzin team (Novosibirsk State University) was among them. All solutions proposed were made in the similar way. Also there were solutions with reduced score, containing some inexactness, but they were still very close to complete ones.

4.11. Problem "Zerosum at AES"

Solution. Let us describe one of the simplest approach to get the solution proposed and implemented by Alexey Udovenko (University of Luxembourg).

Denote by $Y_i = \mathtt{AES}_0(X_i) \oplus X_i$. The equation can be rewritten:

$$\bigoplus_{i=1}^{128}(\mathtt{AES}_0(X_i) \oplus X_i) = \bigoplus_{i=1}^{128} Y_i = 0.$$

First we encrypt random 256 plaintexts and obtain $Y_1, \ldots, Y_{256}$. Then the problem is to find distinct indices $i_1, \ldots, i_{128}$ such that $Y_{i_1} \oplus \ldots \oplus Y_{i_{128}} = 0$.

Let M be the 128×256 binary matrix with columns $Y_1, \ldots, Y_{256}$. Let us consider the solutions of the linear equation $Mz = 0$: for any solution z it holds

$$\bigoplus_{i:z_i=1} Y_i = 0.$$

Moreover, the Hamming weight of a random solution vector z will be close to 128 and with high probability equal to 128. Next, we try to find a solution vector of weight 128.

All information about the block cipher AES can be found in the book [6] of AES authors J. Daemen and V. Rijmen.

The problem was completely solved (proposed an algorithm and the solution) by three teams: Alexey Udovenko (University of Luxembourg), George Beloshapko, Stepan Gatilov, and Anna Taranenko team (Novosibirsk State University, Ledas, Sobolev Institute of Mathematics) and Maxim Plushkin, Ivan Lozinskiy, and Alexey Solovev team (Lomonosov Moscow State University).

4.12. Problem "Latin square"

Solution. A $n \times n$ latin square can be given as the set of its columns permutation σ_i, $i = 1, \ldots, n$. Then, the answer t_3 on the request *abcd* can be calculated as $\sigma_d\sigma_c\sigma_b(a)$ (here $\sigma = \sigma_d\sigma_c\sigma_b$ denotes the permutations composition). Choosing all possible $a \neq b$, one is able to reconstruct σ. It is needed 9 request-answer pairs. The inverse permutations can also be found, $\sigma^{-1} = \sigma_b^{-1}\sigma_c^{-1}\sigma_d^{-1}$.

Moreover, for any distinct $i, j \in \{0, 1, \ldots, 9\}$ the following permutation is recovered using 18 request-answer pairs:

$$\sigma_j^{-1}\sigma_i = (\sigma_j^{-1}\sigma_c^{-1}\sigma_d^{-1})(\sigma_d\sigma_c\sigma_i).$$

Here c, d are arbitrary numbers from the set $\{0, 1, \ldots, 9\} \setminus \{i, j\}$.

If one knows $\sigma_j^{-1}\sigma_i$, then he also knows 10 preimage–image pairs x, y: $y = \sigma_j^{-1}\sigma_i(x)$. Each of such a pair means an equality $\sigma_i(x) = \sigma_j(y)$, from which σ_j can be expressed by σ_i.

For example, let us express σ_j, $j = 1, 2, \ldots, 9$, by σ_0. To do this we need $18 \cdot 9 = 162$ pairs of request–answer. Then we will check all possible variants of σ_0 (there are 10! such variants). For each variant $\hat{\sigma}_0$ we find $\hat{\sigma}_j$, build the corresponding latin square $\hat{L}$ and check whether answers on $\hat{L}$ is equal to answers on the secret latin square. Using $162 \gg \log_{10} 10!$ answers Alice's secret key will be uniquely recovered with high probability.

The answer is the following 10×10 latin square (Fig. 8).

3	9	2	1	7	8	5	0	6	4
4	3	7	9	1	5	2	6	8	0
1	8	6	7	0	9	4	3	5	2
2	6	5	3	9	7	1	4	0	8
6	0	9	4	5	1	8	2	3	7
0	7	3	6	2	4	9	8	1	5
8	4	1	0	3	2	6	5	7	9
5	1	4	8	6	0	7	9	2	3
7	2	8	5	4	3	0	1	9	6
9	5	0	2	8	6	3	7	4	1

Fig. 8.

The problem was completely solved by 26 teams. The best solution was proposed by Sergey Titov (Ural State University of Railway Transport, Yekaterinburg).

4.13. Problem "*nsucoin*"

Solution. The payment method used is an example of blockchain based money working on the proof-of-work principal. The first such a system *bitcoin* was proposed in [7].

To solve this problem one need to restore a history of transactions that leads to the block on height 2 from condition of the problem:

```
height:2; prevHash:0000593b; ctxHash:8fef76cb; nonce:17052
```

Given such a history of transactions one can find how many flowers of different kinds Alice has at the end of trading.

Solution plan consists of the following steps:

1. One need to find each user's private key (using private keys one is able to make a sign of each user and as a result generate transactions).
2. One need to find all special transactions that give to each user 10 coins.
3. One need to find all possible blocks on height 0.

4. Looking through all possible transactions, one need to find blocks on height 1 for each block on height 0 so that hash of each block found will be equal to value of prevHash field of the given block, i.e. `0000593b`.

5. It is known that there is at least one block among the blocks found on the previous step such that there exist transactions such that hash of these transactions concatenation is equal to `8fef76cb`. One need to find these transactions.

6. Thus, all transactions history leading to the given block on height 2 is found. It remains to track the movement of flowers and give an answer.

While searching blocks we need to remember that `nonce` does not exceed 40000. Despite the fact that there are 24 transpositions of special transactions, only 6 of them can be verified by a block. Also, it is useful to remember that each participant can sell only 5 flowers, each of them costs 2 coins, and a participant can not buy anything if he does not have coins.

It is easy to factor a given small module of RSA:

$$n = p \cdot q = 2250339337 \cdot 4044301367 = 9101050456842973679,$$
$$\varphi(n) = (p-1)(q-1) = 9101050450548332976.$$

Then, one can find private keys as inverse numbers to public keys module $\varphi(n)$ (Table 5).

T a b l e 5

User	PubKey	PrivKey
Alice	11	2482104668331363539
Bob	17	3747491361990490049
Caroline	199	9009582606824229127
Daniel	5	7280840360438666381

Despite all the limitations of searching, it turns out to be quite time-consuming. There were laid some hints in the condition of the problem allowing one to get a solution. This two hints were published on the Olympiad website:

18 November. A tip to reduce exhaustive search: The transaction from the example has been verified by the block with height 1.

20 November. All hashes of transactions that are verified by the block from the example correspond to hashes of transactions verified by the block with height 1 into the sought-for blockchain.

The first participant who found out the right answer was Alexey Udovenko (University of Luxembourg). He based only on the first hint and guessed that all four transactions hashes from the example are contained in the block with height 1. By the other words, he guessed the second hint and found the answer we had conceived.

In fact, we knew only one answer but we hoped that someone was able to find another one, and it happened! There were two teams that found two answers based only on the first hint: George Beloshapko, Stepan Gatilov, and Anna Taranenko team (Novosibirsk State University, Ledas, Institute of Mathematics) and the team of Alexey Ripinen, Oleg Smirnov, and Peter Razumovsky team (Saratov State University).

The final flowers distribution and transactions history of two solutions are presented in Tables 6, 7.

There were 9 teams received scores for this problem, 2 teams found two answers, 1 found an answer based only on the first hint, 2 found answer based on both hints, 3 teams found blocks with height 0 and 1 but did not found transactions verified by block with height 2, and 1 team found only a block with height 0. Probably, someone were not able to find

Table 6

Flowers distribution of two solutions

User	Camomiles		Tulips		Gerberas		Roses	
	1st sol.	2d sol.	1st sol.	2d sol.	1st sol.	2d sol.	1st sol.	2d sol.
Alice	1	1	2	3	3	0	1	0
Bob	4	2	0	2	2	2	0	1
Caroline	0	2	0	0	0	1	1	0
Daniel	0	0	3	0	0	2	3	4

answer due to the following fact: each transaction to be verified can be based not only on already verified transactions but also on transactions based on verified transactions. It is transactions of the second type that are in blocks. Also, the difficulties for the decisive could create transactions based on two other transactions. There is one such transaction verified by the block with height 2 in the first answer and four such transactions in the second answer.

Special thanks to the team of George Beloshapko, Stepan Gatilov, and Anna Taranenko for illustrating the solutions (Fig. 9).

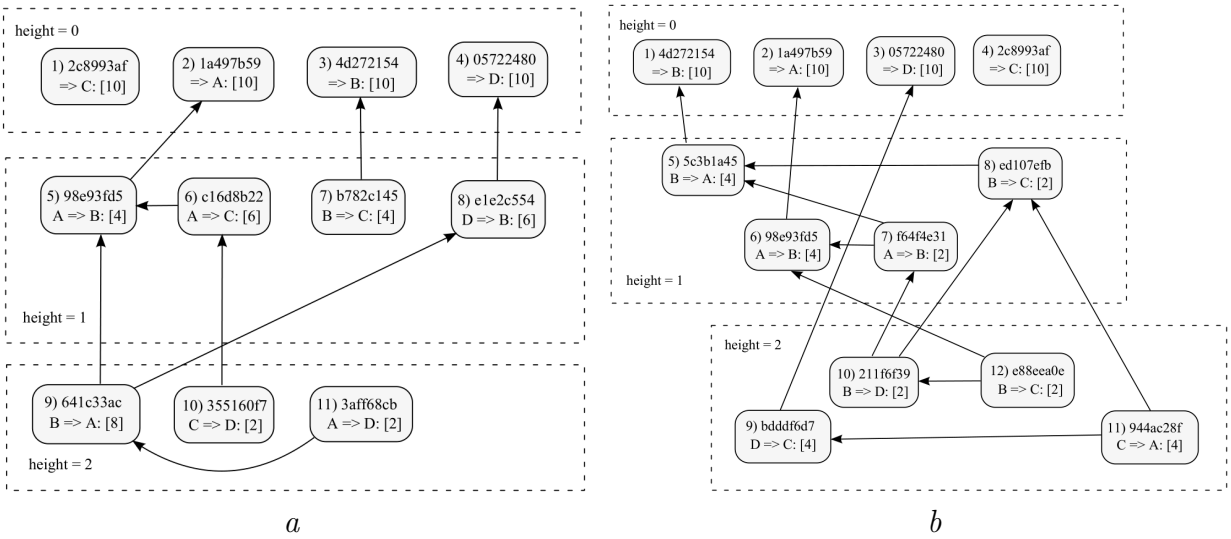

Fig. 9. Illustrations of transactions history of two solutions.

4.14. Problem "Metrical cryptosystem"

Solution. First of all, let us reformulate the problem: the statement "The cryptosystem presented is correct" is equivalent to the statement "It holds $\mathrm{d}(x, A) + \mathrm{d}(x, B) = d$ for any pair of metrically regular sets $A, B \subseteq \mathbb{F}_2^n$ at the Hamming distance d from each other and any vector $x \in \mathbb{F}_2^n$".

After considering pairs of metrically regular sets in spaces of small dimension, one may conclude that the cryptosystem is correct, and try to prove it. However, it is not correct. Here we will present the solution of George Beloshapko, Stepan Gatilov, and Anna Taranenko team (Novosibirsk State University, Ledas, Sobolev Institute of Mathematics) as the most elegant.

Table 7

Transactions history of two solutions

First solution:
Tx1: txHash:2c8993af;inputTx:;value1:10;pubKey1:199;sign1:
Tx2: txHash:1a497b59;inputTx:;value1:10;pubKey1:11;sign1:
Tx3: txHash:4d272154;inputTx:;value1:10;pubKey1:17;sign1:
Tx4: txHash:05722480;inputTx:;value1:10;pubKey1:5;sign1:
Block 1: height:0;prevHash:00000000;ctxHash:bde430dd;nonce:21095 with hash 00003cc0
Tx5: txHash:98e93fd5;inputTx:1a497b59;value1:4;pubKey1:17;sign1:689e297682a9e6f0;value2:6;pubKey2:11;sign2:fec9245898b829c
Tx6: txHash:c16d8b22;inputTx:98e93fd5;value1:6;pubKey1:199;sign1:611cddad83c8e352;value2:0;pubKey2:11;sign2:6ab37bb9f5f4249d
Tx7: txHash:b782c145;inputTx:4d272154;value1:4;pubKey1:199;sign1:5f50adf4a51899e5;value2:6;pubKey2:17;sign2:78a4f7d6d0d578d7
Tx8: txHash:e1e2c554;inputTx:05722480;value1:6;pubKey1:17;sign1:b91ab453e4bcb53;value2:4;pubKey2:5;sign2:3803907748416c12
Block 2: height:1;prevHash:00003cc0;ctxHash:9f8333d4;nonce:21438 with hash 0000593b
Tx9: txHash:641c33ac;inputTx:98e93fd5,e1e2c554;value1:8;pubKey1:11;sign1:110a7c6e1dfd937;value2:2;pubKey2:17;sign2:14ad50a36ef5540d
Tx10: txHash:3aff68cb;inputTx:641c33ac;value1:2;pubKey1:5;sign1:20281a4d62b20cb7;value2:6;pubKey2:11;sign2:7b662b128b1200bf
Tx11: txHash:3aff68cb;inputTx:641c33ac;value1:2;pubKey1:5;sign1:20281a4d62b20cb7;value2:6;pubKey2:11;sign2:7b662b128b1200bf
Block 3: height:2;prevHash:0000593b;ctxHash:8fef76cb;nonce:17052 with hash 000023d4

Second solution:
Tx1: txHash:4d272154;inputTx:;value1:10;pubKey1:17;sign1:
Tx2: txHash:1a497b59;inputTx:;value1:10;pubKey1:11;sign1:
Tx3: txHash:05722480;inputTx:;value1:10;pubKey1:5;sign1:
Tx4: txHash:2c8993af;inputTx:;value1:10;pubKey1:199;sign1:
Block 1: height:0;prevHash:00000000;ctxHash:4ddc0244;nonce:20670 with hash 0000857a
Tx5: txHash:txHash:5c3b1a45;inputTx:4d272154;value1:4;pubKey1:11;sign1:5866152e5bf782f1;value2:6;pubKey2:17;sign2:78a4f7d6d0d578d7
Tx6: txHash:98e93fd5;inputTx:1a497b59;value1:4;pubKey1:17;sign1:689e297682a9e6f0;value2:6;pubKey2:11;sign2:fec9245898b829c
Tx7: txHash:f64f4e31;inputTx:98e93fd5,5c3b1a45;value1:2;pubKey1:17;sign1:6a9369e096c2cd05;value2:8;pubKey2:11;sign2:5021efb4fb05e703
Tx8: txHash:ed107efb;inputTx:5c3b1a45;value1:2;pubKey1:199;sign1:7e12b526fd676d32;value2:4;pubKey2:17;sign2:4d9942d6d31e6392
Block 2: height:1;prevHash:0000857a;ctxHash:00f229f7;nonce:19574 with hash 0000593b
Tx9: txHash:bdddf6d7;inputTx:05722480;value1:4;pubKey1:199;sign1:6b33ced4b96ed36f;value2:6;pubKey2:5;sign2:3803907748416c12
Tx10: txHash:211f6f39;inputTx:ed107efb,f64f4e31;value1:2;pubKey1:5;sign1:6b12e0f356e951ea;value2:4;pubKey2:17;sign2:88b48219f607775
Tx11: txHash:944ac28f;inputTx:ed107efb,bdddf6d7;value1:4;pubKey1:11;sign1:395867768ee9f790;value2:2;pubKey2:199;sign2:555727e07c7e0c97
Tx12: txHash:e88eea0e;inputTx:211f6f39,98e93fd5;value1:2;pubKey1:199;sign1:76a91f809d7468b7;value2:6;pubKey2:17;sign2:1c6905596113f9a6
Block 3: height:2;prevHash:0000593b;ctxHash:8fef76cb;nonce:17052 with hash 000023d4

Consider the Hadamard code of length 16 as A:

$$A = \{(0000000000000000), (0000000011111111), (0000111100001111), (0000111111110000), (0011001100110011), (0011001111001100), (0011110000111100), (0011110011000011), (0101010101010101), (0101010110101010), (0101101001011010), (0101101010100101), (0110011001100110), (0110011010011001), (0110100101101001), (0110100110010110)\}.$$

In other words, A is the set of values vectors of all linear functions in 4 variables. It is easy to check (for example, using computer program), that metrical complement B of the set A is the set of values vectors of all affine function (excluding linear) in 4 variables and vice versa, and the distance between A and B is equal to 8:

$$B = \{(1001011001101001), (1001011010010110), (1001100101100110), (1001100110011001), (1010010101011010), (1010010110100101), (1010101001010101), (1010101010101010), (1100001100111100), (1100001111000011), (1100110000110011), (1100110011001100), (1111000000001111), (1111000011110000), (1111111100000000), (1111111111111111)\}.$$

Now consider vector $x = (0000000000010111)$. It is at distance 4 from set A and at distance 6 from set B, therefore

$$\mathrm{d}(x, A) + \mathrm{d}(x, B) = 10 > 8,$$

which shows us that the cryptosystem is incorrect. Other solutions include straightforward computer search for metrically regular sets and vectors, for which $\mathrm{d}(x, A) + \mathrm{d}(x, B) \neq d$. The smallest dimension for which example is found is equal to 7. Despite the fact that the cryptosystem is not correct, many participants, who tried to prove that the cryptosystem is correct, received 1 or 2 points for creative ideas.

4.15. Problem "Algebraic immunity" (Unsolved, Special Prize)

Solution. The problem was completely solved! It was the first time in the Olympiad history. We would like to describe part of the solution proposed by Alexey Udovenko (University of Luxembourg). At first we mentioned that the notion of component algebraic immunity was proposed by C. Carlet in [8].

"It is natural to try simple constructions to build a vectorial Boolean function from a 1-bit Boolean function. Let $f : \mathbb{F}_2^n \to \mathbb{F}_2$ be some function such that $\mathrm{AI}(f) = \lceil n/2 \rceil$. Let $F_{f,m} : \mathbb{F}_2^n \to \mathbb{F}_2^m$ be defined as $F_{f,m}(x) = f(x)||f(L(x))||f(L^2(x))|| \ldots ||f(L^{m-1}(x))$, where L is some linear mapping, for example rotation of the input vector left by one position. We generated random Boolean functions f and for those which had $\mathrm{AI}(f) = \lceil n/2 \rceil$ we found the largest m such that the corresponding $F_{f,m}$ had high algebraic immunity too, i.e. such that $\mathrm{AI}_{\mathrm{comp}}(F_{f,m}) = \lceil n/2 \rceil$. Note that this approach can produce functions only with $m \leqslant n$ if L is the rotation mapping. Here are our results (L is always rotation left by one):

- $n = 3$: Let $f(x_0, x_1, x_2) = x_0 + x_1 + x_1x_2$. Then $\mathrm{AI}_{\mathrm{comp}}(F_{f,3}) = 2$.
- $n = 4$: Let $f(x_0, \ldots, x_3) = x_0x_1x_2 + x_0x_1 + x_3$. Then $\mathrm{AI}_{\mathrm{comp}}(F_{f,4}) = 2$.
- $n = 5$: Let $f(x_0, \ldots, x_4) = x_0x_1x_2x_3 + x_0x_1x_2 + x_0x_1x_3 + x_0 + x_1x_3 + x_2x_4 + x_4$. Then $\mathrm{AI}_{\mathrm{comp}}(F_{f,5}) = 3$. Interestingly, $F_{f,5}$ happens to be a permutation. Its lookup table is as follows: (0, 24, 12, 20, 6, 29, 10, 1, 3, 23, 30, 26, 5, 22, 16, 19, 17, 9, 27, 2, 15, 21, 13, 7, 18, 4, 11, 14, 8, 28, 25, 31)."

Alexey Udovenko also found a function with optimum component algebraic immunity $\lceil n/2 \rceil$ by a rotational symmetries construction for the following values (n, m): (6,6), (7,3), (8,8), (9,2), (10,10). Moreover, he found function with the maximum component algebraic immunity by random search for the $(n, m) \in \{(4, 5), (6, 8)\}$.

Additionally, he has found F' which is also a permutation of $\mathbb{F}_2^5$ such that $\mathrm{AI}_{\mathrm{comp}}(F') = 3$ but which is differentially 2-uniform (is APN) and its nonlinearity is equal to 12. Therefore, it is more suitable for cryptography. However, the algebraic degree of F' is equal to 3. The lookup table of F' is as follows: $F' =$ (0, 12, 6, 11, 3, 25, 21, 4, 17, 7, 28, 9, 26, 10, 2, 27, 24, 22, 19, 8, 14, 18, 20, 23, 13, 16, 5, 15, 1, 30, 29, 31).

Alexey Udovenko was the only person who completely solved this problem. We also presented several ideas for finding constructions of vectorial Boolean function with optimum component algebraic immunity, but unfortunately it was not completed.

4.16. Problem "Big Fermat numbers" (Unsolved, Special Prize)

Solution. There was no complete solution of this problem. The best solution was proposed by Alisa Pankova (University of Tartu, Estonia). The main idea was to show how to use a prime Fermat number $2^{2^k} + 1$ to construct a larger composite Fermat number $2^{2^n} + 1$ for a certain $n > k$. In this way, if there is an arbitrarily large prime Fermat number, then there exists even larger composite Fermat number, so there is no point after which all Fermat numbers become prime. Unfortunately, there was a mistake in the solution.

The team of Vadzim Marchuk, Anna Gusakova, and Yuliya Yarashenia (Belarusian State University) found a very nice heuristic bound but the statement was not proven with probability 1. Several interesting attempts were also proposed by the team of Alexey Udovenko (University of Luxembourg), George Beloshapko, Stepan Gatilov, and Anna Taranenko team (Novosibirsk State University, Ledas, Sobolev Institute of Mathematics), Roman Ginyatullin, Anatoli Makeyev, and Victoriya Vlasova team (Moscow Engineering Physics Institute), Nikolay Altukhov, Vladimir Bushuev, and Roman Chistiakov team (Bauman Moscow State Technical University), Evgeniy Manaka, Aleksandr Sosenko, and Pavel Ivannikov team (Bauman Moscow State Technical University).

5. Winners of the Olympiad

At all, 10 school students, 19 university students and 5 professionals were awarded by diplomas in the first round, 16 university students teams and 11 professionals teams were awarded in the second round of NSUCRYPTO-2016.

The first places in different categories were won by the following participants:

- *First round, Section A, school student*: **Alexander Grebennikov** (Russia, Saint Petersburg, Presidential PML 239);
- *First round, Section B, university student*: **Robert Spencer** (South Africa, Cape Town, University of Cape Town) and **Henning Seidler** (Germany, Berlin, Technische Universität Berlin);
- *First round, Section B, professional*: **Alexey Udovenko** (Luxembourg, Luxembourg, University of Luxembourg);
- *Second round, university student*: **Maxim Plushkin, Ivan Lozinskiy,** and **Alexey Solovev** team (Russia, Moscow, Lomonosov Moscow State University);
- *Second round, professional*: **Alexey Udovenko** (Luxembourg, Luxembourg, University of Luxembourg). Note that Alexey also got a special prize for solving an unsolved problem "Algebraic immunity".

All information about the winners of NSUCRYPTO-2016 can be found on the official website at http://nsucrypto.nsu.ru/archive/2016/

Acknowledgements

We would like to thank Sergei Kiazhin for his interesting ideas of the problems. We thank Novosibirsk State University for the financial support of the Olympiad and invite you to take part in the next NSUCRYPTO that will be held in October, 2018. Your ideas on the mentioned unsolved problems are also very welcome and can be sent to nsucrypto@nsu.ru.

REFERENCES

1. *Agievich S., Gorodilova A., Kolomeec N., et al.* Problems, solutions and experience of the first international student's Olympiad in cryptography. Prikladnaya Diskretnaya Matematika, 2015, no. 3, pp. 41–62.
2. *Agievich S., Gorodilova A., Idrisova V., et al.* Mathematical problems of the second international student's Olympiad in cryptography. Cryptologia, 2017, vol. 41, iss. 6, pp. 534–565.
3. *Geut K., Kirienko K., Sadkov P., et al.* On explicit constructions for solving the problem "A secret sharing". Prikladnaya Diskretnaya Matematika. Prilozhenie, 2017, no. 10, pp. 68–70. (in Russian)
4. *Rathgeb C. and Uhl C.* A survey on biometric cryptosystems and cancelable biometrics. EURASIP J. Inform. Security, 2011, vol. 2011:3. https://doi.org/10.1186/1687-417X-2011-3
5. *Diffie W., Van Oorschot P. C., and Wiener M. J.* Authentication and authenticated key exchanges. Designs, Codes and Cryptography, 1992, vol. 2, iss. 2, pp. 107–125.
6. *Daemen J. and Rijmen V.* The Design of Rijndael: AES — the Advanced Encryption Standard. Springer Verlag, 2002.
7. *Nakamoto S.* Bitcoin: a peer-to-peer electronic cash system. 2009. Available at https://bitcoin.org/bitcoin.pdf
8. *Carlet C.* On the algebraic immunities and higher order nonlinearities of vectorial Boolean Functions. Proc. NATO Advanced Research Workshop ACPTECC, Veliko Tarnovo, Bulgaria, October 6–9, 2008, Amsterdam, IOS Press, 2009, pp. 104–116.

МАТЕМАТИЧЕСКИЕ МЕТОДЫ СТЕГАНОГРАФИИ

УДК 004.056.5

ЭФФЕКТИВНОЕ ОБНАРУЖЕНИЕ СТЕГАНОГРАФИЧЕСКИ СКРЫТОЙ ИНФОРМАЦИИ ПОСРЕДСТВОМ ИНТЕГРАЛЬНОГО КЛАССИФИКАТОРА НА ОСНОВЕ СЖАТИЯ ДАННЫХ

В. А. Монарев*, А. И. Пестунов**

*Институт вычислительных технологий СО РАН, г. Новосибирск, Россия

**Новосибирский государственный университет экономики и управления, г. Новосибирск, Россия

Предлагается концепция интегрального классификатора, предназначенного для повышения точности методов стегоанализа, которые базируются на машинном обучении. Вместо одиночного классификатора, принимающего решение о пустоте или заполненности контейнера, предлагается обучать набор классификаторов, каждый из которых предназначен для обработки контейнеров с определёнными свойствами. В качестве реализации данной концепции представлен интегральный классификатор, основанный на сжатии данных, что подразумевает выбор отдельного классификатора из набора на основе коэффициентов сжатия контейнеров. Эффективность предлагаемого классификатора для решения задачи обнаружения скрытой информации экспериментально продемонстрирована для современных методов адаптивного внедрения HUGO, WOW и S-UNIWARD на изображениях-контейнерах из известной базы BOSSbase 1.01. Показано, что в зависимости от метода внедрения и количества скрываемой информации ошибку обнаружения можно снизить на 0,05–0,16 по сравнению с лучшими из известных результатов.

Ключевые слова: *стегоанализ, ошибка обнаружения, HUGO, WOW, UNIWARD, метод опорных векторов, PSRM-признаки, SRM-признаки, ансамблевый классификатор, интегральный классификатор.*

DOI 10.17223/20710410/40/5

EFFICIENT STEGANOGRAPHY DETECTION BY MEANS OF COMPRESSION-BASED INTEGRAL CLASSIFIER

V. A. Monarev*, A. I. Pestunov**

*Institute of Computational Technologies of SB RAS, Novosibirsk, Russia

**Novosibirsk State University of Economics and Management, Novosibirsk, Russia

E-mail: viktor.monarev@gmail.com, pestunov@gmail.com

We propose a model of an integral classifier in order to solve the problem of binary steganalysis by means of machine-learning tools more efficiently. The problem of binary steganalysis consists in recognizing whether a given container is empty or contains a certain payload embedded via a certain steganographic algorithm. In steganalysis, such problem is often solved using such machine-learning techniques as the support

vector machine and the ensemble classifier. Instead of using a single classifier (as it is done now) which is intended to make an ultimate decision about whether the container is empty or not, the proposed in this paper integral classifier consists of several classifiers and works in such a way that each of them processes only those containers which satisfy a certain condition. Within the proposed model, we develop a compression-based integral classifier which works as follows. The training set of classifiers is splitted into several subsets according to the containers compression rate; then a corresponding number of classifiers are trained, but each classifier is injected only with an ascribed subset. The testing containers are distributed between the classifiers (also according to their compression rate) and the decision about the certain container is made by the chosen classifier. In order to demonstrate the power of the integral classifier, we performed some experiments using the famous de-facto standard images database BOSSbase 1.01 as a source of the containers along with contemporary content-adaptive embedding algorithms HUGO, WOW and S-UNIWARD. Comparison with state-of-the-art results (obtained for the single support vector machine and the ensemble classifier) demonstrated that, depending on the case, the integral classifier allows to decrease the detection error by 0.05–0.16.

Keywords: *steganalysis, detection error, HUGO, WOW, UNIWARD, support vector machine, ensemble classifier, integral classifier, projected spatial rich model, spatial rich model, compression.*

Введение

Классическая задача бинарного стегоанализа [1 – 3] состоит в том, чтобы определить, присутствует в заданном контейнере информация или нет. При этом, в отличие от так называемого «слепого стегоанализа» [4 – 6], метод внедрения и размер внедрения предполагаются известными. Ошибку обнаружения стегоанализа принято вычислять эмпирически для некоторого контрольного множества как отношение числа ошибок первого и второго рода или, другими словами, ложных срабатываний и пропущенных обнаружений [1 – 3, 7, 8]. В целом, такой подход позволяет сравнивать точность различных методов стегоанализа, однако он тем не менее использует усреднение и не позволяет учесть особенности изображений, влияющие на достоверность обнаружения. Другими словами, в контрольном множестве часто оказывается возможным выбрать подмножества, которые характеризуются тем, что вычисленная только по ним ошибка обнаружения может быть как больше, так и меньше ошибки, вычисленной по всему множеству.

В работе [9] авторами предложен подход, названный «предварительной фильтрацией» и позволяющий учесть эту особенность. Идея подхода заключается в том, что из всего контрольного множества некоторым образом выбирается подмножество, ошибка обнаружения по которому меньше, чем по всему множеству. В рамках данного подхода предложено три метода разной эффективности для решения этой задачи. Однако несмотря на то, что выбранное подмножество может быть достаточно велико (в [9] его размер варьировался приблизительно от 5 до 40 % от всего контрольного множества), все-таки значительная часть контейнеров не используется при подсчёте ошибки обнаружения.

В настоящей работе предлагается подход, позволяющий расширить идею предварительной фильтрации и добиться снижения ошибки обнаружения, которая будет вычислена уже по всему контрольному множеству, а не только по его части. Для этой цели вводится понятие интегрального классификатора, состоящего из набора отдельных

классификаторов, каждый из которых обрабатывает только те контейнеры, которые отфильтрованы для него. Теоретически интегральный классификатор может быть реализован разными способами. В работе предложен интегральный классификатор на основе сжатия данных. Обучающее множество разбивается на несколько частей в соответствии с коэффициентом их сжатия и после этого обучается такое же количество классификаторов, причём каждый из них обучается на своём подмножестве. Во время фазы тестирования контрольного множества очередной контейнер отправляется на классификатор, обученный на контейнерах, коэффициент сжатия которых наиболее близок к коэффициенту сжатия данного контейнера.

Идея использования коэффициента сжатия как критерия выбора классификатора возникла на основе известного факта о том, что проводить стегоанализ «шумных» изображений (noisy images) сложнее, чем изображений с большими областями приблизительно одного цвета (plain images). Как известно, изображения первого типа сжимаются хуже, чем второго. Соответственно наша догадка состоит в том, что классификаторы для плохо сжимаемых контейнеров должны обучаться на плохо сжимаемых контейнерах, и то же самое — для хорошо сжимаемых. Заметим, что сжатие данных уже использовалось в стегоанализе [10, 11], но эти работы посвящены либо обнаружению LSB-стеганографии, либо созданию количественных методов обнаружения, когда размер внедрения является неизвестной величиной, подлежащей определению. Второе отличие заключается в том, что в настоящей работе сжатие используется на предварительном этапе выбора классификатора, но не для построения самого метода стегоанализа непосредственно.

Эффективность предлагаемого подхода к обнаружению скрытой информации экспериментально продемонстрирована на изображениях из стандартизованной базы BOSSbase 1.01 [12], часто используемой специалистами по стеганографии и стегоанализу. Подлежащая обнаружению информация внедрялась при помощи современных методов адаптивной стеганографии HUGO [13], S-UNIWARD [14] и WOW [15]. В качестве эталонных значений ошибки обнаружения взяты лучшие на данный момент результаты В. Холуба и Дж. Фридрич [2]. Эксперименты подтвердили гипотезу о повышении эффективности стегоанализа при использовании нескольких классификаторов и добавлении фазы предварительного распределения контейнеров по ним. В зависимости от метода внедрения и его размера ошибку обнаружения удалось снизить на 0,05–0,16.

1. Описание предлагаемого интегрального классификатора

1.1. Общая схема интегрального классификатора

Процессы обучения интегрального классификатора и обнаружения скрытой информации с его помощью схематично изображены на рис. 1 и 2. Использование интегрального классификатора подразумевает выбор параметра L, определяющего количество составляющих его одиночных классификаторов; в вырожденном случае при $L = 1$ интегральный классификатор фактически становится одиночным (рис. 3).

На стадии обучения интегрального классификатора все обучающее множество разбивается на L непересекающихся подмножеств (возможно, различного размера) в соответствии с коэффициентами сжатия входящих в них контейнеров. Для определённости будем считать, что подмножество 1 содержит контейнеры с наибольшим коэффициентом сжатия, а подмножество L — с наименьшим.

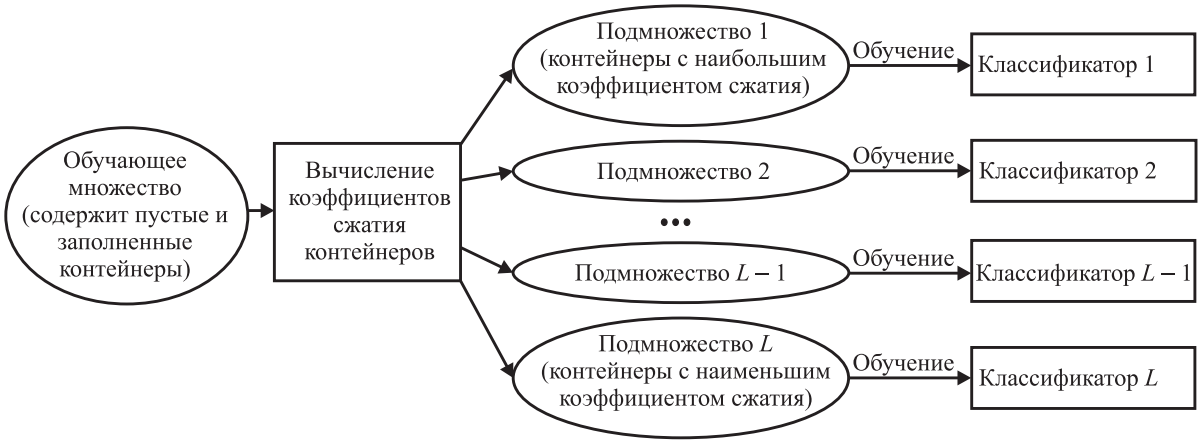

Рис. 1. Обучение интегрального классификатора, основанного на сжатии данных

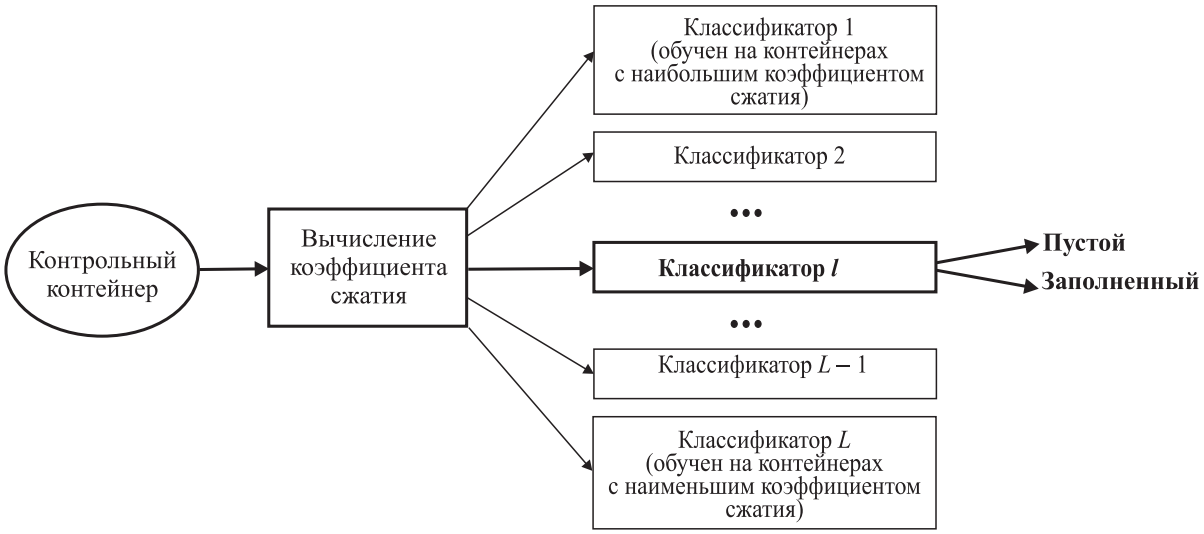

Рис. 2. Определение наличия/отсутствия скрытой в контейнере информации при помощи интегрального классификатора, основанного на сжатии данных (у контрольного контейнера вычисляется коэффициент сжатия, согласно которому выбирается классификатор, дающий ответ; этот классификатор обучен на контейнерах, имеющих коэффициенты сжатия, близкие к коэффициенту сжатия данного контрольного контейнера)

Рис. 3. Обнаружение скрытой информации при помощи одиночного классификатора

После разбиения необходимо взять L классификаторов и обучить каждый из них на контейнерах из соответствующего ему подмножества. На стадии тестирования контрольное множество также разбивается на L подмножеств в соответствии с коэффициентами сжатия, после чего контейнеры из этих подмножеств обрабатываются классификатором, обученным на контейнерах с близким коэффициентом сжатия. Если

контрольные контейнеры поступают по одному, то их можно обрабатывать «на лету», выбирая классификатор согласно заданным порогам для коэффициента сжатия.

В принципе, в описанной схеме разбиение обучающего множества и выбор классификатора может производиться не только на основе коэффициента сжатия, но и на основе любой характеристики, которую можно вычислить как функцию от контейнера в качестве аргумента.

1.2. Алгоритм обнаружения скрытой информации при помощи интегрального классификатора

Алгоритм 1 демонстрирует пошаговую работу интегрального классификатора. В качестве входных параметров алгоритм принимает обучающее множество $\mathcal{X}$ и контрольное множество $\mathcal{Y}$. На первых шагах выбираются: метод сжатия $\texttt{Compress}(\cdot)$; L — количество классификаторов, составляющих интегральный классификатор; размеры подмножеств, на которые разбивается обучающее множество $size_1, \ldots, size_L$ ($size_1 + \ldots + size_L = |\mathcal{X}|$); размеры подмножеств, на которые разбивается контрольное множество $size'_1, \ldots, size'_L$ ($size'_1 + \ldots + size'_L = |\mathcal{Y}|$).

Алгоритм 1. Алгоритм работы интегрального классификатора

1: **Функция** Интегральный-классификатор($\mathcal{X}$, $\mathcal{Y}$)
   $\mathcal{X}$ — обучающее множество, $\mathcal{Y}$ — контрольное множество
2: Выбрать функцию сжатия $\texttt{Compress}(\cdot)$.
3: Выбрать количество классификаторов L.
4: Выбрать размеры $size_l$, $l = 1, \ldots, L$, подмножеств, на которые разбивается обучающее множество ($size_1 + \ldots + size_L = |\mathcal{X}|$).
5: Выбрать размеры $size'_l$, $l = 1, \ldots, L$, подмножеств, на которые разбивается контрольное множество ($size'_1 + \ldots + size'_L = |\mathcal{Y}|$).
6: $(Subset_1, \ldots, Subset_L)$:=Разбить-множество($\mathcal{X}$,$\texttt{Compress}(\cdot)$,$L, size_1, \ldots, size_L$).
7: $(Detector_1, \ldots, Detector_L)$:=Обучить-классификаторы($L, Subset_1, \ldots, Subset_L$).
8: $(Subset'_1, \ldots, Subset'_L)$:=Разбить-множество($\mathcal{Y}$, $\texttt{Compress}(\cdot)$,$L, size'_1, \ldots, size'_L$).
9: $\mathcal{Y}_{\text{Пустые}} = \varnothing$; $\mathcal{Y}_{\text{Заполненные}} = \varnothing$.
10: **Для всех** $y \in \mathcal{Y}$
11: $detectorNumber$:=Номер-классификатора($y, L, Subset'_1, \ldots, Subset'_L$);
12: $detectionResult := Detector_{detectorNumber}(y)$.
13: **Если** $detectionResult = \text{Empty}$, **то**
14: $\mathcal{Y}_{\text{Пустые}} := \mathcal{Y}_{\text{Пустые}} \cup \{y\}$;
15: **иначе**
16: $\mathcal{Y}_{\text{Заполненные}} := \mathcal{Y}_{\text{Заполненные}} \cup \{y\}$.
17: **Вернуть** $\mathcal{Y}_{\text{Пустые}}$ — пустые контейнеры (согласно классификатору);
$\mathcal{Y}_{\text{Заполненные}}$ — заполненные контейнеры.

Функция Разбить-множество (алгоритм 2) возвращает L непересекающихся подмножеств, образующих разбиение множества Z. В интегральном классификаторе (алгоритм 1) эта функция вызывается дважды: в шаге 6 — с целью разбиения обучающего множества $\mathcal{X}$ и в шаге 8 — с целью разбиения контрольного множества $\mathcal{Y}$. На первом шаге данной функции каждый контейнер $z \in \mathcal{Z}$ сжимается, чтобы определить размер его архива $|\texttt{Compress}(z)|$. Далее все контейнеры сортируются согласно размерам архивов в порядке возрастания, отсортированный список обозначим $(z_{(1)}, z_{(2)}, \ldots, z_{(|\mathcal{Z}|)})$. Наконец, на последнем шаге формируются подмножества; первые

$size_1$ контейнеров помещаются в первое подмножество, следующие $size_2$ — во второе и т. д.; последние $size_L$ контейнеров образуют последнее подмножество.

Алгоритм 2. Разбиение множества на подмножества согласно коэффициенту сжатия

1: **Функция** Разбить-множество($\mathcal{Z}$, `Compress(·)`, L, $size_1$, …, $size_L$)
   $\mathcal{Z}$ — обучающее или контрольное множество, подлежащее разбиению;
   `Compress(·)` — функция сжатия;
   L — количество подмножеств (определяется количеством классификаторов);
   $size_l$, $l = 1, \ldots, L$ — размеры подмножеств ($size_1 + \ldots + size_L = |\mathcal{Z}|$).
2: **Для всех** $z \in \mathcal{Z}$
   вычислить $|\texttt{Compress}(z)|$.
3: Отсортировать контейнеры по неубыванию значения $|\texttt{Compress}(z)|$.
4: Сформировать L подмножеств множества $\mathcal{Z}$ следующим образом:
   $Subset_1 = \{z_{(1)}, \ldots, z_{(size_1)}\}$; $Subset_2 := \{z_{(size_1+1)}, \ldots, z_{(size_1+size_2)}\}$;
   $Subset_3 := \{z_{(size_1+size_2+1)}, \ldots, z_{(size_1+size_2+size_3)}\}$; …;
   $Subset_L := \{z_{(size_1+\ldots+size_{L-1}+1)}, \ldots, z_{(|\mathcal{Z}|)}\}$;
5: **Вернуть** $Subset_1$, $Subset_2$, …, $Subset_L$.

Функция Обучить-классификаторы (алгоритм 3) принимает L подмножеств обучающего множества и обучает на них L одиночных классификаторов различать пустые и заполненные контейнеры; каждое подмножество используется для обучения одного классификатора. Классификаторы могут быть различными, но в стегоанализе, как правило, используются классификаторы на основе метода опорных векторов (support vector machine) или ансамблевые (ensemble classifier). Эти классификаторы работают со специфичными для стегоанализа признаками, в частности, для изображений-контейнеров применяются SRM [1], PSRM [2], SPAM [16] и др.

Алгоритм 3. Обучение классификаторов, образующих интегральный классификатор

1: **Функция** Обучить-классификаторы($L, Subset_1, \ldots, Subset_L$)
   L — количество классификаторов;
   $\{Subset_l : l = 1, \ldots, L\}$ — разбиение обучающего множества.
2: Взять L необученных классификаторов $Detector_1$, $Detector_2$, …, $Detector_L$.
3: **Для всех** $l = 1, \ldots, L$
   обучить $Detector_l$ на множестве $Subset_l$.
4: **Вернуть** $Detector_1$, $Detector_2$, …, $Detector_L$.

Фаза обработки контрольного множества состоит из двух этапов: 1) выбор классификатора и 2) его применение. Выбор классификатора осуществляется функцией Номер-классификатора (алгоритм 4), которая по заданному контейнеру из контрольного множества $\mathcal{Y}$ возвращает номер классификатора, решение которого будет принято за решение интегрального классификатора. Номер определяется согласно разбиению контрольного множества.

Алгоритм 4. Выбор классификатора для обработки контрольного контейнера

Функция НОМЕР-КЛАССИФИКАТОРА(y, L, $Subset'_1$, $Subset'_2$, …, $Subset'_L$)
y — контейнер из контрольного множества;
L — количество классификаторов;
$\{Subset'_l : l = 1, \ldots, L\}$ — разбиение контрольного множества.
Для $l = 1, \ldots, L$
  Если $y \in Subset'_l$, **то**
    $detectorNumber := l$.
Вернуть $detectorNumber$.

2. Экспериментальные результаты

2.1. Инструментарий для проведения экспериментов

Изображения-контейнеры

Для выполнения экспериментов в качестве контейнеров использовались изображения из известного стандартизованного множества BOSSbase 1.01, опубликованного в рамках конкурса Break Our Steganographic System (BOSS) [12]. Данное множество изображений за последние годы стало де-факто стандартом при экспериментальном расчёте ошибок обнаружения методов стегоанализа [7, 8]. Множество BOSSbase 1.01 состоит из 10 000 изображений, полученных с семи различных фотокамер в формате RAW, которые затем были преобразованы в 8-битовый черно-белый режим и уменьшены до размера 512×512.

Методы внедрения информации

В экспериментах стеганографическое внедрение информации производилось тремя адаптивными методами: HUGO, WOW и S-UNIWARD, поскольку именно эти методы считаются наиболее трудно обнаружимыми на данный момент и именно по ним приведены лучшие из известных результатов обнаружения [2]. Идея адаптивного внедрения заключается в том, что позиции для внедрения выбираются не произвольно, а исходя из свойств изображения; при этом с большей вероятностью внедрение осуществляется в те области, где обнаружить информацию должно быть труднее. HUGO (Highly Undetectable Steganography) — это алгоритм внедрения, основанный на синдромном решетчатом декодировании (syndrome-trellis codes) [13]. WOW (Wavelet Obtained Weights) использует дискретное вейвлет-преобразование [15], а алгоритм S-UNIWARD [14] является упрощённым вариантом WOW.

Создание обучающего и контрольного множеств

Множество BOSSbase 1.01 состоит из 10 000 изображений. Создание обучающего и контрольного множеств на его основе осуществлялось следующим образом. Обозначим через $\mathcal{X}^p$ обучающее множество и через $\mathcal{Y}^p$ — контрольное, где индекс p указывает на размер внедрения в битах на пиксель (б/п).

Всё исходное множество разбивалось на две части $\mathcal{X}_0$ и $\mathcal{Y}_0$, где $|\mathcal{X}_0| = 7\,500$ и $|\mathcal{Y}_0| = 2\,500$. Затем во все изображения из $\mathcal{X}_0$ и $\mathcal{Y}_0$ внедрялось p б/п случайной информации с помощью одного из трёх методов (HUGO, S-UNIWARD и WOW). Полученные изображения с внедрённой информацией обозначим $\mathcal{X}_1^p$ и $\mathcal{Y}_1^p$ соответственно.

Далее обучающее и контрольное множества формировались следующим образом: $\mathcal{X}^p := \mathcal{X}_0 \cup \mathcal{X}_1^p$ и $\mathcal{Y}^p := \mathcal{Y}_0 \cup \mathcal{Y}_1^p$. Таким образом, $|\mathcal{X}^p| = 15\,000$ и $|\mathcal{Y}^p| = 5\,000$. Оба

множества содержат поровну пустых и заполненных контейнеров. Далее индекс p будем опускать (это не должно привести к разночтениям), а обучающее и контрольное множества будем обозначать через $\mathcal{X}$ и $\mathcal{Y}$ соответственно.

Используемые методы сжатия

В качестве метода сжатия использован известный свободно распространяемый архиватор PAQ, основанный на контекстном моделировании (context mixing model) и предсказании частичным совпадением (partial match). В экспериментах использована готовая реализация М. Махоуни [17]. Скрипт для запуска архиватора: paq -11.

Реализация классификатора

В качестве классификатора использована реализация метода опорных векторов (support vector machine) на языке программирования Python [18]. В данной реализации параметры, заданные по умолчанию, не изменялись, за исключением следующих: ядро — линейное, сокращение (srinking) — включено и штрафной параметр $C = 20\,000$. Параметры изменены с целью достижения приемлемой скорости работы программы.

Извлекаемые из изображений признаки для классификации

Для классификации из изображений извлекались так называемые SRM-признаки (Spatial Rich Model) [1], являющиеся одним из наилучших инструментов, применяемых в стегоанализе. Их более новый усовершенствованный вариант — PSRM-признаки (Projection Spatial Rich Model) [2] — лишь незначительно снижает ошибку обнаружения, но существенно увеличивает сложность реализации и замедляет работу программы вплоть до неприемлемых показателей. Пространство SRM-признаков имеет размерность 34,671.

Формула для вычисления ошибки обнаружения

Ошибка обнаружения вычислялась стандартным для современных работ способом [1, 3, 16, 19]: $P_E = (P_{FA} + P_{MD})/2$, где P_{FA} — вероятность ложных срабатываний (пустой контейнер принимается за заполненный); P_{MD} — вероятность пропуска обнаружения (заполненный контейнер принимается за пустой).

2.2. Результаты экспериментов

Приведём результаты экспериментов, которые демонстрируют, что использование интегрального классификатора позволяет повысить эффективность стегоанализа, другими словами, снизить ошибку обнаружения по сравнению с одиночным классификатором. Результаты экспериментов сравниваются со state-of-the-art значениями из [2], где реализован ансамблевый классификатор [3], который немного уступает в точности, но работает быстрее. Для убедительности результатов помимо этих данных вычислены ошибки для нашей реализации одиночного классификатора на основе метода опорных векторов и SRM-признаков; эти ошибки оказались близкими к ошибкам ансамблевого классификатора. Однако точность предлагаемого интегрального классификатора оказалась выше точности одиночных во всех случаях.

В таблице приведено сравнение ошибок обнаружения методов внедрения HUGO, WOW и S-UNIWARD для одиночных классификаторов и интегрального классификатора на основе метода опорных векторов и SRM-признаков. Параметры интегрального классификатора следующие: метод сжатия — PAQ; $L = 5$; $size_1 = size_2 = size_3 = size_4 = size_5 = 3\,000$; $size'_1 = size'_2 = size'_3 = size'_4 = size'_5 = 1\,000$. Выбор параметров обоснован некоторыми предварительными экспериментами, хотя, в целом, увеличение или уменьшение числа подмножеств слабо влияет на ошибку обнаружения.

Сравнение ошибок обнаружения различных методов на множестве изображений BOSSbase 1.01 при различных размерах внедрения (серым цветом выделены наименьшие ошибки для каждого размера внедрения); приведены ошибки, вычисленные отдельно по подмножествам

Вид классификатора	Контрольное множество	Размер внедрения		
		0,1 б/п	0,2 б/п	0,4 б/п
Внедрение методом HUGO [13]				
Интегральный классификатор	$Subset'_1$	0,01	0,01	0,00
	$Subset'_2$	0,17	0,05	0,01
	$Subset'_3$	0,21	0,11	0,03
	$Subset'_4$	0,34	0,18	0,08
	$Subset'_5$	0,46	0,30	0,19
	Всё контр. множ.	**0,24**	**0,13**	**0,06**
Одиночный классификатор (всё контрольное множество)	Наша реализация: SVM+SRM	**0,35**	0,27	0,15
	Holub, Fridrich [2]: Ансамблевый+SRM	0,36	0,25	0,12
	Holub, Fridrich [2]: Ансамблевый+PSRMQ1	**0,35**	**0,23**	**0,11**
Внедрение методом WOW [15]				
Интегральный классификатор	$Subset'_1$	0,02	0,01	0,00
	$Subset'_2$	0,20	0,08	0,01
	$Subset'_3$	0,25	0,13	0,06
	$Subset'_4$	0,30	0,18	0,11
	$Subset'_5$	0,44	0,29	0,20
	Всё контр. множ.	**0,24**	**0,13**	**0,08**
Одиночный классификатор (всё контрольное множество)	Наша реализация: SVM+SRM	**0,38**	**0,29**	0,21
	Holub, Fridrich [2]: Ансамблевый+SRM	0,39	0,31	0,19
	Holub, Fridrich [2]: Ансамблевый+PSRMQ1	**0,38**	**0,29**	**0,17**
Внедрение методом S-UNIWARD [14]				
Интегральный классификатор	$Subset'_1$	0,01	0,00	0,00
	$Subset'_2$	0,21	0,04	0,00
	$Subset'_3$	0,29	0,14	0,02
	$Subset'_4$	0,33	0,20	0,11
	$Subset'_5$	0,41	0,37	0,16
	Всё контр. множ.	**0,25**	**0,15**	**0,06**
Одиночный классификатор (всё контрольное множество)	Наша реализация: SVM+SRM	**0,37**	**0,30**	**0,17**
	Holub, Fridrich [2]: Ансамблевый+SRM	0,41	0,31	0,20
	Holub, Fridrich [2]: Ансамблевый+PSRMQ1	0,39	**0,30**	0,18

В таблице серым цветом выделены сравниваемые значения. Для каждого размера внедрения и каждого метода внедрения сравниваются значения ошибок интегрального классификатора с наименьшими ошибками среди всех реализаций одиночного классификатора, среди которых две реализации из [2] (ансаблевый классификатор с SRM- или PSRM-признаками) и одна наша реализация (метод опорных векторов с SRM-признаками). Например, для HUGO 0,1 б/п сравнивается 0,24 с 0,35, а для WOW 0,4 б/п — 0,08 с 0,17. Если две реализации дают одинаковую наименьшую ошибку, то выделены два значения (например, для HUGO 0,1 б/п две реализации дают ошибку 0,35).

Результаты показали, что интегральный классификатор снижает ошибку обнаружения. Наиболее впечатляющие результаты достигнуты при внедрениях HUGO 0,1 б/п, WOW 0,1 б/п, WOW 0,2 б/п, S-UNIWARD 0,1 б/п, S-UNIWARD 0,2 б/п, S-UNIWARD 0,4 б/п, где ошибка снизилась более чем на 0,1.

Заключение

В работе предложена концепция интегрального классификатора, предназначенного для повышения точности методов стегоанализа, которые основаны на машинном обучении. Данная концепция расширяет опубликованный ранее подход под названием «предварительная фильтрация» [9], позволяющий выбирать из контрольного множества (отфильтровывать) только те контейнеры, ошибка обнаружения для которых меньше, чем для всего контрольного множества. Хотя количество таких отфильтрованных контейнеров может быть достаточно велико, предварительная фильтрация в чистом виде не позволяет снизить ошибку обнаружения для всего контрольного множества. В то же время интегральный классификатор, предложенный в настоящей работе, предназначен для того, чтобы уменьшить ошибку обнаружения для всего контрольного множества.

Эффективность обнаружения скрытой информации при помощи интегрального классификатора продемонстрирована экспериментально для методов адаптивной стеганографии HUGO, WOW и S-UNIWARD на известном стандартизованном множестве изображений-контейнеров BOSSbase 1.01. Результаты экспериментов показали, что в зависимости от размера внедрения ошибка обнаружения уменьшилась на 0,05–0,16 по сравнению с лучшими известными авторам результатами.

Идея интегрального классификатора заключается в том, что для распознавания пустых и заполненных контейнеров обучается не один классификатор, а несколько, причём каждый из них предназначен для обработки контейнеров, обладающих определёнными свойствами. Так, в настоящей работе предложен интегральный классификатор, основанный на сжатии данных; контейнеры распределяются по отдельным составляющим его классификаторам согласно их коэффициентам сжатия. Для каждого контейнера интегральный классификатор выбирает наиболее подходящий одиночный классификатор, результат которого принимается за решение о заполненности или пустоте контейнера. В целом, методика распределения контейнеров по классификаторам может быть совершенно разной, и поиск других характеристик, согласно которым будут распределяться контейнеры по классификаторам для снижения ошибки обнаружения, может оказаться продолжением данного исследования.

Во время предварительных экспериментов мы проверили некоторые комбинации параметров, чтобы подобрать приемлемый с точки зрения эффективности вариант. Этого оказалось достаточно, чтобы превзойти точность известных методов стегоанализа. Однако для поиска оптимальных в различных случаях параметров, вероятно, может быть целесообразно провести дополнительное исследование теоретического характера и разработать методику их выбора.

В описанных экспериментах контрольное множество разбивалось на подмножества согласно коэффициентам сжатия и все контейнеры из одного подмножества обрабатывались одним и тем же классификатором. Если подлежащие обработке контейнеры поступают один за другим, то во избежание необходимости их накопления можно зафиксировать определённые пороги для коэффициентов сжатия и выбирать классификаторы согласно этим порогам для каждого отдельно взятого контейнера «на лету».

ЛИТЕРАТУРА

1. *Fridrich J.* Rich models for steganalysis of digital images // IEEE Trans. Inform. Forensics and Security. 2012. V. 7. No. 3. P. 868–882.
2. *Holub V. and Fridrich J.* Random projections of residuals for digital image steganalysis // IEEE Trans. Inform. Forensics and Security. 2013. V. 8. No. 12. P. 1996–2006.
3. *Kodovsky J., Fridrich J., and Holub V.* Ensemble classifiers for steganalysis of digital media // IEEE Trans. Inform. Forensics and Security. 2010. V. 7. No. 2. P. 434–444.
4. *Борисенко Б. Б.* Модификация карты Хотеллинга, нивелирующая влияние тренда, и её применение при обнаружении цифровых водяных знаков // Прикладная дискретная математика. 2010. №. 2. С. 42–58.
5. *Menori M. and Munir R.* Blind steganalysis for digital images using support vector machine method // Proc. IEEE Intern. Symp. Electronics and Smart Devices (ISESD). 2016. Bandung, Indonesia. IEEE. P. 132–136.
6. *Pevný T., Fridrich J., and Ker A.* From blind to quantitative steganalysis // IEEE Trans. Inform. Forensics and Security. 2010. V. 7. No. 2. P. 445–454.
7. *Cogranne R., Denemark T., and Fridrich J.* Theoretical model of the FLD ensemble classifier based on hypothesis testing theory // Proc. 6th IEEE Intern. Workshop on Inform. Forensics and Security (WIFS), 2014. Atlanta, GA, USA. IEEE. P. 167–172.
8. *Schöttle P., Korff S., and Böhme R.* Weighted stego-image steganalysis for naive content-adaptive embedding // Proc. 4th IEEE Intern. Workshop on Inform. Forensics and Security (WIFS). 2012. Tenerife, Spain. IEEE. P. 193–198.
9. *Монарев В. А., Пестунов А. И.* Повышение эффективности методов стегоанализа при помощи предварительной фильтрации контейнеров // Прикладная дискретная математика. 2016. №. 2. С. 87–99.
10. *Boncelet C., Marvel L., and Raqlin A.* Lossless compression-based steganalysis of LSB embedded images // Proc. 41st Ann. Conf. on Inform. Sciences and Systems (CISS). 2007. Baltimore, MD, USA. IEEE. P. 923–929.
11. *Monarev V. and Pestunov A.* A new compression-based method for estimating LSB replacement rate in color and grayscale images // Proc. 7th IEEE Intern. Conf. on Intelligent Inform. Hiding and Multimedia Signal Processing (IIH-MSP). 2011. Dalian, China. IEEE. P. 57–60
12. *Bas P., Filler T., and Pevný T.* Break our steganographic system — the ins and outs of organizing BOSS // LNCS. 2011. V. 6958. P. 59–70.
13. *Pevný T., Filler T., and and Bas P.* Using high-dimensional image models to perform highly undetectable steganography // LNCS. 2010. V. 6387. P. 161–177.
14. *Holub V. and Fridrich J.* Digital image steganography using universal distortion // Proc. 1st ACM Workshop on Inform. Hiding and Multimedia Security (IHMMSec). 2013. Montpellier, France. ACM. P. 59–68.
15. *Holub V. and Fridrich J.* Designing steganographic distortion using directional filters // Proc. 4th IEEE Intern. Workshop on Inform. Forensics and Security (WIFS). 2012. Tenerife, Spain. IEEE. P. 234–239.
16. *Pevný T., Bas P., and Fridrich J.* Steganalysis by subtractive pixel adjacency matrix // IEEE Trans. Inform. Forensics and Security. 2010. V. 5. No. 2. P. 215–224.
17. `mattmahoney.net/dc/text.html` — Large Text Compression Benchmark. 2017.
18. `scikit-learn.org` — scikit-learn: Machine Learning in Python. 2017.
19. *Monarev V. and Pestunov A.* A known-key scenario for steganalysis and a highly accurate detector within it // Proc. 10th IEEE Intern. Conf. on Intelligent Inform. Hiding and Multimedia Signal Processing (IIH-MSP). 2014. Kitakyushu, Japan. IEEE. P. 175–178.

REFERENCES

1. *Fridrich J.* Rich models for steganalysis of digital images. IEEE Trans. Inform. Forensics and Security, 2012, vol. 7, no. 3, pp. 868–882.
2. *Holub V. and Fridrich J.* Random projections of residuals for digital image steganalysis. IEEE Trans. Inform. Forensics and Security, 2013, vol. 8, no. 12, pp. 1996–2006.
3. *Kodovsky J., Fridrich J., and Holub V.* Ensemble classifiers for steganalysis of digital media. IEEE Trans. Inform. Forensics and Security, 2010, vol. 7, no. 2, pp. 434–444.
4. *Borisenko B. B.* Modifikatsiya karty KHotellinga, niveliruyushchaya vliyaniye trenda, i eye primeneniye pri obnaruzhenii tsifrovykh vodyanykh znakov [Modified Hotelling's chart excluding trend influence and its application for digital watermarks detection]. Prikladnaya Diskretnaya Matematika, 2010, no. 2, pp. 42–58. (in Russian)
5. *Menori M. and Munir R.* Blind steganalysis for digital images using support vector machine method. Proc. IEEE Intern. Symp. on Electronics and Smart Devices (ISESD), 2016, Bandung, Indonesia, IEEE, pp. 132–136.
6. *Pevný T., Fridrich J., and Ker A.* From blind to quantitative steganalysis. IEEE Trans. Inform. Forensics and Security, 2010, vol. 7, no. 2, pp. 445–454.
7. *Cogranne R., Denemark T., and Fridrich J.* Theoretical model of the FLD ensemble classifier based on hypothesis testing theory. Proc. 6th IEEE Intern. Workshop on Inform. Forensics and Security (WIFS), 2014, Atlanta, GA, USA, IEEE, pp. 167–172.
8. *Schöttle P., Korff S., and Böhme R.* Weighted stego-image steganalysis for naive content-adaptive embedding. Proc. 4th IEEE Intern. Workshop on Inform. Forensics and Security (WIFS), 2012, Tenerife, Spain, IEEE, pp. 193–198.
9. *Monarev V. A. and Pestunov A. I.* Povyshenie effektivnosti metodov stegoanaliza pri pomoshchi predvaritel'noy fil'tratsii konteynerov [Enhancing steganalysis accuracy via tentative filtering of stego-containers]. Prikladnaya Diskretnaya Matematika, 2016, no. 2, pp. 87–99. (in Russian)
10. *Boncelet C., Marvel L., and Raqlin A.* Lossless compression-based steganalysis of LSB embedded images. Proc. 41st Ann. Conf. on Inform. Sciences and Systems (CISS), 2007, Baltimore, MD, USA, IEEE, pp. 923–929.
11. *Monarev V. and Pestunov A.* A new compression-based method for estimating LSB replacement rate in color and grayscale images. Proc. 7th IEEE Intern. Conf. on Intelligent Inform. Hiding and Multimedia Signal Processing (IIH-MSP), Dalian, Chaina, IEEE, 2011, pp. 57–60
12. *Bas P., Filler T., and Pevný T.* Break our steganographic system — the ins and outs of organizing BOSS. LNCS, 2011, vol. 6958, pp. 59–70.
13. *Pevný T., Filler T., and Bas P.* Using high-dimensional image models to perform highly undetectable steganography. LNCS, 2010, vol. 6387, pp. 161–177.
14. *Holub V. and Fridrich J.* Digital image steganography using universal distortion. Proc. 1st ACM Workshop on Inform. Hiding and Multimedia Security (IHMMSec), 2013, Montpellier, France, ACM, pp. 59–68.
15. *Holub V. and Fridrich J.* Designing steganographic distortion using directional filters. Proc. 4th IEEE Intern. Workshop on Inform. Forensics and Security (WIFS), 2012, Tenerife, Spain, IEEE, pp. 234–239.
16. *Pevný T., Bas P., and Fridrich J.* Steganalysis by subtractive pixel adjacency matrix. IEEE Trans. Inform. Forensics and Security, 2010, vol. 5, no. 2, pp. 215–224.
17. `mattmahoney.net/dc/text.html` — Large Text Compression Benchmark, 2017.
18. `scikit-learn.org` — scikit-learn: Machine Learning in Python, 2017.

19. *Monarev V. and Pestunov A.* A known-key scenario for steganalysis and a highly accurate detector within it. Proc. 10th IEEE Intern. Conf. on Intelligent Inform. Hiding and Multimedia Signal Processing (IIH-MSP), 2014, Kitakyushu, Japan, IEEE, pp. 175–178.

МАТЕМАТИЧЕСКИЕ ОСНОВЫ КОМПЬЮТЕРНОЙ БЕЗОПАСНОСТИ

УДК 004.057.4

ОТКАЗУЕМЫЕ ГРУППОВЫЕ КОММУНИКАЦИИ В МОДЕЛИ ГЛОБАЛЬНОГО НЕОГРАНИЧЕННОГО ЗЛОУМЫШЛЕННИКА[1]

В. Ф. Шейдаев*, Д. Ю. Гамаюнов**

*Московский государственный университет имени М. В. Ломоносова, г. Москва, Россия
**Национальный исследовательский университет «Высшая школа экономики», г. Москва, Россия

Предлагается использовать децентрализованные отказуемые групповые коммуникации для обеспечения защищённости общения в модели нарушителя, имеющего доступ к инфраструктуре передачи данных, а также неограниченное финансирование. За основу взят существующий протокол групповых отказуемых коммуникаций multi-party Off-the-Record (mpOTR), допускающий работу на децентрализованном транспорте, его свойства совершенной прямой секретности улучшены за счёт введения процедуры продвижения ключевого материала (Key Ratcheting). Предложена полностью децентрализованная система защищённых групповых коммуникаций, обладающая свойствами отказуемости, согласованности текста переписки и улучшеными свойствами совершенной прямой секретности, а также способная противостоять Sybil-атакам. Соответствующее программное средство реализовано на языке JavaScript и обеспечивает защищённую передачу сообщений между браузерами в условиях отсутствия центрального сервера.

Ключевые слова: *безопасность коммуникаций, децентрализованные коммуникации, отказуемость.*

DOI 10.17223/20710410/40/6

DENIABLE GROUP COMMUNICATIONS IN THE PRESENCE OF GLOBAL UNLIMITED ADVERSARY

V. F. Sheidaev*, D. Y. Gamayunov**

*Lomonosov Moscow State University, Moscow, Russia
**National Research University Higher School of Economics, Moscow, Russia

E-mail: enr1g@seclab.cs.msu.su, dgamaunov@hse.ru

With this paper, we provide our research into the problem of secure communications in the presence of a global unlimited adversary. As a solution, we propose to use decentralised deniable communications. We have made use of our existing multi-party Off-the-Record protocol's modification which is able to function over decentralised

[1]Работа поддержана грантом УМНИК Фонда содействия развитию малых форм предприятий в научно-технической сфере.

transport. Its Perfect Forward Secrecy (PFS) features were improved by adding Key Ratcheting procedure to the protocol's flow. As a result, we propose a fully decentralised cryptosystem which has deniability and transcript consistency features, improved PFS and ability to resist the Sybil attack. We also give a detailed overview of the protocol model implementation in JavaScript designed to function in conditions of centralized server's absence.

Keywords: *secure communications, decentralised communications, deniability.*

Введение

В 2004 г. была опубликована работа Яна Голдберга и Никиты Борисова о протоколе отказуемых коммуникаций OTR — Off-The-Record (Communications) [1]. Цель протокола — устранить недостатки традиционных средств криптографически защищённого общения, которые, по мнению авторов [1], уже не способны обеспечить должного уровня защищённости. В частности, традиционные криптографические протоколы не обладают свойствами отказуемости и совершенной прямой секретности, например, протокол PGP (Pretty Good Privacy) и другие построенные на аналогичных концепциях протоколы используют сильную криптографическую подпись. Это позволяет подтвердить авторство сообщения не только получателю, но и произвольному третьему лицу. Таким образом, участие отправителя в переписке и авторство конкретного сообщения может быть доказано. PGP не способен обеспечить участникам коммуникации возможность подтверждения авторства ограниченным кругом лиц и в течение ограниченного времени (непосредственно во время коммуникации). Если Alice является информатором и не доверяет Bob'у, PGP не позволит ей надёжно аутентифицировать себя и при этом иметь возможность отрицать участие в беседе. Свойство конфиденциальности PGP опирается также на сохранение закрытого ключа в секрете. В случае компрометации долговременного ключевого материала будет скомпрометирована и вся зашифрованная при его помощи корреспонденция. В эпоху распространения мобильной техники потеря или кража ключевого материала особенно актуальны.

В последние годы особое внимание уделяется проблеме защиты частных коммуникаций в присутствии глобального злоумышленника (Global Passive Adversary), то есть в условиях асимметрии возможностей пользователя по обеспечению конфиденциальности своих данных и возможностей владельцев телекоммуникационной инфраструктуры по сбору и анализу данных всех пользователей [2 – 4]. В значительной степени увеличилось количество средств защищённого общения (мессенджеров), в том числе с поддержкой свойств совершенной прямой секретности и отказуемости. Начали проводиться сравнительные исследования в предметной области, в том числе академические [5, 6]. Анализ приведённых работ показывает, что на сегодняшний день среди защищённых мессенджеров отсутствуют практически применимые протоколы групповых отказуемых коммуникаций. Более того, большинство представленных решений являются централизованными. С точки зрения защиты коммуникаций в присутствии глобального злоумышленника любой централизованный сервис обмена сообщениями является возможной точкой компрометации, а в некоторых случаях, когда поставщиком сервиса является крупная глобальная компания (такая, как Facebook, Apple или Microsoft), сам поставщик сервиса является глобальным злоумышленником в рассматриваемой модели нарушителя. Отметим, что использование одного лишь end-to-end шифрования недостаточно. Центральные сервера занимаются пересылкой сообщений между абонентами, хранят метаданные либо владеют другой привлекательной

для злоумышленников информацией, например списками контактов пользователей и их телефонными номерами. Наконец, провайдеры централизованных коммуникаций зачастую используют обезличенные данные пользователей для маркетинговых целей. Всё это делает централизованную инфраструктуру средств общения потенциальной целью для злоумышленников разного масштаба.

Отметим, что разработка протокола децентрализованного защищённого группового общения является нетривиальной задачей, особенно в контексте отказуемых коммуникаций. Во-первых, задача обеспечения отказуемости для группы участников на сегодняшний день не имеет удовлетворительного по эффективности решения [7]. Во-вторых, децентрализация коммуникаций порождает необходимость в обеспечении согласованности текста переписки. Наконец, открытым остаётся вопрос распределения ключевого материала среди участников. PKI не подходит ввиду своей централизованной природы. В свою очередь, Web of Trust предполагает, что доверенные отпечатки ключей будут подписываться долговременными ключами, что не согласуется с концепциями отказуемых коммуникаций. Ранее авторами работы [8] был предложен протокол, являющийся развитием концепции протокола mpOTR (multi-party OTR) [9], и разработана его централизованная модельная реализация [10]. Отметим, что протокол поддерживал и децентрализованный режим функционирования, поэтому дальнейшие исследования были направлены на создание практически применимого протокола распределённого отказуемого общения, а также на улучшение свойств протокола.

В данной работе представлены результаты в области исследования и разработки защищённых групповых коммуникаций, а также децентрализованная модельная реализация, демонстрирующая работоспособность концепции [11].

1. Состояние предметной области

Область отказуемых средств общения является сравнительно молодой и не представлена большим количеством протоколов и прикладных реализаций. Наиболее удачными, с нашей точки зрения, являются Signal, GOTR и RetroShare, однако все они имеют те или иные недостатки.

Signal — протокол защищённого общения, создание которого было вдохновлено оригинальным OTR [12]. Он имеет множество улучшений в сравнении с OTR, связанных с ориентацией Signal на асинхронную среду передачи сообщений: Signal имеет более эффективную процедуру аутентификации [13] и улучшенный механизм продвижения ключевого материала (Key Ratcheting) [14]. Разработчиками Signal также были предприняты попытки создания группового протокола отказуемых коммуникаций [15]. Идея группового протокола Signal — установление попарных соединений между участниками, что порождает накладные расходы при передаче и хранении контекстов коммуникации со всеми участниками. Протокол Signal используется в одноимённом мессенджере, а также в WhatsApp, Facebook Messenger и Allo [16 – 18]. Из недостатков протокола отметим его централизованность, а также то, что учётные записи Signal привязаны к номерам мобильных телефонов, т. е. если Alice и Bob хотят поговорить друг с другом, то они должны раскрыть друг другу свои номера. Наконец, Signal не поддерживает согласованности передачи сообщений при групповых коммуникациях.

GOTR является одной из академических попыток разработки Multi-party Off-the-Record протокола [19]. При создании GOTR авторы ориентировались на устранение недостатка оригинального mpOTR: для изменения состава участников в mpOTR необходимо завершить текущий сеанс коммуникации и запустить новый [9]. В случае с GOTR применяются так называемые кольцевые ключи. Используемый протокол

сложен в описании и реализации и требует повышенного количества ресурсов в связи с тем, что каждый участник коммуникации создаёт по два виртуальных узла на каждое попарное соединение. Авторами [19] заявлено наличие модельной реализации протокола в виде плагина к известному мессенджеру, однако она не обнаружена в публичном доступе. Протокол поддерживает децентрализованную передачу данных, но вопросы обеспечения согласованности не поднимаются.

Наконец, RetroShare является полностью децентрализованным решением [20]. Он построен на принципах F2F (Friend-to-Friend — пиринговая сеть, в которой пользователи соединены напрямую только с доверенными узлами) и поддерживает работу через DHT (Distributed Hash Table — распределённая хеш-таблица, используется при построении распределённых сетей без трекеров). RetroShare применяет хорошо известные криптографические примитивы. Аутентификация в нём производится посредством PGP, а защищённая передача данных — при помощи последних версий TLS. Из последнего утверждения следует, что RetroShare поддерживает совершенную прямую секретность, но в то же время не поддерживает свойств отказуемости. Разработчиками не заявлена поддержка какой-либо модели консистентности передачи сообщений.

Подводя итог, стоит отметить, что среди рассмотренных протоколов нет практически применимого протокола децентрализованного отказуемого общения со свойствами согласованности текста переписки. Это подтверждает актуальность данной работы, где приведены основные результаты проекта, целью которого является создание протокола, удовлетворяющего этим требованиям.

2. Описание протокола

2.1. Модель нарушителя

Рассматривается модель глобального злоумышленника, основными целями которого являются нарушение конфиденциальности передаваемых данных и идентификация участников коммуникации. К нарушителям такого класса можно отнести, например, провайдеров облачных услуг и средств централизованных коммуникаций, а также организации, занимающиеся реализацией персональных данных пользователей.

Особенность подобной модели нарушителя в том, что злоумышленник может попытаться скомпрометировать ключевой материал участников коммуникации путём перебора, так как обладает значительными техническими и финансовыми возможностями, а также может записывать трафик коммуникаций на протяжении длительного времени. Например, в [21] продемонстрирована атака на протокол Diffie — Hellman Key Exchange (DHKE), позволяющая существенно сократить время нахождения дискретного логарифма. Стоит заметить, что многие реализации DHKE из соображений производительности используют фиксированные значения порядка группы и генератора, что позволяет злоумышленнику предвычислить этапы алгоритма дискретного логарифмирования, зависящие только от порядка группы. По мнению авторов [21], уже сегодня организация с неограниченным финансированием способна производить успешные атаки на DHKE с длиной ключа 1024 бит. Если в будущем по каким-то причинам (компрометация, злой умысел участников коммуникации) злоумышленник получит доступ к открытым текстам переписки, он также сможет попытаться привязать сеанс коммуникации к участникам.

Помимо того, что подобный злоумышленник может записывать и хранить транскрипт переписки (пусть даже шифрованный), он же может выполнять задачу взаимной идентификации пользователей. Например, пользователи будут идентифицировать друг друга при помощи номеров мобильных телефонов, которые существенно привя-

заны к своим владельцам, в частности, в большинстве стран номера телефонов регистрируются по удостоверяющим личность документам. Таким образом, злоумышленник в нашей модели может неотказуемо привязать ключевой материал пользователей к ним самим, например через номер телефона.

Подытоживая, сформулируем модель нарушителя следующим образом: в качестве нарушителя будем рассматривать организацию, обладающую существенными вычислительными мощностями и/или доступом к инфраструктуре передачи данных.

2.2. Краткое описание базового протокола

В [8] сформулированы требования к протоколу криптографически защищённых групповых коммуникаций, предложен удовлетворяющий этим требованиям протокол, а также проведена его формальная верификация (model checking). Приведём эти требования:

1) конфиденциальность;
2) целостность;
3) аутентичность;
4) отказуемость;
5) совершенная прямая секретность;
6) согласованность транскрипта.

Первые три свойства являются традиционными в криптографии, гораздо больший интерес представляют остальные — именно они обеспечивают достаточный уровень безопасности в контексте рассматриваемой модели нарушителя. Протокол [8] основан на multi-party Off-the-Record из работы [9]. Актуальность его модификации была вызвана тем, что оригинальный mpOTR имел лишь высокоуровневое описание и многие существенные вопросы реализации остались открытыми. В частности, в [9] не было предложено масштабируемой процедуры групповой аутентификации и не были решены проблемы, связанные с обеспечением согласованности транскрипта во время коммуникации. Предложенный в [8] протокол решает обе эти проблемы. Во-первых, в нём использована улучшенная процедура отказуемой групповой аутентификации IDSKE, имеющая сложность $4N$ по числу сообщений, в сравнении с $12N$ у mpOTR [22]. Во-вторых, для обеспечения согласованности транскрипта используется протокол OldBlue, позволяющий добросовестным участникам поддерживать причинную консистентность передаваемых сообщений [23]. Не будем подробно останавливаться на описании всего протокола, отметим лишь, что протокол описывает полный цикл фаз защищённого общения:

1) установление канала: участники устанавливают соединение между собой, вообще говоря, не защищённое;
2) аутентификация: участники совершают процедуру отказуемой групповой аутентификации. В результате участники вырабатывают сессионный ключ шифрования и отказуемые подписные ключи;
3) коммуникация: на данном этапе участники непосредственно пересылают друг другу сообщения;
4) завершение: безопасное удаление из памяти сессионного ключа и публикация отказуемых подписных ключей.

2.3. Продвижение ключевого материала

Ещё одним недостатком протокола mpOTR является то, что он не обладает свойством совершенной прямой секретности во время коммуникации. Подобное свойство полезно для протокола в случае, если по каким-то причинам сессионный ключ шиф-

рования был скомпрометирован во время коммуникации, т. е. пока он ещё существовал в памяти устройств участников сеанса общения. Факт использования одного ключа на весь сеанс коммуникации отличает mpOTR от OTR [1] или его аналогов [14], в которых новый эфемерный ключ шифрования вырабатывается каждые несколько сообщений, что существенно уменьшает последствия возможной компрометации.

Данный процесс в литературе часто называют Key Ratcheting [5, 14]. Мы будем называть его *продвижением ключевого материала*. Будем обозначать i-й сессионный ключ sk_i. Множество сообщений, которые компрометируются при компрометации ключа sk_i, будем называть *окном компрометации*. Отметим, что в окно компрометации могут входить не только сообщения, которые непосредственно были зашифрованы ключом sk_i, но и сообщения, зашифрованные предыдущими и/или последующими ключами. Это связано с тем, что компрометация sk_i может повлечь за собой компрометацию других ключей в зависимости от выбранной стратегии генерации ключей.

Во введённых терминах цель процедуры продвижения ключевого материала можно сформулировать как минимизацию окна компрометации. В случае с общением двух собеседников задача имеет эффективное решение. В частности, в [5, 14] описываются различные подходы к организации продвижения ключевого материала. Их сравнение можно найти на рис. 1: Naïve KDF Ratchet соответствует алгоритму Silent Circle, Double Ratchet — протоколу Axolotl из [14], а OTR Ratchet — продвижению ключевого материала в оригинальном OTR [1]. Как можно видеть, наиболее удачным с точки зрения минимизации окна компрометации является смешанный подход Double Ratchet, так как при этом так же, как и в случае Naïve KDF Ratchet, каждое сообщение шифруется уникальным ключом и при первой же возможности вырабатывается новый сессионный ключ, независящий от предыдущих.

При описании алгоритма продвижения ключевого материала будем использовать следующие понятия:

— sk_i — мастер-ключ, вырабатываемый в результате групповой процедуры генерации ключа. В качестве неё выбран прокол Бурместера — Десмедта [24], так как в сравнении с другими известными протоколами для симметричной групповой выработки общего секрета, например групповым протоколом Диффи — Хеллмана, он имеет наименьшее число раундов;
— sk_i^n — ключ n-го сообщения, отправленного после порождения мастер-ключа sk_i.

В предложенной реализации для порождения ключа sk_i^n использована схема $Hash(sk_i + Hash(n))$. Она отличается от, например, применяемой в Axolotl и Silent Circle, где новый ключ зависит только от предыдущего, то есть $sk_i^n = Hash(sk_i^{n-1})$, $sk_i^0 = sk_i$. В выбранной схеме при компрометации ключа sk_i^n злоумышленнику становится доступно только одно сообщение, в то время как во второй — все последующие до очередной процедуры групповой выработки ключа.

Процедура продвижения работает следующим образом: в начале переписки сразу после процедуры аутентификации счётчики текущего сообщения n и текущего мастер-ключа i инициализируются нулём, а начальный мастер-ключ sk_0 — общим секретом, выработанным в ходе работы IDSKE [22].

При отправке сообщение шифруется ключом sk_i^n. В метаданные сообщения добавляются значения nr и ir, соответствующие текущим значениям n и i, после чего счётчик n увеличивается на единицу. При получении сообщения из его метаданных извлекаются значения счётчиков nr и ir, формируется ключ sk_{ir}^{nr} и сообщение расшифровывается.

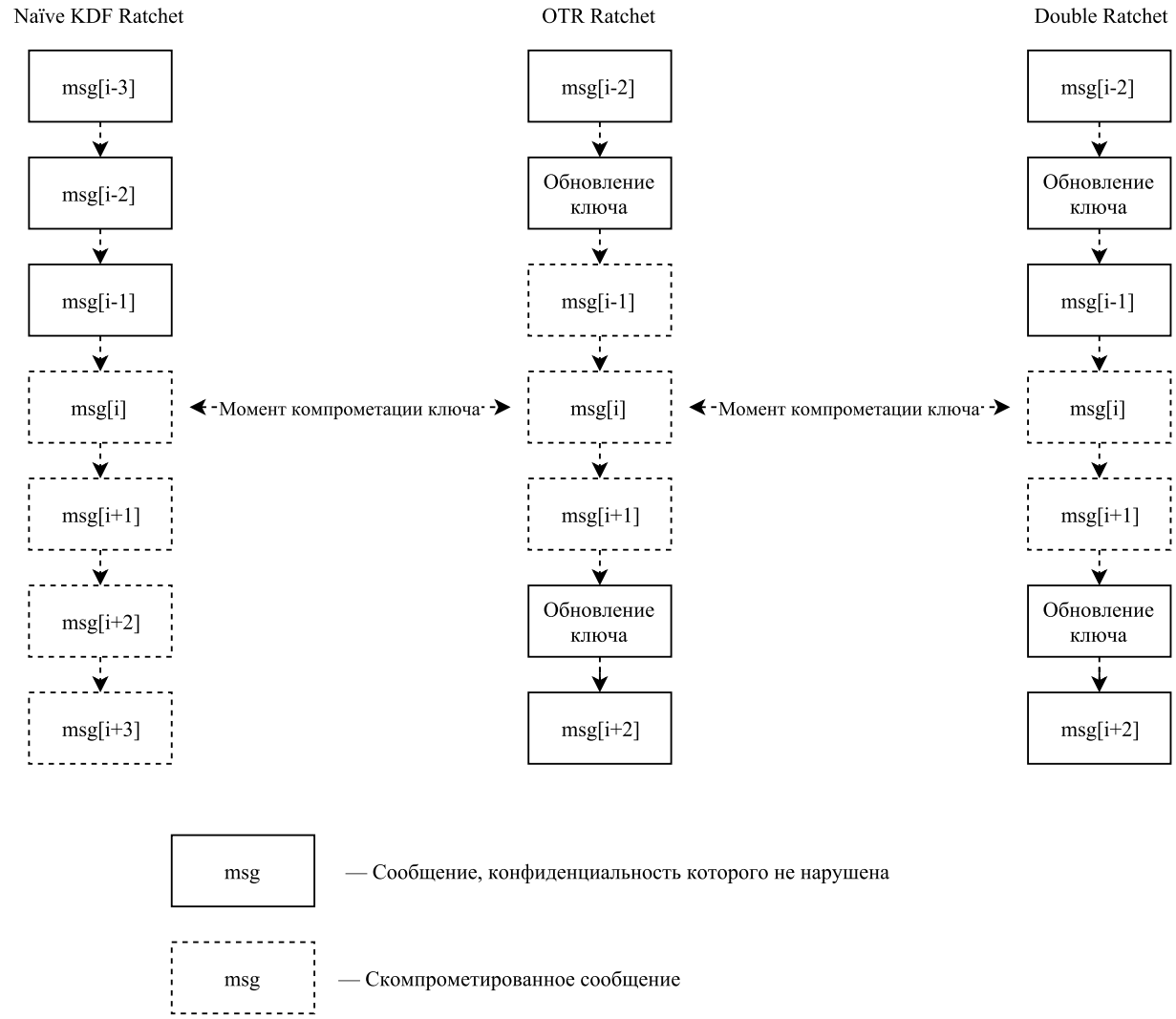

Рис. 1. Сравнение различных подходов к продвижению ключевого материала

Каждые N сообщений или T секунд производится процедура обновления текущего сессионного ключа при помощи алгоритма Бурместера — Десмедта и счётчик сообщений n сбрасывается в ноль. Предыдущие мастер-ключи sk_i хранятся в памяти до тех пор, пока не будет доставлено последнее зашифрованное при их помощи сообщение. В протоколе поддерживается свойство согласованности текста переписки, что позволяет знать, когда пришло последнее сообщение, зашифрованное старым ключом.

2.4. Проблемы аутентификации в условиях децентрализованной среды

При построении полностью децентрализованной инфраструктуры передачи данных для защищённого мессенджера возникает множество проблем. Во-первых, важно рассмотреть вопрос устойчивости предлагаемой схемы к sybil-атакам [25]. Для решения данной задачи ранее предлагались решения, основанные на использовании удостоверяющих центров либо установлении лимитов на число присоединяемых узлов, чтобы ограничить возможности атакующего по внедрению своих узлов. Оба подхода неприменимы в рассматриваемой модели нарушителя, так как требуют высокой степени централизации. Более поздние работы, нацеленные на построение P2P-сетей на DHT, используют в качестве вспомогательной меры знания о социальных связях участников

сети [26, 27]. Однако и эти подходы имеют недостатки, так как предполагают наличие свойств сильной связности у графа социальных связей, а также известную оценку размеров децентрализованной сети. При реализации мы предлагаем использовать модификацию классического DHT протокола Chord [28]. Данный протокол примечателен тем, что имеет минимальные предположения о сети, прост в реализации, а также достаточно эффективен при большом количестве зловредных узлов.

Во-вторых, встаёт вопрос распределения ключей между участниками коммуникации. Рассмотрены три подхода: Public Key Infrastructure (PKI), Web Of Trust (WoT) и Trust On First Use (TOFU). Ввиду стремления максимально отказаться от узких мест доверия (так называемых trust bottleneck), решено не рассматривать PKI из-за его централизованной природы. Что касается WoT, то в нём предполагается, что пользователи будут использовать неотказуемые подписи публичных ключей своих собеседников. Это не вступает в прямое противоречие со свойствами безопасности нашего протокола, однако наличия подобных подписей всё же хотелось бы избежать, когда речь идёт о протоколе отказуемого общения (не столько с точки зрения математической, сколько с точки зрения практического использования протокола). TOFU видится наиболее перспективным способом обмена ключами. Суть протокола заключается в том, что стороны обмениваются отпечатками ключей по третьему каналу, а позже при первом установлении соединения сравнивают предъявленные отпечатки с полученными ранее и запоминают свой выбор на соединяющихся устройствах. Таким образом, участники хранят у себя на устройствах список доверенных публичных ключей. Данный подход хорош минимальностью инфраструктуры и отсутствием какой-либо централизации. Наибольшее распространение TOFU получил в протоколе SSH. При использовании TOFU немаловажным аспектом является используемый тип отпечатка ключа. Существуют исследования, посвящённые разработке удобных в использовании (чтении и сравнении «на глаз») отпечатков ключей [29 – 33].

Мы решили остановиться на подходе [33], так как он не снижает уровня безопасности в сравнении с классическими отпечатками ключей, но при этом является более читаемым, потому что содержит базовые слова английского языка. Оригинальный OTR предоставляет также возможность проверить личность собеседника при помощи секретного вопроса. Для этого используются схемы доказательства с нулевым разглашением, позволяющие проверить знание стороной общего секрета без раскрытия информации о нём. В OTR с этой целью применяется протокол SMP (Socialist Millionaires Protocol) [34]. Мы решили не отходить от этой практики и также предоставлять пользователю возможность подтверждения личности собеседника при помощи SMP без каких-либо изменений в протоколе.

3. Описание реализации

Средство коммуникации, в котором реализован предложенный протокол, представляет собой децентрализованное веб-приложение на языке JavaScript. Узлы устанавливают соединение друг с другом при помощи сервера-трекера. Он используется только для трансляции идентификаторов пользователей в IP-адреса, сообщения пользователей передаются напрямую. Особый случай представляют пользователи, использующие протокол NAT (Network Address Translation — «преобразование сетевых адресов» — механизм в сетях TCP/IP, позволяющий преобразовывать IP-адреса транзитных пакетов). Для них установление прямого соединения не всегда возможно, однако для подобных ситуаций WebRTC поддерживает протокол TURN, позволяющий воспользоваться сервером для ретрансляции.

Узлы соединяются друг с другом в топологию «каждый с каждым». При необходимости выполнить какой-нибудь централизованный алгоритм, например перезапустить коммуникацию в случае отсоединения одного из участников, лидер (т. е. участник, координирующий выполнение централизованного алгоритма) выбирается следующим образом: каждый узел сортирует уникальные идентификаторы всех участников коммуникации и выбирает наибольший из них. Узел с максимальным идентификатором объявляется лидером, при этом для его определения не потребовалась отправка ни одного сообщения.

Для работы с WebRTC используется библиотека с открытым исходным кодом peerjs [35]. Вместе с ней поставляется публичный сервер-трекер, который при необходимости можно развернуть самостоятельно, в том числе в локальной сети.

3.1. Характеристики модельной реализации

На данный момент разработана модельная реализация протокола на языке JavaScript с использованием пиринговой технологии WebRTC [11]. Реализация является децентрализованной, однако для простоты транспортная часть реализована с использованием трекера [35], а не DHT. Алгоритм распределённого транспорта [28] планируется реализовать в будущем.

Проведено исследование свойств модельной реализации в различных сценариях использования. Во время исследования производилось профилирование на предмет того, сколько времени затрачено на криптографические вычисления, а сколько на сетевые операции.

Установление соединения: n пользователей устанавливают первичное соединение друг с другом и проводят процедуру взаимной аутентификации. При этом между участниками используется тип соединения каждый-с-каждым, то есть не предусмотрена ретрансляция. Результаты приведены в табл. 1.

Таблица 1

Время установления соединения, мс

n	3	5	7	10
Установление первичного соединения	153	609	700	1005
Фаза аутентификации	1712	1987	2357	3823
Фаза аутентификации: сетевые операции	1011	1254	1437	2761
Фаза аутентификации: криптографические вычисления	701	733	920	1062

Время приёма-передачи: измеряется время, за которое участник U_i рассылает всей группе сообщение, а другой участник U_j отвечает ему (табл. 2). Время окончания фиксируется после доставки ответного сообщения всем участникам. При этом предполагается отсутствие потерь сообщений на каналах связи. Время между узлами синхронизируется при помощи локального NTP-сервера (Network Time Protocol — сетевой протокол для синхронизации внутренних часов компьютера с использованием сетей с переменной задержкой).

Таблица 2

Время приёма-передачи (RTT), мс

n	3	5	7	10
RTT	76	98	112	120
RTT: сетевые операции	74	95	109	117
RTT: криптографические вычисления	2	3	3	3

Время повторного соединения после разрыва/отключения: в первоначальной конфигурации имеется n участников, далее участник U_n штатно отсоединяется от остальных либо имитируется сетевой разрыв с другими участниками. Измеряется время, через которое коммуникация возвращается в консистентное состояние (табл. 3).

Таблица 3

Восстановление коммуникации после разрыва, мс

n	3	5	7	10
Обнаружение разрыва	4119	5964	4712	5001
Обнаружение отсоединения	5	12	17	19
Перезапуск чата	1472	1719	2374	3773
Сумма	1477	1731	2391	3792
Сумма (разрыв)	5591	7683	7086	8774

3.2. Особенности платформы

Браузеры зачастую воспринимаются как среда, непригодная для построения криптографически защищённых приложений. Подобное мнение сформировано, в частности, из-за того, что JavaScript не имел доступа к источникам достаточной энтропии, необходимых типов данных для работы с большими числами, а также не мог обеспечить константного выполнения некоторых операций для защиты от атак по сторонним каналам; наконец, веб-приложения нельзя считать безопасным способом доставки контента ввиду их динамичности [36, 37]. Поэтому эта тема длительное время обходилась академическими исследованиями стороной.

Справедливо будет отметить, что в последние несколько лет ситуация существенно изменилась в лучшую сторону, как и мнение исследователей по данному вопросу [38, 39]. Во-первых, JavaScript получил поддержку криптографически стойкой генерации псевдослучайных чисел, а также API для работы с криптографическими примитивами [40]. Во-вторых, увеличилось количество библиотек, реализующих различные криптографические примитивы, в том числе академических и активно сопровождаемых [41]. Наконец, на вторую половину 2017 г. более 70 % браузеров в мире поддерживают технологию WebRTC [42]. Это сложный сетевой протокол, позволяющий браузерам устанавливать друг с другом end-to-end соединение, свойства безопасности которого высоко оценены [43].

Что касается атак по сторонним каналам, то данный вопрос исследован в работах [44, 45]. Показано, что контроль злоумышленником одной из вкладок браузера может позволить определить, какой сайт из списка известных был загружен в другой вкладке, какие клавиши были нажаты, какой документ отрисовывался в соседнем окне, наконец, даже определить, что в подключенном шифрованном диске есть скрытый раздел. Атаки проводились при помощи профилирования состояния L3-кэша процессора, что стало возможным благодаря поддержке High Resolution Timer в JavaScript. Это позволяет атаковать не только другие вкладки и окна браузера, но также и приложения, запущенные, например, в виртуальной машине. То есть данная проблема охватывает вычислительные устройства в целом, а не только браузеры. Её решение возможно при помощи модификации аппаратного обеспечения либо, что более вероятно, введения ограничения на использование High Resolution Timer в JavaScript-приложениях.

Специфических атак по сторонним каналам, позволяющих злоумышленнику получить данные из другой вкладки браузера, нами не обнаружено. В целом, это позволяет

сделать вывод о жизнеспособности криптографических приложений на базе стека Web-технологий.

Заключение

Представлено описание протокола защищённых групповых коммуникаций со свойством отказуемости, а также его улучшения, связанные с обновлением ключевого материала. Протокол реализован на языке JavaScript с использованием пиринговой технологии WebRTC, результаты исследования базовых сценариев представлены в работе.

Мы связываем дальнейшее развитие свойств протокола с улучшением свойств удобства пользования, решением вопроса взаимной аутентификации пользователей в контексте отказуемых коммуникаций и переходом на использование децентрализованного транспорта без использования трекера, например на основе DHT.

ЛИТЕРАТУРА

1. *Borisov N., Goldberg I., and Brewer E.* Off-the-record communication, or, why not to use PGP // Proc. Workshop on Privacy in the Electronic Society. ACM, 2004. P. 77–84.
2. `http://www.bbc.com/news/world-us-canada-22837100` — Profile: Edward Snowden, BBC News.
3. `http://www.globalissues.org/article/802/surveillance-state` — Surveillance State: NSA Spying and more.
4. `http://www.theguardian.com/media/2015/jul/02/wikileaks-us-spied-on-angela-merkels-ministers-too-says-german-newspaper` — WikiLeaks: US spied on Angela Merkel's ministers too, says German newspaper, The Guardian.
5. *Unger N. et al.* SoK: secure messaging // Proc. IEEE Symp. Security and Privacy (SP). 2015. P. 232–249.
6. `https://www.eff.org/secure-messaging-scorecard` — Electronic Frontier Foundation. Secure Messaging Scorecard.
7. *Unger N. and Goldberg I.* Deniable key exchanges for secure messaging // Proc. 22nd SIGSAC Conf. Computer and Communications Security. ACM, 2015. P. 1211–1223.
8. *Коростелева М. В., Гамаюнов Д. Ю.* Обеспечение криптографически защищенных групповых коммуникаций с функцией отказуемости // Проблемы информационной безопасности. Компьютерные системы. 2014. №. 3. С. 74–79.
9. *Goldberg I. et al.* Multi-party off-the-record messaging // Proc. 16th Conf. Computer and Communications Security. ACM, 2009. P. 358–368.
10. `https://github.com/maria-msu-seclab/mpotrDevelopment` — Moscow State University Seclab mpOTR.
11. `https://bitbucket.org/Enr1g/p2p_mpotr.js` — p2p_mpOTR.js.
12. `https://whispersystems.org` — Open Whisper Systems.
13. `https://whispersystems.org/blog/simplifying-otr-deniability/` — Simplifying OTR deniability.
14. `https://whispersystems.org/blog/advanced-ratcheting/` — *Marlinspike M.* Advanced Cryptographic Ratcheting.
15. `https://whispersystems.org/blog/private-groups/` — *Marlinspike M.* Private Group Messaging.
16. `https://whispersystems.org/blog/facebook-messenger/` — Facebook Messenger deploys Signal Protocol for end-to-end encryption.
17. `https://whispersystems.org/blog/allo/` — Open Whisper Systems partners with Google on end-to-end encryption for Allo.

18. https://whispersystems.org/blog/whatsapp-complete/ — WhatsApp's Signal Protocol integration is now complete.
19. *Liu H., Vasserman E. Y., and Hopper N.* Improved group off-the-record messaging // Proc. 12th Workshop on Privacy in the Electronic Society. ACM, 2013. P. 249–254.
20. https://retroshare.readthedocs.io/en/latest/concept/topology/ — RetroShare Docs. Topology.
21. *Adrian D. et al.* Imperfect forward secrecy: How Diffie — Hellman fails in practice // Proc. 22nd SIGSAC Conf. Computer and Communications Security. ACM, 2015. P. 5–17.
22. *Van Gundy M.* Improved Deniable Signature Key Exchange for mpOTR. http://matt.singlethink.net/projects/mpotr/improved-dske.pdf. 2013.
23. *Van Gundy M. D. and Chen H.* OldBlue: Causal Broadcast in a Mutually Suspicious Environment (Working Draft). http://matt.singlethink.net/projects/mpotr/oldblue-draft.pdf. 2012.
24. *Burmester M. and Desmedt Y.* A secure and scalable group key exchange system // Inform. Proc. Lett. 2005. V. 94. No. 3. P. 137–143.
25. *Douceur J. R.* The sybil attack // Intern. Workshop on Peer-to-Peer Systems. Berlin; Heidelberg: Springer, 2002. P. 251–260.
26. *Lesniewski-Laas C.* A Sybil-proof one-hop DHT // Proc. 1st Workshop on Social Network Systems. ACM, 2008. P. 19–24.
27. *Lesniewski-Laas C. and Kaashoek M. F.* Whanau: A sybil-proof distributed hash table // Proc. 7th USENIX Conf. on Networked Systems Design and Implementation (NSDI'10). 2010. P. 111–126.
28. *Danezis G. et al.* Sybil-resistant DHT routing // Europ. Symp. Research in Computer Security. Berlin; Heidelberg: Springer, 2005. P. 305–318.
29. *Loss D., Limmer T., and von Gernler A.* The Drunken Bishop: An Analysis of the OpenSSH Fingerprint Visualization Algorithm. http://dirk-loss.de/sshvis/drunken_bishop.pdf. 2009.
30. https://github.com/trevp/keyname — Keyname format for public-key fingerprints.
31. https://github.com/tomrittervg/crypto-usability-study/ *Ritter T. et al.* Crypto Usability Study.
32. https://tools.ietf.org/html/draft-miers-tls-sas-00 — Short Authentication Strings for TLS, Internet Draft.
33. http://philzimmermann.com/docs/PGP_word_list.pdf — PGP word list.
34. *Alexander C. and Goldberg I.* Improved user authentication in off-the-record messaging // Proc. Workshop on Privacy in Electronic Society. ACM, 2007. P. 41–47.
35. http://peerjs.com — The PeerJS Library.
36. https://rdist.root.org/2010/11/29/final-post-on-javascript-crypto/ *Lawson N.* Final post on Javascript crypto.
37. https://www.nccgroup.trust/us/about-us/newsroom-and-events/blog/2011/august/javascript-cryptography-considered-harmful/ *Ptacek T.* Javascript Cryptography Considered Harmful.
38. https://nadim.computer/2013/05/23/critique-javascript-cryptography.html. *Kobeissi N.* Thoughts on Critiques of JavaScript Cryptography.
39. http://blog.kotowicz.net/2014/07/js-crypto-goto-fail.html?m=1 — JS crypto goto fail?
40. https://www.w3.org/TR/WebCryptoAPI/#security-considerations — Web Cryptography API, W3C Recommendation.

41. *Stark E., Hamburg M., and Boneh D.* Symmetric cryptography in javascript // Proc. Computer Security Applications Conf. (ACSAC'09). IEEE, 2009. P. 373–381.
42. http://caniuse.com/#search=webrtc — Can I Use? WebRTC Peer-to-peer connections.
43. http://webrtc-security.github.io — A Study of WebRTC Security.
44. *Hornby T.* Side-Channel Attacks on Everyday Applications: Distinguishing Inputs with FLUSH+RELOAD. https://www.semanticscholar.org/paper/Side-Channel-Attacks-on-Everyday-Applications-Dist-Hornby/a5ea83ad9abffe6c44b93617728e5f06f73bb9be?tab=citations. 2016.
45. *Oren Y. et al.* The spy in the sandbox: Practical cache attacks in javascript and their implications // Proc. 22nd SIGSAC Conf. Computer and Communications Security. ACM, 2015. P. 1406–1418.

REFERENCES

1. *Borisov N., Goldberg I., and Brewer E.* Off-the-record communication, or, why not to use PGP. Proc. Workshop on Privacy in the Electronic Society, ACM, 2004, pp. 77–84.
2. http://www.bbc.com/news/world-us-canada-22837100 — Profile: Edward Snowden, BBC News.
3. http://www.globalissues.org/article/802/surveillance-state — Surveillance State: NSA Spying and more.
4. http://www.theguardian.com/media/2015/jul/02/wikileaks-us-spied-on-angela-merkels-ministers-too-says-german-newspaper — WikiLeaks: US spied on Angela Merkel's ministers too, says German newspaper, The Guardian.
5. *Unger N. et al.* SoK: secure messaging. Proc. IEEE Symp. Security and Privacy (SP), 2015, pp. 232–249.
6. https://www.eff.org/secure-messaging-scorecard — Electronic Frontier Foundation. Secure Messaging Scorecard.
7. *Unger N. and Goldberg I.* Deniable key exchanges for secure messaging. Proc. 22nd SIGSAC Conf. Computer and Communications Security, ACM, 2015, pp. 1211–1223.
8. *Korosteleva M. V. and Gamayunov D. Yu.* Obespecheniye kriptograficheski zashchishchennykh gruppovykh kommunikatsiy s funktsiyey otkazuyemosti [Protocol for secure group communications with deniability features]. Problemy Informatsionnoy Bezopasnosti. Komp'yuternyye Sistemy, 2014, no. 3, pp. 74–79.
9. *Goldberg I. et al.* Multi-party off-the-record messaging. Proc. 16th Conf. Computer and Communications Security, ACM, 2009, pp. 358–368.
10. https://github.com/maria-msu-seclab/mpotrDevelopment — Moscow State University Seclab mpOTR.
11. https://bitbucket.org/Enr1g/p2p_mpotr.js — p2p_mpOTR.js.
12. https://whispersystems.org — Open Whisper Systems.
13. https://whispersystems.org/blog/simplifying-otr-deniability/ — Simplifying OTR deniability.
14. https://whispersystems.org/blog/advanced-ratcheting/ — *Marlinspike M.* Advanced Cryptographic Ratcheting.
15. https://whispersystems.org/blog/private-groups/ — *Marlinspike M.* Private Group Messaging.
16. https://whispersystems.org/blog/facebook-messenger/ — Facebook Messenger deploys Signal Protocol for end-to-end encryption.
17. https://whispersystems.org/blog/allo/ — Open Whisper Systems partners with Google on end-to-end encryption for Allo.

18. https://whispersystems.org/blog/whatsapp-complete/ — WhatsApp's Signal Protocol integration is now complete.
19. *Liu H., Vasserman E. Y., and Hopper N.* Improved group off-the-record messaging. Proc. 12th Workshop on Privacy in the Electronic Society, ACM, 2013, pp. 249–254.
20. https://retroshare.readthedocs.io/en/latest/concept/topology/ — RetroShare Docs. Topology.
21. *Adrian D. et al.* Imperfect forward secrecy: How Diffie — Hellman fails in practice. Proc. 22nd SIGSAC Conf. Computer and Communications Security, ACM, 2015, pp. 5–17.
22. *Van Gundy M.* Improved Deniable Signature Key Exchange for mpOTR. http://matt.singlethink.net/projects/mpotr/improved-dske.pdf. 2013.
23. *Van Gundy M. D. and Chen H.* OldBlue: Causal Broadcast in a Mutually Suspicious Environment (Working Draft). http://matt.singlethink.net/projects/mpotr/oldblue-draft.pdf. 2012.
24. *Burmester M. and Desmedt Y.* A secure and scalable group key exchange system. Inform. Proc. Lett., 2005, vol. 94, no. 3, pp. 137–143.
25. *Douceur J. R.* The sybil attack. Intern. Workshop on Peer-to-Peer Systems, Berlin, Heidelberg, Springer, 2002, pp. 251–260.
26. *Lesniewski-Laas C.* A Sybil-proof one-hop DHT. Proc. 1st Workshop on Social Network Systems, ACM, 2008, pp. 19–24.
27. *Lesniewski-Laas C. and Kaashoek M. F.* Whanau: A sybil-proof distributed hash table. Proc. 7th USENIX Conf. on Networked Systems Design and Implementation (NSDI'10), 2010, pp. 111–126.
28. *Danezis G. et al.* Sybil-resistant DHT routing. Europ. Symp. Research in Computer Security, Berlin, Heidelberg, Springer, 2005, pp. 305–318.
29. *Loss D., Limmer T., and von Gernler A.* The Drunken Bishop: An Analysis of the OpenSSH Fingerprint Visualization Algorithm. http://dirk-loss.de/sshvis/drunken_bishop.pdf. 2009.
30. https://github.com/trevp/keyname — Keyname format for public-key fingerprints.
31. https://github.com/tomrittervg/crypto-usability-study/ *Ritter T. et al.* Crypto Usability Study.
32. https://tools.ietf.org/html/draft-miers-tls-sas-00 — Short Authentication Strings for TLS, Internet Draft.
33. http://philzimmermann.com/docs/PGP_word_list.pdf — PGP word list.
34. *Alexander C. and Goldberg I.* Improved user authentication in off-the-record messaging. Proc. Workshop on Privacy in Electronic Society, ACM, 2007, pp. 41–47.
35. http://peerjs.com — The PeerJS Library.
36. https://rdist.root.org/2010/11/29/final-post-on-javascript-crypto/ *Lawson N.* Final post on Javascript crypto.
37. https://www.nccgroup.trust/us/about-us/newsroom-and-events/blog/2011/august/javascript-cryptography-considered-harmful/ *Ptacek T.* Javascript Cryptography Considered Harmful.
38. https://nadim.computer/2013/05/23/critique-javascript-cryptography.html. *Kobeissi N.* Thoughts on Critiques of JavaScript Cryptography.
39. http://blog.kotowicz.net/2014/07/js-crypto-goto-fail.html?m=1 — JS crypto goto fail?
40. https://www.w3.org/TR/WebCryptoAPI/#security-considerations — Web Cryptography API, W3C Recommendation.

41. *Stark E., Hamburg M., and Boneh D.* Symmetric cryptography in javascript. Proc. Computer Security Applications Conf. (ACSAC'09), IEEE, 2009, pp. 373–381.
42. `http://caniuse.com/#search=webrtc` — Can I Use? WebRTC Peer-to-peer connections.
43. `http://webrtc-security.github.io` — A Study of WebRTC Security.
44. *Hornby T.* Side-Channel Attacks on Everyday Applications: Distinguishing Inputs with FLUSH+RELOAD. `https://www.semanticscholar.org/paper/Side-Channel-Attacks-on-Everyday-Applications-Dist-Hornby/a5ea83ad9abffe6c44b93617728e5f06f73bb9be?tab=citations`. 2016.
45. *Oren Y. et al.* The spy in the sandbox: Practical cache attacks in javascript and their implications. Proc. 22nd SIGSAC Conf. Computer and Communications Security, ACM, 2015, pp. 1406–1418.

ПРИКЛАДНАЯ ТЕОРИЯ ГРАФОВ

УДК 519.6

ИССЛЕДОВАНИЕ ИЗОМОРФИЗМА ГРАФОВ С ПОМОЩЬЮ ЖОРДАНОВЫХ ФОРМ МАТРИЦ СМЕЖНОСТИ

М. И. Володичева*, С. Н. Леора**

*Санкт-Петербургский государственный морской технический университет, г. Санкт-Петербург, Россия,

**Санкт-Петербургский государственный университет, г. Санкт-Петербург, Россия

Для установления отсутствия изоморфизма между орграфами предлагается использовать жорданову форму матриц смежности графов. Задача приведения матрицы к жордановой форме имеет полиномиальную временную сложность, верхняя оценка необходимого числа операций для n-вершинного графа составляет $O(n^4)$. Показано, что жорданова форма матриц смежности орграфов содержит больше информации о структуре графа, чем его спектр, определяемый собственными значениями матрицы смежности и их кратностью. В результате исследования жордановых форм матриц смежности орграфов на отдельных примерах установлено, что изоспектральные матрицы, имеющие одинаковый набор собственных значений, могут приводиться к различным жордановым формам. Это означает, что матрицы смежности не являются подобными, т. е. не являются и перестановочно подобными, что свидетельствует об отсутствии изоморфизма между графами.

Ключевые слова: *граф, орграф, изоморфизм графов, матрица смежности, подобие матриц, жорданова форма.*

DOI 10.17223/20710410/40/7

STUDY OF GRAPH ISOMORPHISM USING JORDAN FORMS OF ADJACENCY MATRICES

M. I. Volodicheva*, S. N. Leora**

*State Marine Technical University of St. Petersburg, Saint Petersburg, Russia,

**Saint Petersburg State University, Saint Petersburg, Russia

E-mail: mvolodicheva@mail.ru, leora2008@mail.ru

It is proposed to use a Jordan form of adjacency matrices to establish the absence of isomorphism between direct graphs. The problem of reduction of a matrix to a Jordan form has polynomial time complexity. The upper estimate of the required number of operations for n-vertex graph is $O(n^4)$. It is shown that the Jordan form of the adjacency matrix contains more information about the structure of the graph than its spectrum determinated by the eigenvalues of the adjacency matrix and their multiplicity. As a result of research on specific examples it was found that the isospectral matrices of the same set of eigenvalues may have different Jordan forms. This means that the adjacency matrices are not similar and therefore are not permutation similar, indicating a lack of isomorphism between direct graphs.

Keywords: *graph, directed graph, graph isomorphism, adjacency matrix, similarity matrices, Jordan form of matrix.*

Введение

Графы широко используются для исследования структурных свойств объектов в различных областях науки и техники: радиоэлектронике, экономике, теоретической физике, энергетике, химии, биологии, социологии, математической лингвистике и др. В частности, графами представляются электронные схемы, искусственные нейронные сети, молекулярные структуры в химии и биологии, сетевые структуры программных систем и баз данных, транспорта, нефтепроводов, газопроводов и др. [1 – 8]. С помощью графов решаются задачи распознавания образов, создаются системы автоматизированного управления, разрабатываются алгоритмы в теории информации и проектирования. Одним из важнейших направлений развития прикладной теории графов является исследование структурного сходства (или структурного различия) графов, которыми моделируются изучаемые объекты.

При построении графовой модели вводятся вершины (множество V), которые описывают компоненты объекта, и рёбра или дуги (множество E), которые описывают отношения между компонентами объекта. Два графа $G_1 = (V_1, E_1)$ и $G_2 = (V_2, E_2)$ называются изоморфными, если существует биекция $f\colon V_1 \to V_2$, сохраняющая смежность, т. е. такая, что для любых двух вершин u и v графа G_1 их образы $f(u)$ и $f(v)$ смежны в графе G_2 тогда и только тогда, когда u и v смежны в графе G_1. В случае ориентированных графов для изоморфных графов требуется дополнительно сохранение ориентации дуг. Отношение изоморфизма графов является отношением эквивалентности, т. е. оно рефлексивно, симметрично и транзитивно. Следовательно, множество всех графов можно разбить на классы изоморфных графов, которые называются абстрактными графами. Проверка изоморфизма графов необходима для ответа на вопрос, являются ли некоторые структуры, которые моделируются заданными графами, принципиально различными или они являются различными представителями одной и той же структуры.

Непосредственная проверка на изоморфизм двух n-вершинных графов заключается в рассмотрении всех $n!$ перестановок вершин и установлении совмещаемости рёбер графов хотя бы при одной перестановке. Однако даже при сравнительно небольших n ($n > 15$) решение проблемы изоморфизма таким образом затруднительно из-за ограниченных возможностей современных вычислительных машин. Задача проверки изоморфизма графов принадлежит к классу задач, для которых пока неизвестно, являются ли они полиномиально разрешимыми или нет, т. е. пока не построен полиномиальный алгоритм проверки изоморфизма любых графов. В 2015–2017 гг. Ласло Бабай разработал новый быстрый алгоритм, решающий задачу изоморфизма графов за квазиполиномиальное время ($\exp(\log(n)^{\mathrm{O}(1)})$, где n — число вершин графа). Уточнение этого алгоритма появилось в январе 2017 г. [9].

Для исследования графов на изоморфизм часто применяются эвристические алгоритмы, которые в общем случае не являются полиномиальными, лишь для отдельных классов графов с определёнными структурными свойствами разработаны полиномиальные алгоритмы. В основе эвристических методов лежит разделение всех вершин графов на классы так, что рассматривается отображение вершин графа в другие вершины того же класса. Для этого используются некоторые свойства вершин графов, инвариантные относительно изоморфизма. В качестве инвариантов, неизменных для всего класса изоморфных графов, рассматриваются числовые, векторные, матричные

инварианты [10 – 12]. Матричные инварианты графа, найденные путём исследования разнообразия специальной матрицы, построенной по матрице смежности и её степеням до n-й включительно, предложены в [11, 12]. В работах [13 – 15] описан полиномиальный алгоритм вычисления полного инварианта графа с помощью интегрального описателя структуры, построенного методом интеграции структурных различий графа (методом ISD). При этом абстрактная структура графа представляется системой числовых кодов для всех вершин графа. Полный инвариант, полученный методом ISD, зависит от используемой системы кодирования, поэтому сравнивать можно только полные инварианты, полученные с помощью одной и той же системы кодирования.

Другой подход исследования графов на изоморфизм изложен в работах [16 – 20]. В 1981 г. Брендан МакКей разработал обладающий большим быстродействием программный пакет nauty [16], который в дальнейшем был усовершенствован и появился под названием traces [17]. Модификациями пакета nauty являются программные пакеты saucy [18], bliss [19], conauto [20]. В основе этих программ лежит построение канонического кода графа, не зависящего от начального порядка нумерации вершин. Два графа изоморфны тогда и только тогда, когда совпадают их канонические коды, которым соответствует каноническая нумерация вершин.

Для решения этой задачи в пакете nauty используется представление множества всех вершин графа в виде разбиения — упорядоченного набора непересекающихся ячеек (подмножеств вершин). Перебору всех перестановок вершин соответствует перебор и уточнение разбиений, в результате которых появляется каноническая нумерация вершин. Уточнение разбиений сопоставляется обходу по дереву разбиений (дереву поиска). Перебор разбиений при нахождении канонической перестановки существенно сокращается при использовании группы автоморфизмов графа. Составной частью nauty является генерация автоморфизмов графа, в результате действия которых появляются орбиты вершин — подмножества вершин, в которые переходят заданные вершины. С помощью nauty можно найти каноническую нумерацию, построить автоморфизмы в неориентированных и ориентированных графах (а также в раскрашенных графах) и провести исследование графов на изоморфизм.

Для больших разреженных графов разработаны программные пакеты saucy и bliss, которые по быстродействию превосходят nauty. Пакет bliss, так же как и nauty, находит каноническую нумерацию вершин, тогда как saucy в настоящее время оптимизирован для нахождения автоморфизмов графов. Пакет conauto осуществляет проверку изоморфности двух графов без построения полной группы автоморфизмов. В основе этого пакета лежит построение последовательности разбиений вершин и некоторых автоморфизмов одного графа, которые затем используются для генерации последовательности разбиений другого графа и обрезания дерева поиска. Пакеты nauty и traces различаются стратегией выбора первой ячейки и направлением обхода дерева поиска (по глубине и по ширине соответственно), что позволяет в случае traces более эффективно использовать операцию обрезания поиска. В настоящее время traces является лидером проверки на изоморфизм неорграфов с большим числом вершин, тогда как nauty целесообразно использовать для массового исследования на изоморфизм небольших графов.

1. Матричный метод проверки изоморфизма графов

Одним из направлений проверки изоморфности графов является матричный метод, в основе которого лежит использование матрицы смежности. Абстрактный граф может иметь различные матрицы смежности в зависимости от нумерации вершин. Пусть

два графа $G_1 = (V_1, E_1)$ и $G_2 = (V_2, E_2)$ различаются только нумерацией вершин, т. е. изоморфны, и имеют матрицы смежности A и B соответственно. В этом случае существует биекция $f\colon V_1 \to V_2$, которая определяет соотношение между элементами матриц A и B:

$$b_{f(i)f(j)} = a_{ij}. \tag{1}$$

Из (1) следует, что матрицы смежности A и B получаются друг из друга одинаковыми перестановками строк и столбцов. Изменение нумерации вершин графа можно описать квадратной $(0,1)$-матрицей перестановки P, которая в каждой строке и каждом столбце имеет ровно одну единицу. Эта матрица удовлетворяет соотношению $PP^T = P^TP = E$, т. е. $P^T = P^{-1}$. Перестановочным преобразованием подобия называется переход от одной матрицы к другой с помощью перестановки строк и такой же перестановки столбцов. Если в исходной нумерации вершин граф G задается матрицей смежности A и тот же граф в новой нумерации вершин задается матрицей B, то матрицы смежности A и B перестановочно подобны:

$$B = P^{-1}AP = P^TAP. \tag{2}$$

Вследствие взаимно однозначного соответствия между графом с конкретной нумерацией вершин и его матрицей смежности можно определить абстрактный граф матрично как класс матриц смежностей графа при всевозможных различных наборах нумераций его вершин, т. е. как класс перестановочно подобных квадратных матриц. Проблема исследования изоморфизма графов сводится к задаче установления перестановочного подобия двух матриц A и B одного порядка, т. е. к проверке условия (2). Если для некоторой матрицы перестановки P условие (2) выполняется, то матрицы A и B перестановочно подобны и являются представителями одного и того же графа. Если ни для одной из $n!$ матриц перестановок P порядка n условие (2) не выполняется, то матрицы A и B не являются перестановочно подобными и задают разные графы.

Частным случаем матричного метода проверки графов на изоморфизм является спектральный подход к решению этой задачи, в основе которого лежит исследование и сравнение спектров графов, т. е. собственных значений и их кратностей, а также собственных векторов матриц смежности исследуемых графов. Одно время полагали, что неизоморфные графы имеют различные спектры. Позднее было обнаружено, что существуют изоспектральные (коспектральные) неизоморфные графы, т. е. спектр графа не является его полным инвариантом. Подробное исследование семейств неизоморфных коспектральных графов содержится в [1].

В данной работе для установления отсутствия изоморфизма орграфов предлагается использовать жордановы формы матриц смежности графов. Решение этой задачи является актуальным в связи с исследованиями искусственных нейронных и других сетей, которые моделируются орграфами.

2. Построение жордановой формы матриц смежности

Жорданова матрица представляет собой клеточно-диагональную матрицу J, которая отличается от диагональной матрицы тем, что выше главной диагонали в параллельном ей ряду могут быть расположены единицы. Её диагональные клетки — жордановы клетки $J_{n_i}(\lambda_i)$ соответствующих порядков с диагональными элементами, равными корням характеристического многочлена матрицы:

$$J = \begin{pmatrix} J_{n_1}(\lambda_1) & 0 & \dots & 0 \\ 0 & J_{n_2}(\lambda_2) & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & J_{n_r}(\lambda_r) \end{pmatrix}, \quad J_{n_i}(\lambda_i) = \begin{pmatrix} \lambda_i & 1 & 0 & \dots & 0 \\ 0 & \lambda_i & 1 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots & \\ 0 & 0 & 0 & \dots & \lambda_i \end{pmatrix},$$

где числа λ_i в некоторых клетках могут быть одинаковыми, при этом порядки n_i некоторых клеток могут совпадать. Число жордановых клеток, соответствующих фиксированному собственному значению λ_i, равно геометрической кратности собственного значения, которая определяется как кратность t_i корня λ_i минимального многочлена и равна размерности соответствующего собственного подпространства. Сумма порядков всех жордановых клеток, соответствующих фиксированному собственному значению λ_i, равна алгебраической кратности k_i корня λ_i характеристического многочлена. Полное число жордановых клеток, среди которых могут быть повторяющиеся, равно максимальному числу линейно независимых собственных векторов матрицы.

Обычно принимается соглашение о том, что различные вещественные собственные значения $\lambda_1, \lambda_2, \dots, \lambda_r$ вещественной матрицы A упорядочиваются в порядке убывания и в жордановой форме сначала располагаются клетки, соответствующие λ_1, затем λ_2 и т. д. Для каждого собственного значения соответствующие жордановы клетки располагают по невозрастанию их порядка. Если матрица A вещественна и обладает только вещественными собственными значениями, то существует вещественная преобразующая матрица S [21]. В случае комплексных собственных значений вещественной матрицы A жорданова форма становится комплексной. Частным случаем жордановой матрицы является диагональная матрица, диагональными элементами которой являются собственные числа матрицы.

Для решения задачи об отсутствии изоморфизма между графами будем использовать теорему Жордана, в основе которой лежат следующие два утверждения.

Утверждение 1. Каждая квадратная матрица подобна некоторой жордановой матрице, т. е. для каждой матрицы $A \in M_n$ существует невырожденная матрица $S \in M_n$, такая, что $S^{-1}AS = A_J$, где A_J — жорданова форма. Жорданова матрица A_J для матрицы A определяется однозначно с точностью до перестановки жордановых клеток на её главной диагонали.

Утверждение 2. Для подобия двух матриц A и B необходимо и достаточно, чтобы соответствующие жордановы матрицы совпадали с точностью до порядка следования клеток.

Если жордановы формы матриц смежности A и B двух графов отличаются не только порядком следования жордановых клеток, то матрицы A и B не являются подобными, а следовательно, они не являются и перестановочно подобными, так как перестановочное подобие является частным видом подобия матриц. В этом случае графы не являются изоморфными. Следовательно, отсутствие изоморфизма между графами можно установить с помощью жордановых форм соответствующих матриц смежности.

Для построения жордановой формы матрицы можно использовать разные методы, один из которых основан на использовании инвариантных множителей характеристической матрицы $A - \lambda E$, с помощью которых затем находятся элементарные делители и жорданова форма матрицы [22, 23]. Простой способ построения жордановой формы для некоторых классов бинарных и взвешенных матриц смежности, основанный на

представлении графа в виде объединения циклов и цепей, приведён в [24, 25]. В данной работе для нахождения жордановой формы матрицы применяется метод, приведённый в [26] и использованный в [27]. Способы приведения матрицы к жордановой форме зависят от вида характеристического многочлена. Пусть характеристический многочлен для матрицы смежности порядка n имеет вид

$$p(\lambda) = (\lambda - \lambda_1)^{k_1}(\lambda - \lambda_2)^{k_2} \dots (\lambda - \lambda_r)^{k_r},$$

где k_i — кратность корня λ_i и $k_1 + \ldots + k_r = n$. Минимальный многочлен имеет вид

$$\mu(\lambda) = (\lambda - \lambda_1)^{t_1}(\lambda - \lambda_2)^{t_2} \dots (\lambda - \lambda_r)^{t_r}.$$

Показатели степеней t_i, входящие в минимальный многочлен, определяются из равенства

$$r_{t_i} = \operatorname{rang}(A - \lambda_i E)^{t_i} = n - k_i. \tag{3}$$

Для определения t_i нужно возводить матрицу $B_i = A - \lambda_i E$ в степени $j = 1, 2, \dots$, вычисляя при этом ранги матриц B_i^j и сравнивая эти ранги с числом $n - k_i$. Тогда наименьший показатель степени, для которого выполняется равенство (3), равен t_i. Величина t_i определяет максимальный размер клетки Жордана, соответствующей собственному значению λ_i. Число s_{t_i} клеток максимального порядка t_i равно

$$s_{t_i} = r_{t_i - 1} - r_{t_i}.$$

Число s_{j_i} жордановых клеток любого порядка j_i, кроме максимального, находится из равенства

$$s_{j_i} = r_{j_{i+1}} - 2r_{j_i} + r_{j_{i-1}}. \tag{4}$$

В частности, число жордановых клеток первого порядка равно $s_1 = r_2 - 2r_1 + n$. Отметим, что некоторые из клеток низшего порядка могут отсутствовать.

В том случае, когда имеется одно собственное значение λ_1 кратности n и характеристический многочлен матрицы имеет вид $p(\lambda) = (\lambda - \lambda_1)^n$, а минимальный многочлен равен $\mu(\lambda) = (\lambda - \lambda_1)^t$, порядок максимальной клетки Жордана равен t. Чтобы найти t, нужно последовательно вычислять степени матрицы $B = A - \lambda_1 E$ до тех пор, пока не получится нулевая матрица. Число клеток меньшей размерности находится по формуле (4).

Найдём верхнюю оценку числа операций, которые необходимо осуществить при установлении отсутствия изоморфизма между графами с помощью жордановой формы матрицы. Вычисление собственных значений матрицы смежности порядка n и определение их кратности требует проведения $\mathrm{O}(n^3)$ операций [22, 23]. Для сортировки собственных значений матрицы необходимо $\mathrm{O}(n \ln(n))$ операций. Для каждого собственного значения λ_i $(1 \leqslant i \leqslant r\ )$ верхняя оценка числа операций, необходимых для вычисления степеней матрицы B_i, составляет $\mathrm{O}(n^4)$. Вычисление ранга матриц B_i^j методом Гаусса требует $\mathrm{O}(n^3)$ операций. Сравнение вычисленного ранга матрицы B_i^j с $(n - k_i)$ требует $\mathrm{O}(n)$ операций. Сопоставление жордановых форм матриц смежности двух графов требует $\mathrm{O}(n)$ операций. Общая верхняя оценка необходимого числа операций составляет $\mathrm{O}(n^4)$.

3. Примеры исследования изоморфизма графов с помощью жордановых форм матриц смежности

Пример 1. Рассмотрим изоспектральные орграфы G_1 и G_2, представленные на рис. 1, собственные значения матриц смежности которых равны нулю и имеют кратность, равную 10, что следует из общей теории структуры спектров орграфов без циклов: все орграфы порядка n, не содержащие циклов, имеют один и тот же спектр (0,0,. . . ,0) [1].

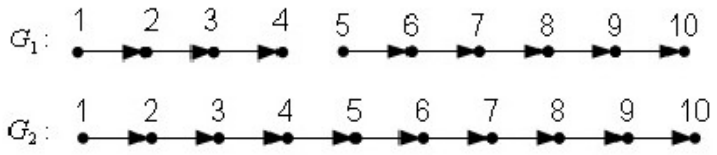

Рис. 1. Изоспектральные неизоморфные графы

Жордановы формы $A1_J$ и $A2_J$ матриц смежности этих графов не подобны: матрица $A1_J$ содержит две клетки Жордана шестого и четвёртого порядков, а матрица $A2_J$ представляет собой одну клетку Жордана десятого порядка:

$$A1_J = \begin{pmatrix} 0&1&0&0&0&0&0&0&0&0\\ 0&0&1&0&0&0&0&0&0&0\\ 0&0&0&1&0&0&0&0&0&0\\ 0&0&0&0&1&0&0&0&0&0\\ 0&0&0&0&0&1&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&1&0&0\\ 0&0&0&0&0&0&0&0&1&0\\ 0&0&0&0&0&0&0&0&0&1\\ 0&0&0&0&0&0&0&0&0&0 \end{pmatrix}, A2_J = \begin{pmatrix} 0&1&0&0&0&0&0&0&0&0\\ 0&0&1&0&0&0&0&0&0&0\\ 0&0&0&1&0&0&0&0&0&0\\ 0&0&0&0&1&0&0&0&0&0\\ 0&0&0&0&0&1&0&0&0&0\\ 0&0&0&0&0&0&1&0&0&0\\ 0&0&0&0&0&0&0&1&0&0\\ 0&0&0&0&0&0&0&0&1&0\\ 0&0&0&0&0&0&0&0&0&1\\ 0&0&0&0&0&0&0&0&0&0 \end{pmatrix}.$$

Вследствие различия жордановых форм $A1_J$ и $A2_J$ матриц смежности изоспектральные графы G_1 и G_2 не изоморфны. Отсюда следует, что жорданова форма матрицы несет в себе больше информации о структуре графа, чем его спектр, т. е. изоспектральные матрицы могут быть не подобны, а значит, и не перестановочно подобны.

Пример 2. Рассмотрим изоспектральные орграфы G_3, G_4 и G_5, приведённые на рис. 2.

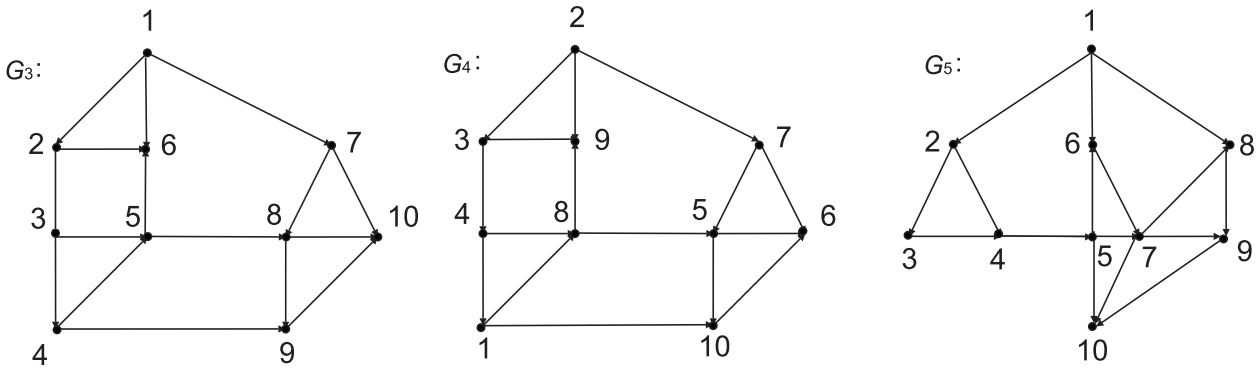

Рис. 2. Изоспектральные орграфы

Жордановы формы $A3_J$ и $A4_J$ матриц смежности для графов G_3 и G_4 совпадают: они соответствуют собственному значению 0 кратности 10 и представляют собой две жордановы клетки порядка 2 и 8 соответственно. Жорданова форма $A5_J$ матрицы смежности для графа G_5 содержит одну клетку десятого порядка, она отличается от жордановой формы матриц смежности для графов G_3 и G_4:

$$\mathrm{A3}_J = \begin{pmatrix} 0&1&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&0\\ 0&0&0&1&0&0&0&0&0&0\\ 0&0&0&0&1&0&0&0&0&0\\ 0&0&0&0&0&1&0&0&0&0\\ 0&0&0&0&0&0&1&0&0&0\\ 0&0&0&0&0&0&0&1&0&0\\ 0&0&0&0&0&0&0&0&1&0\\ 0&0&0&0&0&0&0&0&0&1\\ 0&0&0&0&0&0&0&0&0&0 \end{pmatrix}, \mathrm{A4}_J = \begin{pmatrix} 0&1&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&0\\ 0&0&0&1&0&0&0&0&0&0\\ 0&0&0&0&1&0&0&0&0&0\\ 0&0&0&0&0&1&0&0&0&0\\ 0&0&0&0&0&0&1&0&0&0\\ 0&0&0&0&0&0&0&1&0&0\\ 0&0&0&0&0&0&0&0&1&0\\ 0&0&0&0&0&0&0&0&0&1\\ 0&0&0&0&0&0&0&0&0&0 \end{pmatrix}, \mathrm{A5}_J = \begin{pmatrix} 0&1&0&0&0&0&0&0&0&0\\ 0&0&1&0&0&0&0&0&0&0\\ 0&0&0&1&0&0&0&0&0&0\\ 0&0&0&0&1&0&0&0&0&0\\ 0&0&0&0&0&1&0&0&0&0\\ 0&0&0&0&0&0&1&0&0&0\\ 0&0&0&0&0&0&0&1&0&0\\ 0&0&0&0&0&0&0&0&1&0\\ 0&0&0&0&0&0&0&0&0&1\\ 0&0&0&0&0&0&0&0&0&0 \end{pmatrix}$$

Следовательно, графы G_3 и G_5, а также графы G_4 и G_5 не изоморфны. Эти орграфы не содержат циклов и поэтому все собственные значения их матриц смежности равны нулю. Для графа G_6, отличающегося от G_5 изменением направления дуги $(5, 7)$, в результате чего образуется цикл $(5, 6, 7, 5)$, жорданова форма матрицы смежности имеет следующий вид:

$$A6_J = \begin{pmatrix} -0{,}5 - 0{,}866i & 0 & 0&0&0&0&0&0&0&0\\ 0 & -0{,}5 + 0{,}866i & 0&0&0&0&0&0&0&0\\ 0 & 0 & 0&1&0&0&0&0&0&0\\ 0 & 0 & 0&0&1&0&0&0&0&0\\ 0 & 0 & 0&0&0&1&0&0&0&0\\ 0 & 0 & 0&0&0&0&1&0&0&0\\ 0 & 0 & 0&0&0&0&0&1&0&0\\ 0 & 0 & 0&0&0&0&0&0&1&0\\ 0 & 0 & 0&0&0&0&0&0&0&0\\ 0 & 0 & 0&0&0&0&0&0&0&1 \end{pmatrix}.$$

Эта матрица представляет собой прямую сумму трёх жордановых клеток первого порядка, соответствующих собственным значениям $1, -0{,}5 + 0{,}8661, -0{,}5 - 0{,}8661$, и одной жордановой клетки седьмого порядка, соответствующей собственному значению 0 кратности 7. Она отличается от жордановых форм матриц смежности графов G_3 и G_4. Следовательно, графы G_3 и G_6, а также графы G_4 и G_6 не изоморфны.

Полученные результаты подтверждены с помощью пакета Mathematica 9.

Пример 3. Рассмотрим неорграфы $\widetilde{G}_3$, $\widetilde{G}_4$, $\widetilde{G}_5$, соответствующие орграфам G_3, G_4, G_5. Матрицы смежности этих графов симметричны и приводятся к диагональной форме. Собственные числа матриц смежности неорграфов $\widetilde{G}_3$, $\widetilde{G}_4$ совпадают и имеют следующие значения:

$$3{,}257, 2{,}217, 1{,}721, 0{,}219, 0, -0{,}359, -1{,}037, -1{,}750, -1{,}920, -2{,}348.$$

Собственные значения матрицы смежности графа $\widetilde{G}_5$ равны

$$3{,}213, 2{,}138, 1{,}365, 0{,}927, -0{,}330, -0{,}592, -1{,}103, -1{,}574, -1{,}876, -2{,}169.$$

Они отличаются от собственных значений матриц смежности неорграфов $\widetilde{G}_3$, $\widetilde{G}_4$. Следовательно, графы $\widetilde{G}_3$ и $\widetilde{G}_4$ коспектральны, тогда как графы $\widetilde{G}_3$ и $\widetilde{G}_5$, а также $\widetilde{G}_4$

и $\widetilde{G}_5$ имеют различные спектры. Отсюда следует, что графы $\widetilde{G}_3$ и $\widetilde{G}_5$ ($\widetilde{G}_4$ и $\widetilde{G}_5$) не изоморфны. Такой же результат получается с помощью пакета Mathematica 9.

В том случае, когда матрицы смежности двух орграфов подобны, с помощью их жордановых форм нельзя дать однозначного ответа на вопрос об изоморфизме графов, так как матрицы смежности могут быть подобны, но не перестановочно подобны.

Пример 4. Рассмотрим орграфы G_7, G_8, G_9, представленные на рис. 3.

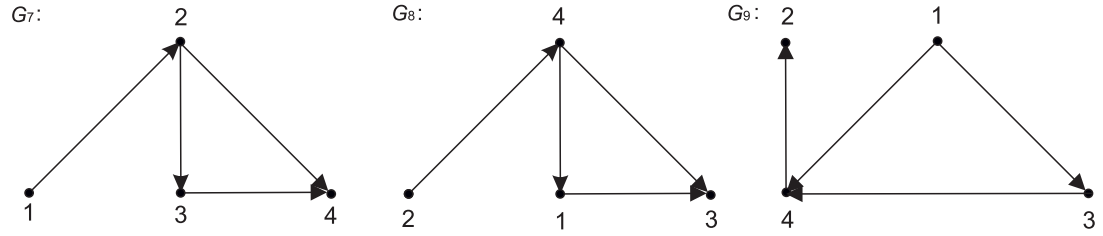

Рис. 3. Изоспектральные орграфы с одинаковыми жордановыми формами матриц смежности

Собственные значения матриц смежности орграфов G_7, G_8, G_9 равны нулю и имеют кратность $r = 4$. Жордановы формы этих матриц одинаковы и представляют собой жордановы клетки четвёртого порядка:

$$A7_J = A8_J = A9_J = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix}.$$

Следовательно, матрицы смежности графов G_7, G_8, G_9 подобны. Однако отсюда не следует, что эти графы изоморфны. Вычислим степенные последовательности вершин графов G_7, G_8, G_9. Пусть $\deg^- v$ — полустепень исхода вершины v, $\deg^+ v$ — полустепень захода вершины v. Тогда для графа G_7 полустепени исхода и захода вершин равны

$$\deg^- 1 = 1, \deg^+ 1 = 0, \deg^- 2 = 2, \deg^+ 2 = 1, \deg^- 3 = 1, \deg^+ 3 = 1, \deg^- 4 = 1, \deg^+ 4 = 2,$$

т. е. степенная последовательность вершин имеет вид $(2,1), (1,1), (0,2), (1,0)$. Для графа G_9 полустепени исхода и захода вершин равны

$$\deg^- 1 = 2, \deg^+ 1 = 0, \deg^- 2 = 0, \deg^+ 2 = 1, \deg^- 3 = 1, \deg^+ 3 = 1, \deg^- 4 = 1, \deg^+ 4 = 2,$$

т. е. степенная последовательность вершин — $(1,2), (1,1), (2,0), (0,1)$. Степенные последовательности различны, следовательно, графы G_7 и G_9 не изоморфны. Для графа G_8 степенная последовательность вершин имеет вид $(2,1), (1,1), (0,2), (0,1)$, которая совпадает с последовательностью для графа G_7. В данном случае существует биективное отображение между вершинами графов G_7 и G_8, сохраняющее смежность: $1 \to 2, 2 \to 4, 3 \to 1, 4 \to 3$, т. е. графы G_7 и G_8 изоморфны.

Для ответа на вопрос об изоморфизме графов, матрицы смежности которых имеют одинаковые жордановы формы, можно применить методы, основанные на построении различных инвариантов графов [11–20, 28].

В последнее время появились работы [29, 30], в которых жордановы формы и жорданов базис матриц смежностей графов применяются при обработке графических сигналов (GSP) для изучения структурных и функциональных свойств различных сетей.

Пространство графических сигналов в [29, 30] представлено в виде прямой суммы жордановых подпространств, натянутых на жордановы цепочки векторов матрицы смежности. Жордановы подпространства являются спектральными компонентами графического преобразования Фурье, которое инвариантно относительно переименования вершин графа. В [29] введены жордановы классы эквивалентности графов, для которых разложение пространства графических сигналов на жордановы подпространства имеет один и тот же вид. При этом изоморфные графы имеют изоморфные жордановы пространства. В [29] отмечено, что из жордановой эквивалентности графов не следует их изоморфизм. Операция проектирования графических сигналов на жордановы подпространства, описанная в [29], позволяет в значительной степени упростить обработку числовых данных сетей, что показано на примере расчёта числовых характеристик сети автомобильных дорог в Нью-Йорке [30].

При исследовании графов на изоморфизм с помощью жордановых форм матриц смежности могут возникать трудности в случае больших значений коэффициентов перекоса матрицы смежности, когда задача определения собственных значений становится неустойчивой [27]. Эти же трудности возникают при использовании спектрального метода.

Авторы выражают благодарность рецензенту за полезные замечания.

ЛИТЕРАТУРА

1. *Цветкович Д., Дуб М., Захс Х.* Спектры графов. Теория и применение. Киев: Наукова думка, 1984. 378 с.
2. *Касьянов В. Н., Евстигнеев В. А.* Графы в программировании: обработка, визуализация и применение. СПб.: БХВ-Петербург, 2003. 1104 с.
3. *Ильяшенко М. Б., Голдобин А. А.* Решение задачи поиска изоморфизма графов для проектирования специализированных вычислителей // Радиоэлектроника, информатика, управление. 2012. № 1. С. 31–36.
4. *Марлей В. Е., Плотников С. Н.* Алгоритм распознавания изоморфного вложения алгоритмических сетей // Вестник ВГУ ИТ. 2014. № 3. С. 72–75.
5. *Newman M. E. J.* Networks: an Introduction. Oxford: Oxford University Press, 2010. 784 p.
6. *Sandryhaila A. and Moura J. M. F.* Discrete signal processing on graphs // IEEE Trans. Signal Process. 2013. V. 61. No. 7. P. 1644–1656.
7. *Sandryhaila A. and Moura J. M. F.* Discrete signal processing on graphs: frequency analysis // IEEE Trans. Signal Process. 2014. V. 62. No. 12. P. 3042–3054.
8. *Teke O. and Vaidyanathan P. P.* Uncertainty principles and sparse eigenvectors of graphs // IEEE Trans. Signal Process. 2017. V. 65. No. 20. P. 5406–5420.
9. *Babai L.* Graph isomorphism in quasipolynomial time. arXiv:1512.03547 (v1:11 Dec 2015, v2:19 January 2016). `http://people.cs.uchicago.edu/~laci/update.html`
10. *Зыков А. А.* Основы теории графов. М.: Вузовская книга, 2004. 664 с.
11. *Хитров Г. М.* О разнообразии графа и применении этого понятия к проблеме изоморфизма графов // Вестн. С.-Петерб. ун-та. 2006. Сер. 10. № 2. С. 91–100.
12. *Погожев С. В., Хитров Г. М.* О проблеме изоморфизма графов и об одном матричном алгоритме ее решения // Вестн. С.-Петерб. ун-та. 2008. Сер. 10. № 1. С. 1–5.
13. *Погребной А. В.* Полный инвариант графа и алгоритм его вычисления // Известия Томского политехнического университета. 2014. Т. 325. № 5. С. 110–122.
14. *Погребной А. В.* Метод определения сходства структур графов на основе выделения частичного изоморфизма в задачах геоинформатики // Известия Томского политехнического университета. 2015. Т. 326. № 11. С. 56–66.

15. *Погребной В. К., Погребной А. В.* Полиномиальный алгоритм вычисления полного инварианта графа на основе интегрального описателя структуры // Известия Томского политехнического университета. 2013. Т. 323. № 5. С. 152–159.
16. *McKay B. D.* Practical graph isomorphism // Congressus Numerantium. 1981. V. 30. P. 45–87.
17. *McKay B. D. and Piperno A.* Practical graph isomorphism, II // J. Symbolic Computation. 2014. V. 60. P. 94–112.
18. *Darga P. T., Sakallah K. A., and Markov I. L.* Faster symmetry discovery using sparsity of symmetries // Proc. 45th Design Automation Conf. 2004. P. 149–154.
19. *Junttila T. and Kaski P.* Engineering an efficient canonical labeling tool for large and sparse graphs // Proc. 9th Workshop on Algorithm Engineering and Experiments and the 4th Workshop on Analytic Algorithms and Combinatorics. SIAM, 2007. P. 135–149.
20. *López-Presa J. L. and Fernández A. A.* Fast algorithm for graph isomorphism testing // Proc. 8th Intern. Symp. Experimental Algorithms. 2009. P. 221–232.
21. *Хорн Р., Джонсон Ч.* Матричный анализ. М.: Мир, 1989. 655 с.
22. *Фаддеев Д. К., Фаддеева В. Н.* Вычислительные методы линейной алгебры. СПб.: Лань, 2009. 736 с.
23. *Уилкинсон Дж. Ч.* Алгебраическая проблема собственных значений. М.: Наука, 1970. 564 с.
24. *Cardon D. A. and Tuckfield B.* The Jordan canonical form for a class of zero–one matrices // Linear Algebra and its Appl. 2011. V. 435. No. 11. P. 2942–2954.
25. *Nina H., Soto R. L., and Cardoso D. M.* The Jordan canonical form for a class of weighted directed graphs // Linear Algebra and its Appl. 2013. V. 438. No. 1. P. 261–268.
26. *Беклемишев Д. В.* Дополнительные главы линейной алгебры. М.: Наука, 1983. 336 с.
27. *Володичева М. И., Григорьев-Голубев В. В., Леора С. Н.* Собственные векторы, жордановы формы, функции матриц. MatLab, Mathematica, Maple. СПб.: Изд. центр СПбГМТУ, 2009. 175 с.
28. *Пролубников А. В.* Прямой алгоритм проверки изоморфизма графов // Компьютерная оптика. 2007. Т. 31. № 3. С. 86–92.
29. *Deri J. A. and Moura J. M. F.* Graph Equivalence Classes for Spectral Projector-Based Graph Fourier Transforms. arXiv:1701.02864 v1 [cs.SI] 11 Jan 2017.
30. *Deri J. A. and Moura J. M. F.* Spectral projector-based graph Fourier transforms // IEEE J. Selected Topics in Signal Processing. 2017. V. 11. No. 6. P. 785–795.

REFERENCES

1. *Tsvetkovich D., Dub M., and Zakhs H.* Spektry grafov. Teoriya i primeneniye [Spectra of graphs. Theory and application]. Kiev, Naukova Dumka, 1984. 378 p.(in Russian)
2. *Kas'yanov V. N. and Yevstigneyev V. A.* Grafy v programmirovanii: obrabotka, vizualizatsiya i primeneniye [Graphs in programming: processing, visualization and application]. SPb., BKHV-Peterburg, 2003. 1104 p. (in Russian)
3. *Il'yashenko M. B. and Goldobin A. A.* Resheniye zadachi poiska izomorfizma grafov dlya proyektirovaniya spetsializirovannykh vychisliteley [The solution of problem of search of graph isomorphism for designing special computers]. Radioelektronika, Informatika, Upravleniye, 2012, no. 1, pp. 31–36. (in Russian)
4. *Marley V. Ye. and Plotnikov S. N.* Algoritm raspoznavaniya izomorfnogo vlozheniya algoritmicheskikh setey [The algorithm for recognizing isomorphic investment of algorithmic networks]. Vestnik VGU IT, 2014, no. 3, pp. 72–75. (in Russian)
5. *Newman M. E. J.* Networks: an Introduction. Oxford, Oxford University Press, 2010. 784 p.

6. *Sandryhaila A. and Moura J. M. F.* Discrete signal processing on graphs. IEEE Trans. Signal Process., 2013, vol. 61, no. 7, pp. 1644–1656.
7. *Sandryhaila A. and Moura J. M. F.* Discrete signal processing on graphs: frequency analysis. IEEE Trans. Signal Process., 2014, vol. 62, no. 12, pp. 3042–3054.
8. *Teke O. and Vaidyanathan P. P.* Uncertainty principles and sparse eigenvectors of graphs. IEEE Trans. Signal Process., 2017, vol. 65, no. 20, pp. 5406–5420.
9. *Babai L.* Graph isomorphism in quasipolynomial time. arXiv:1512.03547 (v1:11 Dec 2015, v2:19 January 2016. `http://people.cs.uchicago.edu/~laci/update.html`
10. *Zykov A. A.* Osnovy teorii grafov [Fundamentals of Graph Theory]. Moscow, Vuzovskaya kniga, 2004. 664 p. (in Russian)
11. *Khitrov G. M.* O raznoobrazii grafa i primenenii etogo ponyatiya k probleme izomorfizma grafov [On the graph diversity and the application of this concept to the problem of graph isomorphism]. Vestn. S.-Peterb. un-ta, 2006, ser. 10, no. 2, pp. 91–100. (in Russian)
12. *Pogozhev S. V. and Khitrov G. M.* O probleme izomorfizma grafov i ob odnom matrichnom algoritme yeye resheniya [On the problem of graph isomorphism and one matrix algorithm to solve it]. Vestn. S.-Peterb. un-ta, 2006, ser. 10, no. 1, pp. 1–5. (in Russian)
13. *Pogrebnoy A. V.* Polnyy invariant grafa i algoritm yego vychisleniya [Complete invariant of the graph and the algorithm of its calculation]. Izvestiya Tomskogo Politekhnicheskogo Universiteta, 2014, vol. 325, no. 5, pp. 110–122. (in Russian)
14. *Pogrebnoy A. V.* Metod opredeleniya skhodstva struktur grafov na osnove vydeleniya chastichnogo izomorfizma v zadachakh geoinformatiki [Method for determining the similarity of structures of graphs based on the allocation of partial isomorphism in geoinformatics tasks]. Izvestiya Tomskogo Politekhnicheskogo Universiteta, 2015, vol. 326, no. 11, pp. 56–66. (in Russian)
15. *Pogrebnoy V. K. and Pogrebnoy A. V.* Polinomial'nyy algoritm vychisleniya polnogo invarianta grafa na osnove integral'nogo opisatelya struktury [A polynomial algorithm for computing the complete invariant of the graph based on the integral handle structure]. Izvestiya Tomskogo Politekhnicheskogo Universiteta, 2013, vol. 323, no. 5, pp. 152–159. (in Russian)
16. *McKay B. D.* Practical graph isomorphism. Congressus Numerantium, 1981, vol. 30, pp. 45–87.
17. *McKay B. D. and Piperno A.* Practical graph isomorphism, II. J. Symbolic Computation, 2014, vol. 60, pp. 94–112.
18. *Darga P. T., Sakallah K. A., and Markov I. L.* Faster symmetry discovery using sparsity of symmetries. Proc. 45th Design Automation Conf., 2004, pp. 149–154.
19. *Junttila T. and Kaski P.* Engineering an efficient canonical labeling tool for large and sparse graphs. Proc. 9th Workshop on Algorithm Engineering and Experiments and the 4th Workshop on Analytic Algorithms and Combinatorics, SIAM, 2007, pp. 135–149.
20. *López-Presa J. L. and Fernández A. A.* Fast algorithm for graph isomorphism testing. Proc. 8th Intern. Symp. Experimental Algorithms, 2009, pp. 221–232.
21. *Khorn R. and Dzhonson Ch.* Matrichnyy analiz [Matrix Analysis]. Moscow, Mir Publ., 1989. 655 p.(in Russian)
22. *Faddeyev D. K. and Faddeyeva V. N.* Vychislitel'nyye metody lineynoy algebry [Computational Methods of Linear Algebra]. SPb., Lan' Publ., 2009. 736 p. (in Russian)
23. *Uilkinson Dzh. H.* Algebraicheskaya problema sobstvennykh znacheniy [The Algebraic Problem of Eigenvalues]. Moscow, Nauka Publ., 1970. 564 p. (in Russian)
24. *Cardon D. A. and Tuckfield B.* The Jordan canonical form for a class of zero–one matrices. Linear Algebra and its Appl., 2011, vol. 435, no. 11, pp. 2942–2954.

25. *Nina H., Soto R. L., and Cardoso D. M.* The Jordan canonical form for a class of weighted directed graphs. Linear Algebra and its Appl., 2013, vol. 438, no. 1, pp. 261–268.
26. *Beklemishev D. V.* Dopolnitel'nyye glavy lineynoy algebry [Additional Chapters of Linear Algebra]. Moscow, Nauka Publ., 1983. 336 p. (in Russian)
27. *Volodicheva M. I., Grigor'yev-Golubev V. V., and Leora S. N.* Sobstvennyye vektory, zhordanovy formy, funktsii matrits. MatLab, Mathematica, Maple [Eigenvectors, Jordan Forms and Matrix Functions. MatLab, Mathematica, Maple]. SPb., SPbGMTU Publ., 2009. 175 p. (in Russian)
28. *Prolubnikov A. V.* Pryamoy algoritm proverki izomorfizma grafov [Direct algorithm of graph isomorphism testing]. Komp'yuternaya Optika, 2007, vol. 31, no. 3, pp. 86–92. (in Russian)
29. *Deri J. A. and Moura J. M. F.* Graph Equivalence Classes for Spectral Projector-Based Graph Fourier Transforms. arXiv:1701.02864 v1 [cs.SI] 11 Jan 2017.
30. *Deri J. A. and Moura J. M. F.* Spectral Projector-Based Graph Fourier Transforms. IEEE J. Selected Topics in Signal Processing, 2017, vol. 11, no. 6, pp. 785–795.

МАТЕМАТИЧЕСКИЕ ОСНОВЫ ИНФОРМАТИКИ И ПРОГРАММИРОВАНИЯ

УДК 510.52

РЕЛЯТИВИЗОВАННЫЕ ГЕНЕРИЧЕСКИЕ КЛАССЫ P И NP[1]

А. Н. Рыбалов

Омский государственный технический университет, г. Омск, Россия

Бейкер, Гилл и Соловей в 1975 г. построили такие два оракула A и B, что $\mathrm{P}^A = \mathrm{NP}^A$, но $\mathrm{P}^B \neq \mathrm{NP}^B$. Тем самым они показали, что неравенство $\mathrm{P} \neq \mathrm{NP}$ не может быть доказано с использованием метода диагонализации. В рамках генерического подхода алгоритмическая проблема рассматривается не на всём множестве входов, а на некотором подмножестве «почти всех» входов. Такие входы образуют так называемое генерическое множество. Понятие «почти все» формализуется введением естественной меры на множестве входных данных. В работе определяются генерические аналоги genP и genNP классов вычислительной сложности P и NP, а также их релятивизованные версии. Доказывается генерический аналог теоремы Бейкера — Гилла — Соловея: существуют такие оракулы A и B, что $\mathrm{genP}^A = \mathrm{genNP}^A$, но $\mathrm{genP}^B \neq \mathrm{genNP}^B$. Таким образом, для решения генерического аналога проблемы совпадения классов P и NP метод диагонализации также неприменим.

Ключевые слова: *генерическая сложность, проблема $P = NP$, оракулы.*

DOI 10.17223/20710410/40/8

RELATIVIZED GENERIC CLASSES P AND NP

A. N. Rybalov

Omsk State Technical University, Omsk, Russia

E-mail: alexander.rybalov@gmail.com

Classical theorem of Baker, Gill and Solovay states that there exist two oracles A and B such that $\mathrm{P}^A = \mathrm{NP}^A$, but $\mathrm{P}^B \neq \mathrm{NP}^B$. This result indicates that the classical tools of computability theory (such as diagonalization) are inapplicable to prove the inequality $\mathrm{P} \neq \mathrm{NP}$. 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. Many classical undecidable or hard algorithmic problems become feasible in the generic case. In this paper we introduce generic analogs genP and genNP of the classical computational complexity classes P and NP. We prove a generic analog of the Baker — Gill — Solovay theorem: there exist two oracles A and B such that $\mathrm{genP}^A = \mathrm{genNP}^A$, but $\mathrm{genP}^B \neq \mathrm{genNP}^B$. Therefore the diagonalization arguments cannot be applied to prove the inequality $\mathrm{P} \neq \mathrm{NP}$ in the generic case too.

[1]Работа поддержана грантом РНФ № 17-11-01117.

Keywords: *generic complexity, P vs NP problem, oracles.*

Введение

Наиболее сильные результаты в теории вычислительной сложности, такие, как теоремы о временной и пространственной иерархии классов сложности, получены с применением метода так называемой диагонализации. Этот метод, пришедший из классической теории алгоритмов и берущий свое начало ещё в работах Геделя и Тьюринга, активно использовался в 1960–70-х гг. в попытках решения многих центральных проблем теории сложности вычислений, прежде всего, проблемы равенства классов P и NP. Однако в 1975 г. Бейкер, Гилл и Соловей в [1] доказали теорему, которая говорит о том, что с помощью метода диагонализации нельзя доказать неравенство $\mathrm{P} \neq \mathrm{NP}$. Оказывается, что любое доказательство неравенства каких-либо классов проблем $C_1 \neq C_2$, полученное с помощью диагонализации, релятивизуемо. Это означает, что та же схема проходит и для доказательства неравенства $C_1^A \neq C_2^A$, где A — любой счётный оракул. Оракул A является просто некоторым подмножеством входов. Релятивизованные классы определяются при помощи машин, которые имеют команды обращения к оракулу вида $x \in A?$. Такие машины в процессе вычислений могут делать запросы к оракулу за одну единицу времени и проверять принадлежность различных элементов множеству A. При надлежащем выборе оракула вычислительная мощность релятивизованных машин увеличивается: они могут быстро решать труднорешаемые и даже неразрешимые проблемы. Формальные определения релятивизованных машин и классов можно найти в [2, 3]. Бейкер, Гилл и Соловей построили такие два оракула A и B, что $\mathrm{P}^A = \mathrm{NP}^A$, но $\mathrm{P}^B \neq \mathrm{NP}^B$. Тем самым они показали, что неравенство $\mathrm{P} \neq \mathrm{NP}$ не может быть доказано с использованием метода диагонализации.

В рамках генерического подхода [4] алгоритмическая проблема рассматривается не на всём множестве входов, а на некотором подмножестве «почти всех» входов. Такие входы образуют так называемое генерическое множество. Понятие «почти все» формализуется введением естественной меры на множестве входных данных. С точки зрения практики алгоритмы, решающие быстро проблему на генерическом множестве, так же хороши, как и быстрые алгоритмы для всех входов. Классическим примером такого алгоритма является симплекс-метод — он за полиномиальное время решает задачу линейного программирования для большинства входных данных, но имеет экспоненциальную сложность в худшем случае. Более того, может так оказаться, что проблема трудноразрешима или вообще неразрешима в классическом смысле, но легкоразрешима на генерическом множестве.

В данной работе определяются генерические аналоги genP и genNP классов вычислительной сложности P и NP, а также их релятивизованные версии. Доказывается генерический аналог теоремы Бейкера — Гилла — Соловея: существуют такие оракулы A и B, что $\mathrm{genP}^A = \mathrm{genNP}^A$, но $\mathrm{genP}^B \neq \mathrm{genNP}^B$. Таким образом, для решения генерического аналога проблемы совпадения классов P и NP метод диагонализации также неприменим.

1. Генерические классы P и NP

Пусть I — некоторое множество. Функция $\mathrm{size} : I \to \mathbb{N}$ называется *функцией размера*, если для любого $n \in \mathbb{N}$ множество $I_n = \{x \in I : \mathrm{size}(x) = n\}$ конечно. Например, если $I = \Sigma^*$ — множество слов над конечным алфавитом Σ, то функцией размера будет функция, определённая для любого слова w как его длина $|w|$. Для множества натуральных чисел $\mathbb{N}$ функция размера сопоставляет любому натуральному числу длину

его двоичной записи. Как обычно делается в теории вычислимости, под алгоритмическими проблемами будем понимать проблемы распознавания подмножеств из некоторого множества входов с определённой на нем функцией длины. Для 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. Заметим, что $\rho_n(S)$ — это вероятность попасть в S при случайной и равновероятной генерации входов из I_n. *Асимптотической плотностью* S назовем предел (если он существует)

$$\rho(S) = \lim_{n\to\infty} \rho_n(S).$$

Множество S называется *генерическим*, если $\rho(S) = 1$, и *пренебрежимым*, если $\rho(S) = 0$.

В дальнейшем будем рассматривать в качестве множества входов $\{0,1\}^*$ — множество всех слов над двоичным алфавитом $\{0,1\}$. Тем не менее все последующие определения и результаты без особого труда могут быть перенесены на любое конструктивное множество входов.

Определим генерические аналоги классических классов вычислительной сложности P и NP. Множество $S \subseteq \{0,1\}^*$ принадлежит *классу* genP, если существует разрешимое за полиномиальное время генерическое множество $G \subseteq \{0,1\}^*$, такое, что $G \cap S$ разрешимо за полиномиальное время. Проблема из класса genP — это проблема, которая решается эффективно (за полиномиальное время), но не для всех входов, а для почти всех входов (входов из генерического множества G). Множество $S \subseteq \{0,1\}^*$ принадлежит *классу* genNP, если существует разрешимое за полиномиальное время генерическое множество $G \subseteq \{0,1\}^*$, такое, что $S \cap G \in \mathrm{NP}$. Напомним определение класса NP. Проблема $S \in \mathrm{NP}$, если существует такое множество $S' \in \mathrm{P}$ и такой полином $p(n)$, что $x \in S \Leftrightarrow \exists y \in \{0,1\}^*\,(|y| < p(|x|)$ и $(x,y) \in S')$. Здесь строка y называется подсказкой (или сертификатом, или решением), а множество S' — проверятелем. Проблема из класса genNP — это проблема, решение которой проверяется эффективно (за полиномиальное время) для почти всех входов (входов из генерического множества G).

Теперь определим релятивизованные версии этих классов. Пусть A есть некоторое подмножество I. *Машина с оракулом* A в своей программе может содержать команды обращения к оракулу вида «if $x \in A$ goto m», причём выполнение этой команды происходит за один шаг работы вычислительного устройства. Более формально машины с оракулом определены, например, в [3]. Множество $S \subseteq \{0,1\}^*$ принадлежит *классу* genP^A, если существует разрешимое за полиномиальное время на машинах с оракулом A генерическое множество $G \subseteq \{0,1\}^*$, такое, что $G \cap S$ разрешимо за полиномиальное время на машине с оракулом A. Множество $S \subseteq \{0,1\}^*$ принадлежит *классу* genNP^A, если существует разрешимое за полиномиальное время на машинах с оракулом A генерическое множество $G \subseteq \{0,1\}^*$, такое, что $S \cap G \in \mathrm{NP}^A$. Релятивизованный класс NP^A определяется аналогично обычному классу NP: проблема $S \in \mathrm{NP}^A$, если существует такое разрешимое за полиномиальное время на машинах с оракулом A множество S' и такой полином $p(n)$, что $x \in S \Leftrightarrow \exists y \in \{0,1\}^*\,(|y| < p(|x|)$ и $(x,y) \in S')$.

2. Генерическая теорема Бейкера — Гилла — Соловея

Теорема 1. Существует такой оракул A, что $\mathrm{genP}^A = \mathrm{genNP}^A$.

Доказательство. В качестве оракула A можно взять оракул из теоремы Бейкера — Гилла — Соловея, такой, что $\mathrm{P}^A = \mathrm{NP}^A$. Действительно, пусть $S \in \mathrm{genNP}^A$. Тогда существует A-разрешимое за полиномиальное время генерическое множество G, такое, что $S \cap G \in \mathrm{NP}^A$. Но $\mathrm{NP}^A = \mathrm{P}^A$, поэтому $S \cap G \in \mathrm{P}^A$. Это означает, что $S \in \mathrm{genP}^A$. Таким образом, $\mathrm{genP}^A = \mathrm{genNP}^A$. ■

Теорема 2. Существует такой оракул B, что $\mathrm{genP}^B \neq \mathrm{genNP}^B$.

Доказательство. Идея построения оракула B такова: построить связанное с ним множество из NP^B так, чтобы любая полиномиальная машина с оракулом B не распознавала это множество на непренебрежимом множестве. Этот множество следующее:

$$\Omega_B = \{\omega \in \{0,1\}^* : \text{ в } B \text{ существует строка длины } 2|\omega|\}.$$

Заметим, что $\Omega_B \in \mathrm{NP}^B$ для любого оракула B. Действительно, для входа ω сертификатом является строка x длины $2|\omega|$, такая, что $x \in B$. Если $\omega \in \Omega_B$, то такой сертификат найдётся, иначе нет. Процесс проверки сертификата выполняется за полиномиальное время на машине с оракулом B.

Теперь нужно построить B так, чтобы $\Omega_B \notin \mathrm{genP}^B$. Пусть есть эффективная нумерация полиномиальных машин с оракулами $P_1, P_2, P_3, \ldots$ Образуем из них следующую последовательность:

$$\{M_i, i = 1, 2, 3, \ldots\} = \{P_1, P_1, P_2, P_1, P_2, P_3, P_1, P_2, P_3, P_4, P_1, P_2, \ldots\}.$$

В ней каждый раз после того, как выписаны машины $P_1, \ldots, P_k$, выписываются машины $P_1, \ldots, P_{k+1}$. Таким образом, каждая полиномиальная машина P_i выписывается бесконечно много раз.

Построение оракула B проходит по шагам, стартуя с множества $B = \{0,1\}^*$, на i-м шаге вычеркиваем строки из B так, чтобы машина M_i не смогла распознать Ω_B на большинстве входов некоторого фиксированного размера. Опишем сначала шаг 1.

Ш а г 1. Выберем число k таким, что $p(k) < 2^k$, где $p()$ — это полином, ограничивающий время работы машины M_1. Запускаем M_1 на каждой строке размера k (их 2^k штук). Во время своей работы на каждом входе, как только она делает запрос к оракулу вида $a \in B?$, вычёркиваем эту строку a из B (вынуждая оракул всегда отвечать НЕТ). В процессе работы M_1 на всех 2^k входах мы вычеркнем некоторые $a_1, \ldots, a_m$, где $m < p(k)2^k < 2^{2k}$. Далее смотрим: если M_1 выдаёт на 2^k входах длины k выход 1 не менее чем в половине случаев (т. е. $\geqslant 2^{k-1}$ раз), то вычеркиваем все строки длины $2k$ из B, тем самым гарантируя, что $\omega \notin \Omega_B$ для любой строки ω длины k. Это означает, что машина M_1 ошибается на входах длины k не менее чем в половине случаев. Если же M_1 выдает в большинстве случаев 0, мы выбираем некоторую строку длины $2k$ из оставшихся в B и помечаем её (такая строка найдётся, так как мы вычеркнули $< 2^{2k}$ строк). Это также гарантирует, что $\omega \in \Omega_B$ для любой строки ω длины k. Таким образом, опять машина M_1 ошибается на входах длины k не менее чем в половине случаев.

Ш а г i. Итак, машины $M_1, \ldots, M_{i-1}$ не распознают Ω_B, ошибаясь на входах с длинами $k_1, \ldots, k_{i-1}$ не менее чем в половине случаев. Берем $k > k_{i-1}$, такое, что $p(k) < 2^k$, где $p()$ — это полином, ограничивающий время работы машины M_i. Кроме

того, k должно быть больше размеров всех строк, которые были помечены или вычеркнуты на предыдущих шагах. Опять запускаем машину M_i на каждом входе длины k и вычеркиваем строчки-запросы (делая ответы на запросы к оракулу НЕТ), но только те, которые не помечены. Если строка в запросе $a \in B$? помечена на предыдущих шагах, она уже точно будет в окончательном оракуле. Это гарантирует, что работа машины M_i на любом входе ω длины k не будет меняться в дальнейшем (то есть ответы на её запросы $a \in B$ будут теми же и на последующих шагах). Как и на шаге 1, если M_i выдаёт 1 на большинстве входов длины k, вычеркиваем все строки длины $2k$ из оракула B. Если она для большинства выдаёт 0, то помечаем одну из оставшихся строк длины $2k$ и переходим к следующему шагу. Опять M_i не может распознать Ω_B не менее чем для половины входов длины k.

Предельное множество в этом процессе и есть искомый оракул B. Действительно, $\Omega_B \in \mathrm{NP}^B$, а тем более $\Omega_B \in \mathrm{genNP}^B$. Чтобы убедиться, что $\Omega_B \notin \mathrm{genP}^B$, заметим, что для любой полиномиальной машины M с оракулом B множество входов, на которых M даёт неправильный ответ, имеет вид

$$E(M) = \bigcup_{i=1}^{\infty} A_i,$$

где

$$A_i = \{\omega \in \{0,1\}^* : |\omega| = m_i\}, \quad m_i > m_{i-1},$$

причём $|A_i| \geqslant 2^{m_i}$ для любого i. Если теперь рассмотреть последовательность

$$\rho_n(E(M)) = \frac{|E(M)_n|}{2^n}, \quad n = 1, 2, 3, \ldots,$$

то легко видеть, что $\rho_n(E(M)) \geqslant 1/2$ для бесконечно большого числа значений n. Поэтому множество $E(M)$ непренебрежимо. ■

ЛИТЕРАТУРА

1. *Baker T., Gill J., and Solovay R.* Relativizations of the P=?NP question // SIAM J. Computing. 1975. V. 4. P. 431–442.
2. *Вялый М., Китаев А., Шень А.* Классические и квантовые вычисления. М.: МЦНМО, ЧеРо, 1999. 192 с.
3. *Гэри М., Джонсон Д.* Вычислительные машины и труднорешаемые задачи. М.: Мир, 1982. 419 с.
4. *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.

REFERENCES

1. *Baker T., Gill J., and Solovay R.* Relativizations of the P=?NP question. SIAM J. Computing, 1975, vol. 4, pp. 431–442.
2. *Vyalyi M., Kitaev A., and Shen A.* Classical and Quantum Computation (Graduate Studies in Mathematics). AMS, 2002. 272 p.
3. *Garey M. and Johnson D.* Computers and Intractability. N. Y., Freeman & Co, 1979. 340 p.
4. *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.

УДК 621.396:621.372

ПОСТРОЕНИЕ СЕТИ ХЕММИНГА НА ОСНОВЕ КРОССБАРА С БИНАРНЫМИ МЕМРИСТОРАМИ

М. С. Тарков

Институт физики полупроводников им. А. В. Ржанова СО РАН, г. Новосибирск, Россия

Описаны свойства аналоговых и бинарных мемристоров (резисторов с памятью), которые могут быть использованы для аппаратной реализации синапсов нейронов, а также мемристорные матрицы, называемые кроссбарами. Бинарные мемристоры, сопротивление которых принимает только два значения (максимальное и минимальное), основаны на механизме переключения филамента и распространены более широко, чем аналоговые мемристоры. Они гораздо более устойчивы к статистическим флуктуациям по сравнению с аналоговыми. Предложена аппаратная реализация ассоциативной памяти Хемминга на основе использования кроссбара на бинарных мемристорах и КМОП-схемотехники. Максимальное сопротивление бинарного мемристора соответствует значению -1 компоненты хранимого эталонного вектора, а минимальное — значению $+1$. Показано, что кроссбар на бинарных мемристорах реализует свойства первого слоя сети Хемминга, согласно которым выходной сигнал нейрона первого слоя неотрицателен. При этом он максимален для нейрона, эталонный вектор которого наиболее близок к вектору входных данных. Для заданной размерности эталонного вектора получено соотношение между максимальным и минимальным сопротивлениями бинарных мемристоров, которое гарантирует корректную работу первого слоя сети Хемминга. Моделирование в системе LTSPICE предложенной схемы памяти Хемминга подтвердило её работоспособность.

Ключевые слова: *ассоциативная память Хемминга, мемристор, кроссбар, КМОП-технология, LTSPICE.*

DOI 10.17223/20710410/40/9

CONSTRUCTION OF A HAMMING NETWORK BASED ON A CROSSBAR WITH BINARY MEMRISTORS

M. S. Tarkov

Rzhanov Institute of Semiconductor Physics SB RAS, Novosibirsk, Russia

E-mail: tarkov@isp.nsc.ru

The properties of analog and binary memristors (resistors with memory) are described. The memristors can be used for the hardware implementation of neurons synapses. The memristor matrices are called crossbars. The binary memristors, whose resistance takes only two values (maximum and minimum), are based on the switching filament mechanism and are distributed more widely than analog memristors. They are much more stable to statistical fluctuations compared to analog memristors. The Hamming associative memory's hardware realization based on the use of a binary memristors crossbar and CMOS circuitry is proposed. The maximum binary memristor resistance corresponds to the stored reference vector component value -1, and the minimum resistance corresponds to the value $+1$. It is shown that the binary memristors crossbar

realizes the Hamming network first layer properties according to which the output first layer neuron signal is non-negative. This signal is maximal for a neuron with the reference vector closest to the input vector. For a given reference vector dimension, the relationship between the maximum and minimum binary memristors resistances is obtained. It guarantees the Hamming network first layer correct operation. Simulation in the LTSPICE system of the proposed Hamming memory scheme confirmed its operability.

Keywords: *associative Hamming memory, memristor, crossbar, CMOS-technology, LTSPICE.*

Введение

Искусственная нейронная сеть обычно использует матрицу весовых коэффициентов для представления множества синапсов слоя нейронов. Соответственно вычисление активации слоя нейронов можно рассматривать как умножение этой матрицы весов на вектор входных сигналов слоя. Аппаратная реализация нейронной сети требует много памяти для хранения матрицы весов слоя нейронов и является дорогостоящей. Решение этой проблемы упрощается при использовании в качестве ячейки памяти устройства, называемого мемристором. Мемристор был предсказан теоретически в 1971 г. Леоном Чуа [1]. Первую физическую реализацию мемристора продемонстрировала в 2008 г. лаборатория фирмы «Hewlett Packard» в виде тонкоплёночной структуры TiO_2 [2]. В России первый мемристор на основе TiO_2 получен в 2012 г. в Тюменском государственном университете [3]. Мемристор имеет много достоинств, таких, как энергонезависимость хранения информации, малое потребление энергии, высокая плотность интеграции и замечательная масштабируемость. Уникальная способность сохранять следы возбуждения устройства делает его идеальным кандидатом для реализации синапсов в электронных нейронных сетях [4].

Мемристор ведет себя подобно синапсу: он «запоминает» полный электрический заряд, прошедший через него [5]. Память, основанная на мемристорах, может достигать очень высокой степени интеграции 100 Гбит/см^2, в несколько раз более высокой, чем на основе технологии флэш-памяти [6]. Эти уникальные свойства делают мемристор многообещающим устройством для создания массово-параллельных нейроморфных систем [7 – 9].

Понятие мемристора покрывает широкий диапазон явлений [10]:

а) изменение фазы, вызванное нагреванием в устройствах памяти, основанных на халькогенидах;
б) образование проводящего филамента, вызванное электрохимическими процессами окисления/восстановления, наблюдаемыми в оксидах металлов, халькогенидах и полимерах;
в) дрейф/диффузия ионов в аморфных плёнках;
г) образование проводящего филамента, вызванное нагреванием и наблюдаемое в некоторых оксидах.

В «аналоговых» мемристорах (например, на основе окиси титана TiO_2) непрерывное изменение напряжения вызывает непрерывное и постепенное (инкрементное) изменение проводимости. Тот же эффект достигается при применении последовательности положительных или отрицательных программирующих импульсов. Сопротивление аналогового мемристора (рис. 1) может быть представлено [2] как

$$M(p) = p \cdot R_{\text{on}} + (1 - p)R_{\text{off}},$$

где p $(0 \leqslant p \leqslant 1)$ — положение фронта примеси относительно полной толщины h плёнки TiO_2; R_{on} и R_{off} — минимальное и максимальное сопротивления мемристора. Когда на мемристор подаётся напряжение V выше некоторого порога V_{th}, его сопротивление уменьшается за счёт расширения примесной зоны D, имеющей низкое сопротивление, и сокращения зоны U чистого окисла, имеющей высокое сопротивление. Соответственно сопротивление мемристора возрастает при подаче напряжения V ниже $-V_{th}$ за счёт сокращения зоны D и расширения зоны U. После отключения напряжения текущее сопротивление мемристора сохраняется.

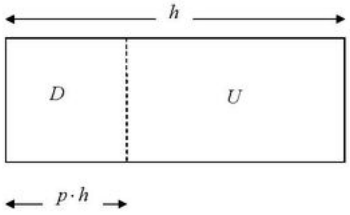

Рис. 1. Мемристор

В «цифровых» (бинарных) мемристорах сопротивление изменяется резко (с низкого на высокое и обратно), когда приложено напряжение, «пересекающее» порог. Скачкообразное изменение сопротивления обусловлено образованием проводящего филамента. Например, при образовании Ag-филамента в аморфном кремнии (a-Si, который служит материалом электролита) наблюдается устойчивый эффект резкого изменения сопротивления. Такой мемристор имеет чёткий порог переключения и резкое изменение сопротивления при напряжении считывания 1 В. Время переключения составляет менее 10 нс. Особенно интересны реализации синаптических устройств на окиси гафния HfO_2, поскольку такая память имеет блестящие характеристики: большая масштабируемость (до размеров менее 10 нм), малое время переключения (порядка 1 нс).

Бинарные мемристоры [11, 12], основанные на механизме переключения филамента, распространены более широко, чем аналоговые, материалы для которых встречаются значительно реже и требуют более сложного процесса обработки. При переключении филамента мемристоры могут иметь либо высокое сопротивление (ВС), либо низкое (НС). Таким образом, мы можем хранить только 1 или 0 (-1) при переключении филамента бинарного мемристора. Бинарные мемристоры гораздо более устойчивы к статистическим флуктуациям по сравнению с аналоговыми. Это связано с тем, что ВС намного выше, чем НС, несмотря на большое количество флуктуаций в НС и ВС.

На рис. 2 представлен мемристорный кроссбар, который является типичной структурой мемристорной памяти. Он содержит мемристор на каждом пересечении горизонтальных и вертикальных проводников. Мемристорный кроссбар интересен для реализации матриц соединений в нейронных сетях, поскольку он может обеспечить большое число сигнальных связей и вычислить взвешенную комбинацию входных сигналов. На вертикальные шины кроссбара подаются компоненты вектора входных сигналов. Каждая горизонтальная шина позволяет вычислить скалярное произведение вектора входных сигналов на вектор весов, заданных проводимостями мемристоров соответствующей строки кроссбара.

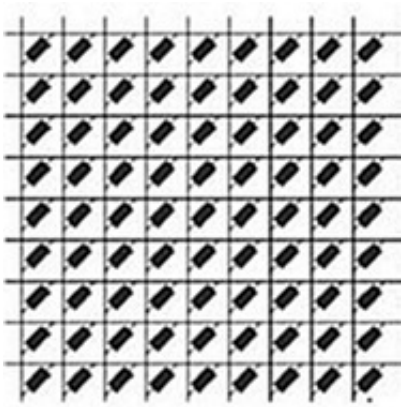

Рис. 2. Мемристорный кроссбар

1. Программирование кроссбара на бинарных мемристорах

Установка сопротивлений бинарных мемристоров кроссбара осуществляется на основе методики, предложенной в [12]. Рассмотрим установку мемристора в состояние +1 (НС) на примере фрагмента кроссбара, представленного на рис. 3. Для определённости рассмотрим мемристор M11. Чтобы сделать низким сопротивление этого мемристора, сделаем установки напряжений, указанные в таблице (в вольтах):

H1	H2	H3	V1	V2	V3
+7	0	0	−7	+7	+7

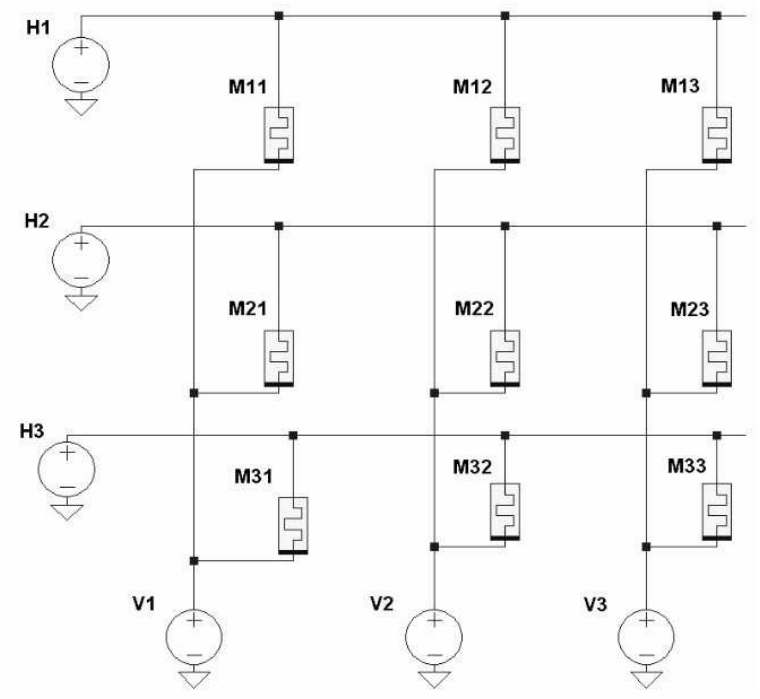

Рис. 3. Фрагмент кроссбара

Эксперимент показал, что при таких установках напряжений сопротивление мемристора M11 станет минимальным (100 Ом) в течение 2 нс. При этом сопротив-

ления мемристоров M12 и M13 останутся неизменными, а сопротивления остальных мемристоров (M21, M22, M23, M31, M32, M33) изменятся незначительно (не более 0,5 %).

Чтобы установить мемристор M11 в состояние -1 (ВС), нужно поменять полярность напряжений в таблице. При этом время установки увеличивается до 5 нс.

2. Сеть Хемминга и её реализация с использованием мемристорного кроссбара

В сети Хемминга [13] нейроны первого слоя рассчитывают расстояния Хемминга между поданным на вход сети вектором x и векторами весов (эталонными векторами) x^i нейронов этого слоя. Расстоянием Хемминга $\mathrm{d}(x, y)$ между векторами x и y называется количество несовпадающих компонент этих векторов. Значения выходных сигналов нейронов первого слоя определяются по формуле

$$y_i = 1 - \mathrm{d}(x^i, x)/N, \tag{1}$$

где N — число компонент вектора x. Из (1) следует, что $y_i \in [0, 1]$, причём для $x = x^i$ выполняется $y_i = 1$ (максимально возможное значение).

Сигналы y_i становятся начальными состояниями нейронов второго слоя, которые функционируют по принципу WTA («Winner Takes All»). Этот слой определяет победителя среди нейронов первого слоя, т. е. нейрон, выходной сигнал которого наиболее близок к 1. Такой нейрон указывает на эталонный вектор x^i с минимальным расстоянием Хемминга до входного вектора x.

Поставим в однозначное соответствие эталонному вектору сети Хемминга x^i, $i = 1, \ldots, p$, вектор весов w^i, задаваемых проводимостями мемристоров кроссбара, по правилу

$$w^i_j = \begin{cases} g_{\mathrm{L}}, & \text{если } \ x^i_j = -1, \\ g_{\mathrm{H}}, & \text{если } \ x^i_j = 1, \end{cases}$$

где $g_{\mathrm{H}} > g_{\mathrm{L}}$, $j = 1, \ldots, N$. Пусть w — произвольный вектор размерности N с компонентами g_{L} и g_{H}.

Утверждение 1. $(x^i, w^i) - (x^i, w) \geqslant g_{\mathrm{H}} - g_{\mathrm{L}} > 0$, если $w \neq w^i$.

Доказательство. Пусть m компонент вектора w^i равны g_{H}, а n компонент — g_{L}, $m + n = N$. Согласно определению вектора x^i, имеем

$$(x^i, w^i) = mg_{\mathrm{H}} - ng_{\mathrm{L}}. \tag{2}$$

Чтобы получить из w^i вектор w, нужно заменить k ($k \in \{1, \ldots, m\}$) компонент g_{H} на g_{L} и/или l ($l \in \{1, \ldots, n\}$) компонент g_{L} на g_{H}. Тогда

$$(x^i, w) = (x^i, w^i) + k(g_{\mathrm{L}} - g_{\mathrm{H}}) - l(g_{\mathrm{H}} - g_{\mathrm{L}}). \tag{3}$$

Из (2) и (3) следует $(x^i, w^i) - (x^i, w) = (k + l)(g_{\mathrm{H}} - g_{\mathrm{L}}) \geqslant g_{\mathrm{H}} - g_{\mathrm{L}} > 0$, так как $g_{\mathrm{H}} > g_{\mathrm{L}}$. ■

Из утверждения 1 следует, что вычисления по формуле (1) можно заменить на вычисление скалярного произведения (x, w^i), которое может быть реализовано на мемристорном кроссбаре. Мемристорный кроссбар с указанными выше свойствами не реализует формулу (1) вычислений первого слоя сети Хемминга, но обеспечивает требуемое свойство этого слоя: нейрон с весовым вектором w^i, представленный i-й строкой кроссбара, для эталонного вектора w^i дает максимальный выходной сигнал $y_i = (x^i, w^i)$.

Будем называть вектор w нетривиальным, если хотя бы одна его компонента равна g_{H}. Тогда $(x^i, w) \geqslant g_{\mathrm{H}} - (N - 1)g_{\mathrm{L}}$, откуда следует

Утверждение 2. $(x^i, w) > 0$ для нетривиального w, если

$$g_{\mathrm{L}} \leqslant \frac{g_{\mathrm{H}}}{N-1}. \tag{4}$$

Неравенство (4) задаёт ограничение на отношение между максимальной g_{H} и минимальной g_{L} проводимостями мемристоров кроссбара, которое обеспечивает неотрицательность выходного сигнала нейрона первого слоя сети Хемминга. Предложенный здесь подход к построению ассоциативной памяти Хемминга позволил сократить вдвое количество используемых мемристоров по сравнению с реализацией в [14].

3. Пример реализация сети Хемминга

На рис. 4 представлены изображения символов «L», «T», «X» размером 3×3 пикселей. Эти изображения могут быть описаны эталонными векторами (построчная развертка изображений)

$$L = (1, -1, -1, 1, -1, -1, 1, 1, 1),$$
$$T = (1, 1, 1, -1, 1, -1, -1, 1, -1),$$
$$X = (1, -1, 1, -1, 1, -1, 1, -1, 1),$$

где 1 соответствует белым пикселям, а -1 — чёрным.

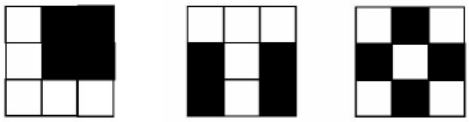

Рис. 4. Изображения символов

Запоминание этих изображений в памяти Хемминга реализовано мемристорным кроссбаром, представленным на рис. 5, где каждая строка соответствует одному из изображений. Компоненте 1 эталонного вектора в кроссбаре соответствует мемристор с минимальным сопротивлением $R_{\mathrm{L}} = 100\,\mathrm{Ом}$ (с максимальной проводимостью $g_{\mathrm{H}} = 1/R_{\mathrm{L}}$), а компоненте -1 — мемристор с максимальным сопротивлением $R_{\mathrm{H}} = 100\,\mathrm{кОм}$ (с минимальной проводимостью $g_{\mathrm{L}} = 1/R_{\mathrm{H}}$). Обозначив векторы весов через w^L, w^T и w^X, можем записать выходные сигналы строк кроссбара (рис. 5):

$$\mathrm{SumL} = (x, w^L), \quad \mathrm{SumT} = (x, w^T), \quad \mathrm{SumX} = (x, w^X),$$

где x — вектор входных сигналов, компоненты которого — напряжения 0,3 В (соответствует 1) и $-0{,}3$ В (соответствует -1). Модуль напряжений выбран ниже порога бинарного мемристора, чтобы исключить возможность изменения проводимости мемристоров кроссбара при его эксплуатации.

Второй слой сети Хемминга (рис. 6) реализован на основе КМОП-технологии [14, 15] и преобразует вектор входных сигналов по принципу WTA, то есть максимальной входной компоненте он ставит в соответствие 1 на выходе (высокое напряжение, для схемы рис. 6 оно составляет около 4,5 В), а остальным входным компонентам ставится в соответствие 0 (напряжение, близкое к нулю, для схемы рис. 6 оно составляет около 0,1 мВ). Для примера на рис. 5 выполняется $\mathrm{SumT} > \mathrm{SumL}$ и $\mathrm{SumT} > \mathrm{SumX}$. В этом случае на выходе второго слоя (рис. 6) получаем $\mathrm{OutT} \approx 4{,}5$ В, $\mathrm{OutL} = \mathrm{OutX} \approx 0{,}1$ мВ. Схема памяти Хемминга (рис. 5 и 6) реализована в системе моделирования LTSPICE [16].

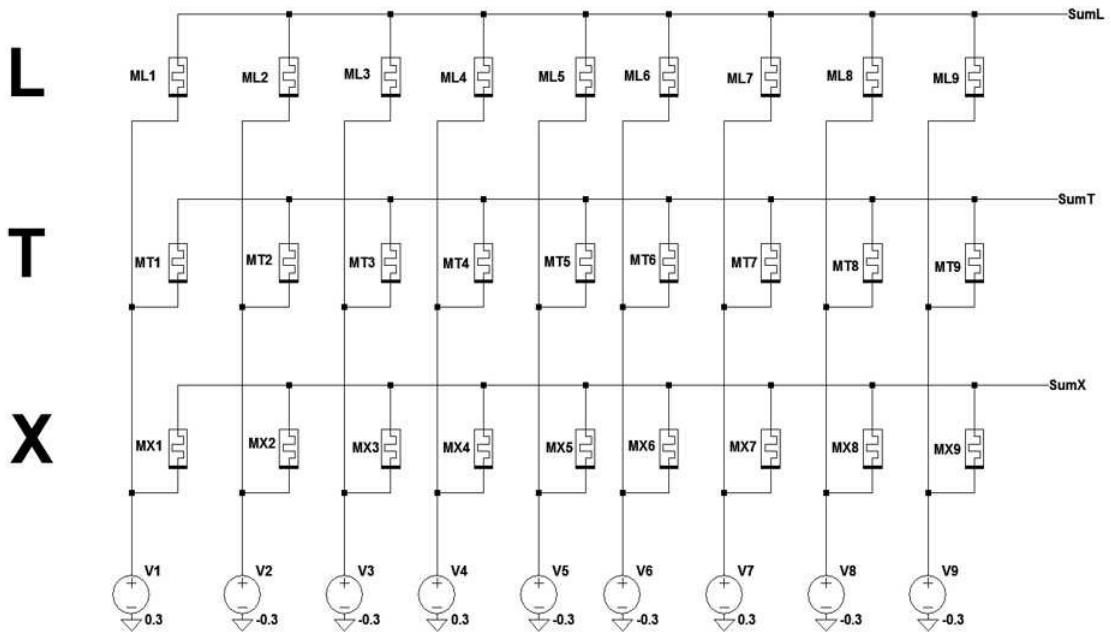

Рис. 5. Первый слой сети Хемминга

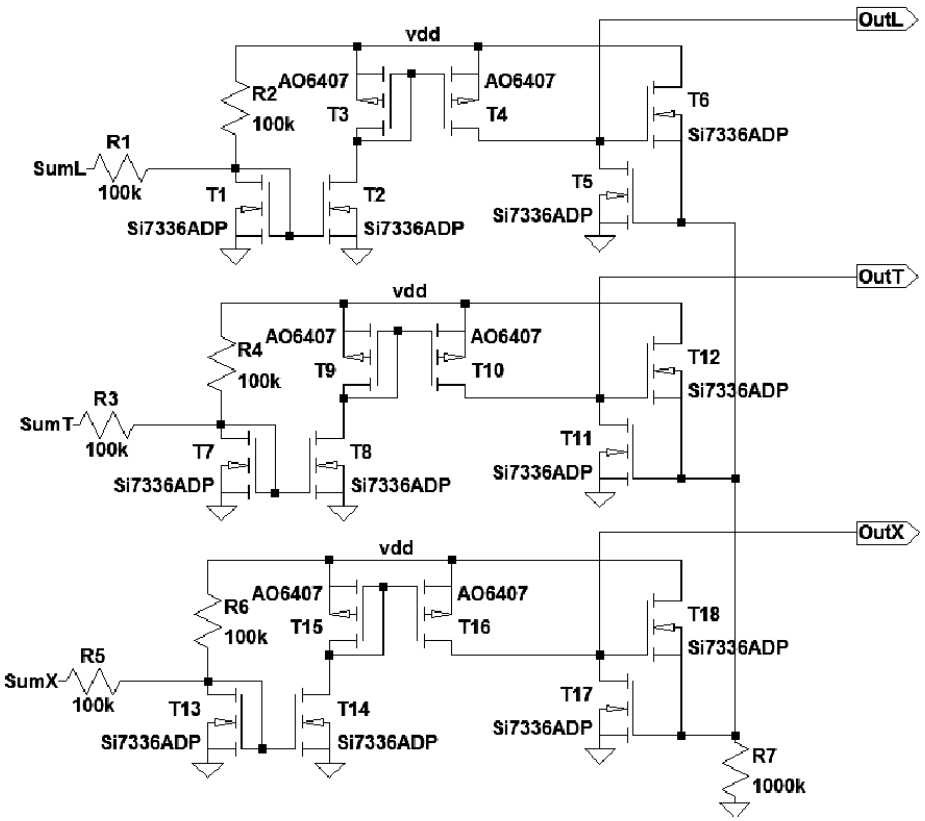

Рис. 6. Второй слой сети Хемминга (vdd = 7 В)

Заключение

Предложена аппаратная реализация ассоциативной памяти Хемминга на основе использования кроссбара на бинарных мемристорах и КМОП-схемотехники. Показано,

что кроссбар на бинарных мемристорах, где максимальное сопротивление мемристора соответствует значению -1 компоненты хранимого эталонного вектора, а минимальное — значению $+1$, реализует свойства первого слоя сети Хемминга, согласно которым выходной сигнал нейрона первого слоя неотрицателен, причём он максимален для нейрона, эталонный вектор которого наиболее близок к входному. Для заданной размерности эталонного вектора получено соотношение между максимальным и минимальным сопротивлениями бинарных мемристоров, которое гарантирует корректную работу первого слоя сети Хемминга. Моделирование в системе LTSPICE предложенной схемы памяти Хемминга подтвердило её работоспособность.

ЛИТЕРАТУРА

1. *Chua L.* Memristor — the missing circuit element // IEEE Trans. Circuit Theory. 1971. V. 18. P. 507–519.
2. *Strukov D. B., Snider G. S., Stewart D. R., and Williams R. S.* The missing memristor found // Nature. 2008. V. 453. P. 80–83.
3. `http://www.utmn.ru/presse/teleradiokanal-evrazion/videonovosti-tyumgu/89986/`
4. *Pershin Y. and Di Ventra M.* Experimental demonstration of associative memory with memristive neural networks // Neural Networks. 2010. V. 23. No. 7. P. 881–886.
5. *Chua L.* Resistance switching memories are memristors // Appl. Phys. A: Mater. Sci. & Process. 2011. V. 102. No. 4. P. 765–783.
6. *Ho Y., Huang G. M., and Li P.* Nonvolatile memristor memory: device characteristics and design applications // Proc. Int. Conf. ICCAD. 2009. P. 485–490.
7. *Jo S. H., Chang T., Ebong I., et al.* Nanoscale memristor device as synapse in neuromorphic systems // Nanoletters. 2010. V. 10. No. 4. P. 1297–1301.
8. *Kavehei O.* Memristive Devices and Circuits for Computing, Memory, and Neuromorphic Applications. PhD Thesis. The University of Adelaida, Australia, 2011.
9. *Lehtonen E.* Memristive Computing. Doctoral Thesis. University of Turku, Finland, 2012.
10. *Lu W., Kim K.-H., Chang T., and Gaba S.* Two-terminal resistive switches (memristors) for memory and logic applications // Proc. 16th Asia and South Pacific Design Automation Conf., Yokohama, Japan, 2011.
11. *Truong S. N., Ham S.-J., and Min K.-S.* Neuromorphic crossbar circuit with nanoscale filamentary-switching binary memristors for speech recognition // Nanoscale Res. Lett. 2014. No. 9:629. `http://www.nanoscalereslett.com/content/9/1/629`
12. *Yakopcic C., Taha T. M., Subramanyam G., and Pino R. E.* Memristor SPICE model and crossbar simulation based on devices with nanosecond switching time // Proc. Int. Joint Conf. Neural Networks. Dallas, Texas, USA. August 4–9, 2013.
13. *Осовский С.* Нейронные сети для обработки информации. М.: Финансы и статистика, 2002. 344 с.
14. *Zhu X., Yang X., Wu C., et al.* Hamming network circuits based on CMOS/memristor hybrid design // IEICE Electronics Express. 2013. V. 10. No. 12. P. 1–9.
15. *Lazzaro J., Ryckebusch S., Mahowald M. A., and Mead C. A.* Winner-take-all networks of O(n) complexity // Advances in Neural Information Processing Systems. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1989. P. 703–711.
16. *Володин В. Я.* Компьютерное моделирование электронных схем. СПб.: БХВ-Петербург, 2010. 400 с.

REFERENCES

1. *Chua L.* Memristor — the missing circuit element. IEEE Trans. Circuit Theory, 1971, vol. 18, pp. 507–519.
2. *Strukov D. B., Snider G. S., Stewart D. R., and Williams R. S.* The missing memristor found. Nature, 2008, vol. 453, pp. 80–83.
3. `http://www.utmn.ru/presse/teleradiokanal-evrazion/videonovosti-tyumgu/89986/`
4. *Pershin Y. and Di Ventra M.* Experimental demonstration of associative memory with memristive neural networks. Neural Networks, 2010, vol. 23, no. 7, pp. 881–886.
5. *Chua L.* Resistance switching memories are memristors. Appl. Phys. A: Mater. Sci. & Process., 2011, vol. 102, no. 4, pp. 765–783.
6. *Ho Y., Huang G. M., and Li P.* Nonvolatile memristor memory: device characteristics and design applications. Proc. Int. Conf. ICCAD, 2009, pp. 485–490.
7. *Jo S. H., Chang T., Ebong I., et al.* Nanoscale memristor device as synapse in neuromorphic systems. Nanoletters, 2010, vol. 10, no. 4, pp. 1297–1301.
8. *Kavehei O.* Memristive Devices and Circuits for Computing, Memory, and Neuromorphic Applications. PhD Thesis. The University of Adelaida, Australia, 2011.
9. *Lehtonen E.* Memristive Computing. Doctoral Thesis. University of Turku, Finland, 2012.
10. *Lu W., Kim K.-H., Chang T., and Gaba S.* Two-terminal resistive switches (memristors) for memory and logic applications. Proc. 16th Asia and South Pacific Design Automation Conf., Yokohama, Japan, 2011.
11. *Truong S. N., Ham S.-J., and Min K.-S.* Neuromorphic crossbar circuit with nanoscale filamentary-switching binary memristors for speech recognition. Nanoscale Res. Lett., 2014, no. 9:629. `http://www.nanoscalereslett.com/content/9/1/629`
12. *Yakopcic C., Taha T. M., Subramanyam G., and Pino R. E.* Memristor SPICE model and crossbar simulation based on devices with nanosecond switching time. Proc. Int. Joint Conf. Neural Networks, Dallas, Texas, USA, August 4–9, 2013.
13. *Osovskiy S.* Neironnye seti dlya obrabotki informatsii [Neural Networks for Data Processing]. Moscow, Finansy i Statistika, 2002. 344 p. (in Russian)
14. *Zhu X., Yang X., Wu C., et al.* Hamming network circuits based on CMOS/memristor hybrid design. IEICE Electronics Express, 2013, vol. 10, no. 12, pp. 1–9.
15. *Lazzaro J., Ryckebusch S., Mahowald M. A., and Mead C. A.* Winner-take-all networks of O(n) complexity. Advances in Neural Information Processing Systems. San Francisco, CA, USA, Morgan Kaufmann Publishers Inc., 1989, pp. 703–711.
16. *Volodin V. Ya.* Kompyuternoe modelirovanie elektronnykh skhem [Computer Simulation of Electronic Circuitry] SPb., BHV-Peterburg Publ., 2010. 400 p. (in Russian)

ДИСКРЕТНЫЕ МОДЕЛИ РЕАЛЬНЫХ ПРОЦЕССОВ

УДК 519.17

ФОРМАЛЬНАЯ ГРАММАТИКА РУССКОГО ЯЗЫКА В ПРИМЕРАХ

Я. М. Мирзабеков, Ш. Б. Шихиев

Дагестанский государственный университет, г. Махачкала, Россия

На примере одного предложения показана возможность построения конструктивной теории естественного языка и реализации морфологии и синтаксиса на компьютере путём формализации соответствующих правил. В лексическом составе рассматриваемого примера несколько слов. Грамматика русского языка упрощена. Модель морфологии определяется лексическим наполнением предложений и правилами морфологии, которые хранятся в словообразовательном (морфологическом) словаре. В компьютерной модели морфологии реализованы два алгоритма: синтез и анализ словоформ. Модель синтаксиса языка определяется его морфологией и правилами его синтаксиса. Правила синтаксиса порождают словосочетания, используемые в языке. Правила синтаксиса представлены в виде пар (прямых производных слов) разных лексических групп. Лексическая группа есть множество словоформ, имеющих одинаковые морфологические параметры. Словосочетания «оставляют» на лексическом наполнении предложений некоторый граф, который и представляет собой синтаксис. В терминах теории графов определяются понятия синтаксиса, в частности понятие предложения. Формулируются две задачи: синтез и анализ предложения; описаны и реализованы алгоритмы их решения в среде программирования Delphi 7.0.

Ключевые слова: *естественный язык, морфология, синтаксис, словоформа, правила морфологии, правила синтаксиса, словосочетание, реализация алгоритма на компьютере, компьютерная модель, граф, дерево, анализ предложения.*

DOI 10.17223/20710410/40/10

FORMAL GRAMMAR OF RUSSIAN LANGUAGE IN EXAMPLES

Ya. M. Mirzabekov, Sh. B. Shihiev

Dagestan State University, Makhachkala, Russia

E-mail: yash831@mail.ru

The possibility of a constructive theory creation for a natural language and of its morphology and syntax implementation on a computer by formalizing the appropriate rules is shown in the article. There are a few words in the lexicon of the example under consideration. The grammar of the Russian language is simplified. The morphology model is determined by the vocabulary and the rules of morphology which are stored in the word-morphological (morphological) dictionary. In the computer model of morphology, two algorithms are implemented: synthesis and analysis of word forms. The syntax model is defined by the morphology and syntax rules. Syntax rules generate word combinations used in the language. The rules of syntax are presented in

the form of pairs (direct products) of lexical groups. A lexical group is a set of word forms that have the same morphological parameters. Phrases are defined in the lexicon of some graph which is a syntax. The concepts of syntax are defined in terms of graph theory. In particular, the concept of the sentence in a syntax is defined. Then, two problems are formulated: synthesis and analysis of the sentence. Algorithms for solution of them are described and implemented in the Delphi 7.0 programming environment.

Keywords: *natural language, morphology, syntax, word form, morphology rules, syntax rules, the combinations of two words, the implementation of the algorithm on the computer, the computer model, graph, tree, analysis of the sentence.*

Формулировка задачи

Если сочетание слов «компьютерная лингвистика» обозначает раздел лингвистики, предмет, методы и задачи которого представляются и реализуются на компьютере, то, разумеется, надо иметь описание естественного языкового явления в логико-технических терминах вычислительной машины. Такое описание принято называть компьютерной моделью языка или программой, реализующей на компьютере алгоритмы решения задач лингвистики. Из известных формулировок задачи моделирования естественного языка наиболее корректной является следующая.

«Основным орудием исследования и описания» естественного языка, как утверждает И. А. Мельчук [1], является модель языка как «сложно организованной совокупности правил, чисто механическое применение которых должно» реализовать отображение (желательно, биекцию) двух множеств (субстанций, по Ф. де Соссюру): «планов содержания» и «планов выражения» (по Л. Ельмслеву) или «смыслов» и «текстов» (по Мельчуку). Но ни одна такая модель, продолжает И. А. Мельчук, «хотя бы в виде чисто экспериментального варианта» ему не известна. «Можно представить себе два пути упрощения этой задачи ... ограничившись её достаточно точно разработанными фрагментами... Второй путь — чтобы как-то были намечены обязательно все основные компоненты модели (пусть даже приблизительно) ... и необходимые связи между ними» [1].

Однако когда речь идёт о компьютерной модели такого сложного и ещё не изученного явления, как язык, общими и приблизительными описаниями задача не решается. В данной работе делается попытка показать один из возможных приёмов моделирования языка: построены синтаксические примитивы и продемонстрированы их возможности на примере.

Имеет смысл вспомнить и тот факт, что в начале прошлого века известный исследователь языка Л. Витгенштейн для демонстрации различных функций и возможностей языка рассматривал языки с небольшим числом слов. Такие примеры слов он назвал языковыми играми. Содержание данной работы можно называть и языковой игрой, подчеркивая значительные перспективы моделирования отдельных фрагментов естественного языка.

1. Морфология

Рассмотрим пример грамматики, лексика которой включает следующие слова и их формы:

1) **пол, дом, угол, топор, нож** (обозначение — ИС, код — 01);
2) **серый, белый, легкий, быстрый** (обозначение — ИП, код — 02);
3) **находиться, лежать, спать, строить** (обозначение — Гл, код — 07);

4) **у/2, с/2, с/5** (код предлога — 08).

В каждом пункте напротив слов указано сокращённое обозначение части речи и код. Номер падежа, которым он управляет, указан за предлогом.

В примере морфология включает две категории:

числа — код: 20 = (21, 22);
падежа — код: 30 = (31, 32, 33, 34, 35, 36).

Рассмотрим правила морфологии для этих слов. Категории числа и падежа являются словоизменительными для ИС и ИП. Следовательно, каждое из имён имеет по 12 беспредложных форм (ИП имеет только форму мужского рода). Каждое ИС имеет ещё шесть предложных форм, образованных в сочетании с тремя предлогами в единственном и множественном числах, например: **у дома, с домами, с пола** и т. д.

Глагол имеет форму настоящего времени третьего лица в единственном и множественном числах, например: **лежит, лежат** и т. д.

Чтобы иметь исчерпывающие сведения о морфологии для лексического набора из 16 слов и двух категорий, нужны описания морфологических правил, по которым образуются формы слов. Эти правила известны из грамматики русского языка, они же описаны в грамматических словарях, например в словаре А. А. Зализняка [2]. Там слово **пол** имеет следующий формат: «**пол м 1с**». Буква «м» означает «ИС мужского рода», а «1с» есть код морфологического правила, по которому образуются формы слова **пол**. Морфологические правила и их кодировки описаны в начале словаря.

В данной работе используется иная кодировка морфологических правил. ИС **пол**, например, имеет следующий формат:

пол01:2130:0,а,у,,ом,е,:2230:0ы,ов,ам,ы,ами,ах.

Данная конструкция (словарная статья) состоит из трёх частей, разделённых двоеточиями. В первой части — слово (**пол**) и код его лексической группы (**01**). Далее идут два морфологических правила.

Морфологические правила имеют унифицированный формат: состоят из пятизначного числа и последовательности постфиксов (окончаний). Рассмотрим подробно формат и содержание правила «**2130:0,а,у,,ом,е,**», в котором упакованы следующие четыре параметра данного правила:

1) 21 — формы единственного числа;
2) 30 — все падежные формы;
3) 0 — число букв, отбрасываемых справа от данного слова, чтобы получить его основу;
4) «,а,у,,ом,е,» — окончания, разделённые запятыми, приписываются к основе слова для получения форм данного слова.

Аналогично читается и второе правило: «**2230:0 ы,ов,ам,ы,ами,ах,**». Окончания слов имеет смысл хранить в отдельном списке, а в словаре вместо окончаний указывать их порядковый номер в списке.

Унифицированный формат словарной статьи даёт возможность описать процедуру MorfForms(s: string; var forms: smas12), которая по словарной статье строит все двенадцать форм любого из девяти имён. ИП имеет такой же формат, как ИС. Например:

белый 02:2130:2ый,ого,ому,ый,ым,ом,:2230:2ые,ых,ым,ые,ыми,ых.

Словарь (словообразовательный, грамматический или морфологический) из 16 слов приводится ниже:

1) пол 01:2130:0,а,у,,ом,е,:2230:0ы,ов,ам,ы,ами,ах;
2) дом 01:2130:0,а,у,,ом,е,:2230:0а,ов,ам,а,ами,ах;
3) угол 01:2130:0,а,у,,ом,е,:2230:2лы,лов,лам,лы,лами,лах;
4) топор 01:2130:0,а,у,,ом,е,:2230:0ы,ов,ам,а,ами,ах;
5) нож 01:2130:0,а,у,,ом,е,:2230:0и,ей,ам,и,ами,ах;
6) серый 02:2130:2ый,ого,ому,ый,ым,ом,:2230:2ые,ых,ым,ые,ыми,ых;
7) белый 02:2130:2ый,ого,ому,ый,ым,ом,:2230:2ые,ых,ым,ые,ыми,ых;
8) легкий 02:2130:2ий,ого,ому,ий,им,ом,:2230:2ие,их,им,ие,ими,их;
9) быстрый 02:2130:2ый,ого,ому,ый,ым,ом,:2230:2ые,ых,ым,ые,ыми,ых;
10) находиться 07: 21-находится 22-находятся;
11) лежать 07: 21-лежит 22-лежат;
12) спать 07: 21-спит 22-спят;
13) строить 07: 21-строит 22-строят;
14) у08:2;
15) с08:2;
16) с08:5.

Извлечение форм глагола и предлога из словарной статьи не представляет труда. Процедура MorfForms используется для синтеза форм всех слов лексики.

Анализ или распознавание словоформы является еще одной задачей морфологии. Она заключается в определении исходной формы слова и всех её морфологических параметров. Например, результатом анализа словоформ

полом, у пола, белым, белыми, лежит, у

являются следующие строки (исходные формы слов и их параметры):

«пол 01 21 35», «пол 01 21 32 14», «белый 02 22 35»,
«белым 02 21 35; 02 22 33», «лежит 07 21», «у08:2».

Привычные термины, например «ИП, множественное число, винительный падеж», записаны кодами «02 22 35». Четвёртый параметр в форме ИС указывает на порядковый номер предлога в словаре, более того, наличие этого параметра в форме ИС является признаком предложной формы имени. Словоформа **белым** имеет два набора морфологических параметров.

Процедура анализа словоформы — довольно трудоёмкий процесс из-за того, что словарная статья начинается с исходной формы слова. Например, чтобы обнаружить в словаре словоформу **белым**, потребуется построить все двенадцать форм слова **белый** и среди них найти данное слово. Сложность поиска (в общем случае) заключается в отсутствии информации о том, что слово **белым** окажется формой слова **белый**, а не **бельмо, белить** и т. д. Поэтому требуется искать слово по всему словарю: распаковать каждую словарную статью с помощью процедуры MorfForms и сравнить искомое слово со всеми элементами массива forms.

Словарь, содержащий в себе информацию для образования всех словоформ лексики, будем называть математической моделью морфологии (МММ).

2. Лексические группы

Лексическая группа — это множество, состоящее из словоформ с одинаковыми морфологическими параметрами. В синтаксисе, который строится в этом примере, участвуют следующие лексические группы.

Через ИС31 обозначим множество ИС в именительном падеже, оно образует лексическую группу 012031 и представляет собой объединение $012031 = 012131 \cup 012231$, где

1) $012131 = \{$**пол, дом, угол, топор, нож**$\}$;
2) $012231 = \{$**полы, дома, углы, топоры, ножи**$\}$.

Через ИС/pr обозначено множество ИС в предложной форме. Каждое ИС имеет по шесть предложных форм. Например: **у дома** и **у домов**, **с дома** и **с домов**, **с домом** и **с домами**. То есть предлог ведёт себя как элемент падежной информации. ИС/pr из 30 словоформ разбито на шесть лексических групп:

1) $01213201 = \{$**у пола, у дома, у угла, у топора, у ножа**$\}$;
2) $01213202 = \{$**с пола, с дома, с угла, с топора, с ножа**$\}$;
3) $01213503 = \{$**с полом, с домом, с углом, с топором, с ножом**$\}$;
4) $01223201 = \{$**у полов, у домов, у углов, у топоров, у ножа**$\}$;
5) $01223202 = \{$**с полов, с домов, с углов, с топоров, с ножа**$\}$;
6) $01223503 = \{$**с полами, с домами, с углами, с топорами, с ножами**$\}$.

Имена прилагательные имеют только полные формы и в словосочетаниях рассматриваемого синтаксиса участвуют только ИП из группы $ИП125 = 022131 \cup 022132 \cup \cup 022135 \cup 022231 \cup 022232$, где

1) $022131 = \{$**серый, белый, легкий, быстрый**$\}$;
2) $022132 = \{$**серого, белого, легкого, быстрого**$\}$;
3) $022135 = \{$**серым, белым, легким, быстрым**$\}$;
4) $022231 = \{$**серые, белые, легкие, быстрые**$\}$;
5) $022232 = \{$**серых, белых, легких, быстрых**$\}$;
6) $022135 = \{$**серыми, белыми, легкими, быстрыми**$\}$.

Лексическая группа Гл12, или $0720 = 0721 \cup 0722$, где

1) $0721 = \{$**находится, лежит, спит, строит**$\}$;
2) $0722 = \{$**находятся, лежат, спят, строят**$\}$.

Объединение лексических групп ИС31, ИС/pr, ИП125 и Гл12 обозначим через $LG1$. В нём всего $10 + 30 + 24 + 8 = 72$ словоформы.

Далее под лексической группой, если не уточнено её определение, подразумевается любая из перечисленных шестнадцати групп.

3. Синтаксис

Одним из основных понятий синтаксиса является понятие словосочетания. Формально, словосочетание есть пара словоформ. Но не каждая пара словоформ образует словосочетание. Синтаксис языка устроен так, что если пара словоформ $w1$ и $w2$ из лексических групп $L1$ и $L2$ соответственно образует словосочетание, то любая пара словоформ из этих групп образует словосочетание. В таких случаях будем говорить, что пара лексических групп $L1$ и $L2$ образует словосочетание; множество словосочетаний, образованных группами $L1$ и $L2$, обозначается $L1 \times L2$.

Словосочетания — неделимые и базовые единицы синтаксиса — задаются как элементы прямого произведения различных лексических групп. Описание и именование лексических групп необходимы для определения синтаксиса. Например: $\{$**дом, пол**$\}\times\{$**белый, крепкий**$\} = \{$**(дом, белый), (дом, крепкий), (пол, белый), (пол, крепкий)**$\}$. Переписывая пары словоформ в виде **«дом белый», «дом крепкий», «пол белый», «пол крепкий»** или, как принято в синтаксисе русского языка, **«бе-**

лый дом», «крепкий дом», «белый пол», «крепкий пол», получим словосочетания.

В таблице перечислены 34 синтаксических отношения (СО) (или 34 произведения лексических групп), порождающих словосочетания русского языка из словоформ $LG1$.

СО, порождающие словосочетания

№	СО	Комментарии
1	012131×0721	Подлежащее и сказуемое
2	012231×0722	
3	012131×022131	Согласованное определение
4	012231×022231	
5	01213201×022132	
6	01213202×022132	
7	01213503×022135	
8	01223201×022232	
9	01223202×022232	
10	01223503×022235	
11	012131×01213201	Отношение беспредложного ИС с предложным ИС
12	012131×01213202	
13	012131×01213503	
14	012131×01223201	
15	012131×01223202	
16	012131×01223503	
17	012231×01213201	
18	012231×01213202	Отношение беспредложного ИС с предложным ИС
19	012231×01213503	
20	012231×01223201	
21	012231×01223202	
22	012231×01223503	
23	0721×01213201	Отношение глагола с предложным ИС
24	0721×01213202	
25	0721×01213503	
26	0721×01223201	
27	0721×01223202	
28	0721×01223503	
29	0722×01213201	
30	0722×01213202	
31	0722×01213503	
32	0722×01223201	
33	0722×01223202	
34	0722×01223503	

Обозначим через $R1$ множество словосочетаний, содержащихся в СО из таблицы. В этих словосочетаниях присутствуют только 72 словоформы из $LG1$. Мы имеем граф $G = (LG1, R1)$, который называется синтаксисом нижнего уровня. Параллельно с G имеется граф $CodeG = (CodeLG1, CodeR1)$, который называется синтаксисом верхнего уровня или просто синтаксисом. Через $CodeR1$ обозначено множество из СО, перечисленных в таблице. Все применённые без определения понятия из теории графов можно найти в [3].

Графы G и $CodeG$ оправдывают свои названия в том смысле, что все понятия синтаксиса будут определены в терминах этих графов, в частности понятие предложения. Граф $CodeG$ как математическая структура представляет собой математическую

модель синтаксиса (ММС). Лексика $LG1$, ММЛ и ММС образуют математическую модель грамматики.

4. Предложение

Предложением в синтаксисе G считается любое корневое дерево с корнем из ИС1, если в нём присутствует дуга из $012131 \times 0721 \cup 012231 \times 0722$ [4]. Иначе говоря, в предложении должны быть подлежащее (из ИС1) и согласованное с ним по тем или иным морфологическим характеристикам сказуемое (из Гл1).

Корневое дерево в $CodeG$ с корнем в 012131 или 012231 называется синтаксической формой (СФ), если в нём присутствует дуга 012131×0721 или 012231×0722.

Между предложениями и СФ имеют место следующие отношения.

Если в СФ $(CodeV, CodeA)$ код каждой лексической группы заменить некоторым элементом этой группы, то получится некоторое предложение (V, A). В этом случае будем говорить, что предложение (V, A) порождено СФ $(CodeV, CodeA)$.

Два корневых дерева $RT1 = (V1, A1)$ и $RT2 = (V2, A2)$ из G имеют одинаковую синтаксическую форму, если они изоморфны и в изоморфном отображении $f: V1 \to V2$ вершины $v1$ из $V1$ и $f(v1)$ из $V2$ принадлежат одной и той же лексической группе. Если два предложения имеют одинаковые синтаксические формы, то они порождены одной и той же СФ. Все предложения, порождённые данной СФ, имеют одинаковые синтаксические формы. Таким образом, каждая СФ представляет собой правило, порождающее предложения определённого класса.

Размер графа G зависит от объёма лексики, размер графа $CodeG$ никак не зависит от этого. Граф $CodeG$ на самом деле есть редукция графа G. Вершинам графа $CodeG$ соответствуют лексические группы, а вершинам графа G — словоформы из этих лексических групп.

Граф $CodeG$ показан на рис. 1. В $CodeG$ клонированы (дублированы) вершины ИП125\16-1 (A1) и ИС2\30-1 (S1) для упразднения циклов. При редукции графа G в граф $CodeG$ были стянуты в одну дугу (графа $CodeG$) несколько параллельных дуг (графа G); их число указано у каждой дуги графа $CodeG$. В верхнем правом углу записаны условные обозначения вершин, в каждой вершине указано имя соответствующего ей множества и коды групп, входящих в это множество. После имени множества через «\» указано число элементов в нём. Структуры множеств ИП125\16-1 и ИП125\16-2, ИС2\30-1 и ИС2\30-2 идентичны.

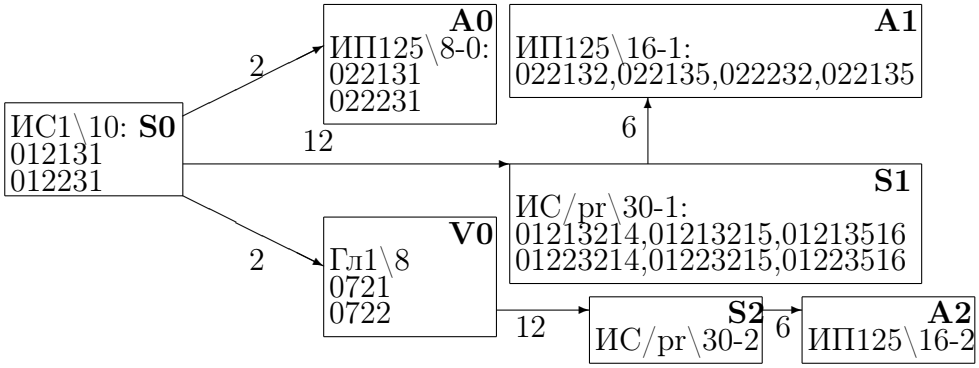

Рис. 1. Граф $CodeG$ без циклов

Нетрудно посчитать, что в графе на рис. 1 имеется 18 СФ. Выпишем их в скобочной форме, используя обозначения вершин, после записи дерева указано число СФ, содержащихся в этом дереве:

1) S0(V0) содержит 2 СФ;
2) S0(S1, V0) содержит 12 СФ;
3) S0(S1(A1), V0) содержит 12 СФ;
4) S0(A0, V0) содержит 2 СФ;
5) S0(A0, S1, V0) содержит 12 СФ;
6) S0(A0, S1(A1), V0) содержит 12 СФ;
7) S0(V0(S2)) содержит 12 СФ;
8) S0(S1, V0(S2)) содержит 12 СФ;
9) S0(S1(A1), V0(S2)) содержит 72 СФ;
10) S0(A0, V0(S2)) содержит 12 СФ;
11) S0(A0, S1, V0(S2)) содержит 72 СФ;
12) S0(A0, S1(A1), V0(S2)) содержит 72 СФ;
13) S0(V0(S2(A2))) содержит 12 СФ;
14) S0(S1, V0(S2(A2))) содержит 12 СФ;
15) S0(S1(A1), V0(S2(A2))) содержит 72 СФ;
16) S0(A0, V0(S2(A2))) содержит 12 СФ;
17) S0(A0, S1, V0(S2(A2))) содержит 72 СФ;
18) S0(A0, S1(A1), V0(S2(A2))) содержит 72 СФ.

Получится внушительное число: 556 СФ в небольшом графе $(LG1, R1)$. Последнему дереву с номером 18 соответствует сам граф $CodeG$. Выпишем одину из 72 СФ, соответствующих этому дереву:

$$012131(022131, 01213516(022135), 0721(01213214(022132). \tag{1}$$

Каждый из семи кодов групп, входящих в (1), может быть заменён любой словоформой из этой группы. Следовательно, СФ (1) порождает $5{\cdot}4{\cdot}5{\cdot}4{\cdot}4{\cdot}5{\cdot}4 = 32000$ предложений.

Замечание 1. Если к лексике $LG1$ добавить тысячи новых имён и глаголов, размер графа $CodeG$ не изменится.

Тем не менее 556 — большое число, когда речь идет о правилах синтаксиса, порождающих предложения. Однако видно, что все 18 деревьев — поддеревья дерева 18, а все 556 правил — производные от 72 правил. Поэтому можно считать, что правило в рассматриваемом синтаксисе одно: задано оно деревом 18 и допустимым набором лексических групп, прикреплённых к его вершинам. Это важное для синтаксиса утверждение демонстрируется ниже на одном примере синтеза и анализа предложения.

Представим дерево (1) в привычном виде (рис. 2) и назовём его «Правило 1». В вершинах графа за кодом лексической группы указана переменная, определённая в этой группе. У каждой дуги дерева указано соответствующее ей СО:

$$\begin{gathered}\mathbf{A} = 01112131 \times 02112131,\\ \mathbf{B} = 01112131 \times 02112131,\\ \mathbf{D} = 01112131 \times 02112131,\\ \mathbf{C} = 01112131 \times 02112131,\\ \mathbf{F} = 01112131 \times 02112131,\\ \mathbf{G} = 01112131 \times 02112131.\end{gathered}$$

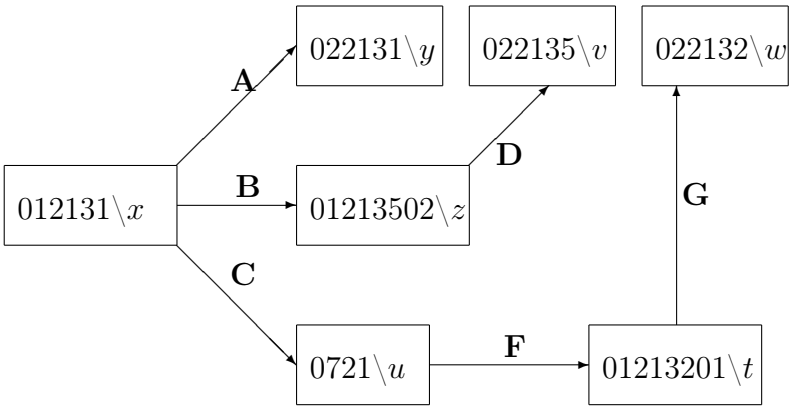

Рис. 2. Пример СФ. Правило 1

Дерево можно записать через его дуги: **0 (A, B(D), C(F(G)))**. Сделано допущение, что корень дерева является концом фиктивной дуги «0».

Присваивая переменным x, y, z, u, v, w, t из Правила 1 значения из соответствующих групп, можно получить различные предложения. Этот процесс называется синтезом предложений. На рис. 3 показан пример предложения, синтезированного по Правилу 1.

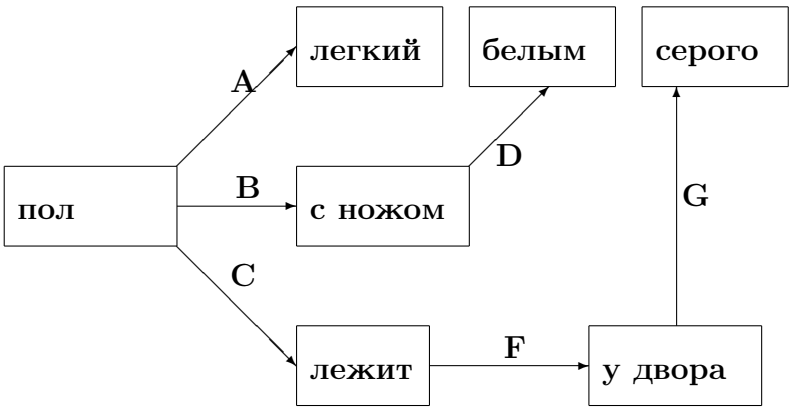

Рис. 3. Пример предложения, синтезированного по Правилу 1

Предложение, соответствующее рис. 3, в скобочной форме выглядит так:

пол (легкий, с ножом(белым), лежал (на дворе (сером))). (2)

Предложение-дерево (2) трансформируется в предложение-последовательность

легкий пол с белым ножом лежал на сером дворе

по определённому правилу: односторонним обходом предложения-дерева и перестановкой местами ИС и ИП.

Задача анализа обратна задаче синтеза: для заданного предложения требуется найти порождающее его правило. Обобщением задачи анализа является задача построения предложения из элементов заданной последовательности словоформ.

Эти две задачи отличаются по той причине, что между фрагментами предложения-последовательности (отрезком из соседних словоформ в последовательности) и сегментами предложения-дерева (членами предложения, принадлежащих одной ветви предложения-дерева) имеют место известные в грамматике закономерности. Например, сегмент сказуемого (члены предложения, принадлежащие ветви, корнем которого является сказуемое) и сегмент подлежащего делят предложение на два непересекающихся фрагмента.

Рассмотрим следующий пример задачи анализа предложения. Дана строка WFs, состоящая из словоформ, разделенных пробелами:

легкий пол с белым ножом лежал на сером дворе.

Требуется выяснить: образуют ли предложение словоформы из WFs? Решается эта задача следующим образом.

Ш а г 1. Из строки WFs выделяются все его словоформы и подвергаются морфологическому анализу. Для нашего примера имеем:

1) **легкий** — **легкий** 022131, 022134;
2) **пол** — **пол** 012131, 012134;
3) **белым** — **белым** 022135, 022233;
4) **с** — **с** 0832, 0835;
5) **ножом** — **ножом** 011135;
6) **лежит** — **лежит** 0721;
7) **серого** — **сером** 022132;
8) **у** — **у** 0832;
9) **двора** — **дворе** 012132.

Ш а г 2. Предлоги прикрепляются к соответствующим ИС, и таким образом образуются предложные формы ИС. В нашем примере они таковы:

1) **легкий** — **легкий** 022131, 022134;
2) **пол** — **пол** 012131, 012134;
3) **белым** — **белым** 022135, 022233;
4) **с ножом** — **ножом** 01113502;
5) **лежит** — **лежит** 0721;
6) **серого** — **сером** 022132;
7) **у двора** — **дворе** 01213201.

Дальнейшие действия зависят от решаемой задачи. Для построения предложения из заданных словоформ строится граф на множестве из семи лексических групп, к которым принадлежат данные слова. Если слово принадлежит нескольким группам (как в случае слов **легкий**, **пол** и **белым**), то следует рассмотреть всевозможные варианты. Строится граф следующим образом. Если $WFsCode$ — множество кодов групп, которым принадлежат данные слова, а $Codes$ — множество кодов из таблицы (с. 119), то множество дуг графа будет $(WFsCode \times WFsCode) \cap Codes$. Любая СФ (корневое дерево) в этом графе будет структурой предложения из заданных слов.

Рассматриваемый пример задачи анализа предложения проще: образуют ли предложение словоформы из WFs (в том порядке, в котором они находятся в WFs)? Это значит, что есть возможность воспользоваться следующими правилами расположения членов предложения в Правиле 1:

1) предлог прикреплён к первому ИС, которое находится после него;
2) ИП связано с первым ИС, которое находится после него;
3) подлежащее предшествует сказуемому;
4) группа (сегмент) сказуемого находится за группой (сегментом) подлежащего;
5) синтаксической формой предложения может быть одна из рассмотренных 556 СФ.

Первые четыре требования обязательны для предложений русского языка. Пятое требование реализовано в алгоритме прикрепления словоформ к вершинам дерева СФ (структура заранее известна): если в заданной последовательности имеется слово, соответствующее данной вершине, то она включается в искомое дерево СФ, иначе её не будет в СФ.

В ходе анализа словоформ из примера уточняются их группы:

1) **легкий** — **легкий** 022131;
2) **пол** — **пол** 012131;
3) **белым** — **белым** 022135;
4) **с ножом** — **ножом** 01113516;
5) **лежит** — **лежит** 0721;
6) **у двора** — **дворе** 01213214.

В результате построения графа получится дерево с множеством вершин Vrt и дуг Arc:

$$Vrt = \{022131, 012131, 022135, 01113515, 020721, 022132, 01213214\};$$
$$Arc = \{(012131, 022131), (012131, 01113516), (01113516, 022135),$$
$$(012131, 0721), (0721, 01213214), (01213214, 022132)\}.$$

Данное дерево (Vrt, Arc) представляет собой СФ, порождающее предложение-дерево на рис. 3.

Выводы

На рассмотренном примере показана возможность конструктивной теории грамматики естественного языка, без которой компьютерная лингвистика будет довольствоваться только эвристическими методами индуктивного поиска и перебора предполагаемых решений в самых актуальных задачах лингвистики.

1) Построение (синтез) и распознавание (анализ) морфологических форм слов (словоформ) русского языка на компьютере не нуждается в поиске и теоретическом обосновании соответствующих алгоритмов. Они известны и формализованы [2, 4].
2) Форма слова задаётся и однозначно определяется исходной формой этого слова и его морфологическими параметрами, которые могут быть кодированы двухразрядными числами. Словоформы распределены по лексическим группам; лексическая группа — множество, состоящее из словоформ с одинаковым набором морфологических параметров; лексическая группа имеет такой же код, что и её элементы.
3) Основу синтаксиса составляет конечное множество словосочетаний; словосочетание — связанная пара словоформ; признаком связности являются отношения между морфологическими параметрами словоформ. Связность двух словоформ не зависит от их значений.
4) Из п. 2 следует, что понятие связности двух словоформ механически (по способу их кодирования) переносится на лексические группы.
5) Если две лексические группы образуют связанную пару, то их прямое произведение состоит из связанных словоформ (словосочетаний). Связанных лексических

групп в грамматике русского языка имеется около двухсот [5, 6]. В нашем примере были рассмотрены только пять из них, которые были разбиты на 34 группы для уточнения и удобства представления словосочетаний. Мы их назвали СО.

6) Два множества — одно состоит из словоформ, другое — из словосочетаний — образуют структуру данных, известную в математике как ориентированный граф. Обозначим его $G1$.
7) Введённое в п. 4 понятие связности двух словоформ задает ещё один граф (обозначим его через $G2$); вершинами $G2$ являются лексические группы; две вершины связаны дугой, если они — связанные группы. Граф $G2$ представляет редукцию графа $G1$, размеры (число вершин и дуг) которого в тысячи раз больше размеров $G2$ (если в $G1$ включить всю лексику русского языка).
8) Все понятия синтаксиса (подлежащее, сказуемое, однородные члены, дополнения и т. д.) определяются в терминах теории графов. В частности, предложением является любое корневое дерево в $G1$, содержащее дугу, соединяющую подлежащее (корень дерева) со сказуемым [4, 6]. Поэтому граф $G1$, образованный из словоформ и словосочетаний, назван синтаксисом $G1$.
9) Понятие предложения естественным образом переносится из графа $G1$ в граф $G2$: корневое дерево в $G2$, содержащее дугу, соединяющую лексическую группу из подлежащих (корень дерева) с лексической группой из сказуемых, называется синтаксической формой (предложения). Синтаксическая форма представляет собой правило, порождающее определенный класс предложений.
10) По заданной синтаксической форме легко строятся соответствующие ей предложения. Этот процесс называется синтезом предложений.
11) Для заданного предложения нетрудно найти порождающую его синтаксическую форму. Этот процесс называется анализом предложения.
12) Процедура нахождения синтаксической формы из п. 11 необходима для экспериментального обоснования корректности описанной грамматики: предложения русского языка (из литературных источников) являются предложениями в формальной грамматике.
13) Процедура нахождения синтаксической формы может быть использована для экспериментального изучения статистики разнообразия синтаксических форм, которыми пользуются различные авторы — их не более двух десятков в арсенале известных писателей.
14) В дальнейшем при построении конструктивной теории языка фрагменты семантики инкапсулируются в синтаксические формы, из которых состоит семантический словарь языка.
15) Если носители русского языка пользуются двумя десятками синтаксических форм и программы, анализирующие предложения, порождённые этими формами, невелики, то имеет смысл разработать соответствующие программы (синтаксические примитивы, по аналогии с примитивами в графических редакторах) для реализации на компьютере грамматики русского языка, а далее и языка в целом.

ЛИТЕРАТУРА

1. *Мельчук И. А.* Опыт теории лингвистических моделей «СМЫСЛ — ТЕКСТ». М.: Школа «Языки русской культуры», 1999. 346 с.
2. *Зализняк А. А.* Грамматический словарь русского языка. М.: Русский язык, 1977. 880 с.
3. *Харари Ф.* Теория графов. М.: Мир, 1973. 300 с.
4. *Тестелец Я. Г.* Введение в общий синтаксис. М.: РГГУ, 2001. 796 с.

5. *Виноградов В. В.* Грамматика русского языка. В 2 т. М.: АН СССР, 1960. 1864 с.
6. *Шихиев Ф. Ш.* Формализация и сетевая формулировка задачи синтаксического анализа: дис. ... канд. физ.-мат. наук. СПб.: СПбГУ, 2006. 171 с.

REFERENCES

1. *Mel'chuk I. A.* Opyt teorii lingvisticheskikh modeley «SMYSL — TEKST» [The Experience of the Theory of Linguistic Models "SENSE — TEXT"]. Moscow, Shkola "Yazyki russkoy kul'tury", 1999. 346 p. (in Russian)
2. *Zaliznyak A. A.* Grammaticheskiy slovar' russkogo yazyka [Russian Grammar Dictionary]. Moscow, Russkiy Yazyk Rubl., 1977. 880 p. (in Russian)
3. *Harary F.* Graph Theory. Addison-Wesley Pub. Co., 1969. 274 p.
4. *Testelets Ya. G.* Vvedenie v obshchiy sintaksis [Introduction to the Common Syntax]. Moscow, RSUH Publ., 2001. 796 p. (in Russian)
5. *Vinogradov V. V.* Grammatika russkogo yazyka [Russian Grammar]. Moscow, USSR AS Publ., 1960. 1864 p. (in Russian)
6. *Shikhiev F. Sh.* Formalizatsiya i setevaya formulirovka zadachi sintaksicheskogo analiza [Formalization and Network Formulation of the Task of Parsing]. PhD Thesis, St. Petersburg, SPBU, 2006. 171 p. (in Russian)

СВЕДЕНИЯ ОБ АВТОРАХ

АГИБАЛОВ Геннадий Петрович — доктор технических наук, профессор, профессор кафедры защиты информации и криптографии Национального исследовательского Томского государственного университета, г. Томск. E-mail: **agibalov@isc.tsu.ru**

АГИЕВИЧ Сергей Валерьевич — кандидат физико-математических наук, заведующий лабораторией НИИ прикладных проблем математики и информатики БГУ, г. Минск. E-mail: **agievich@bsu.by**

ВОЛОДИЧЕВА Маргарита Ивановна — кандидат физико-математических наук, доцент, доцент кафедры математики Санкт-Петербургского государственного морского технического университета, г. Санкт-Петербург. E-mail: **mvolodicheva@mail.ru**

ГАМАЮНОВ Денис Юрьевич — кандидат физико-математических наук, старший научный сотрудник факультета компьютерных наук НИУ ВШЭ, г. Москва.
E-mail: **gamajun@seclab.cs.msu.su**

ГОРОДИЛОВА Анастасия Александровна — кандидат физико-математических наук, научный сотрудник Института математики им. С. Л. Соболева СО РАН, старший преподаватель Новосибирского государственного университета, г. Новосибирск. E-mail: **gorodilova@math.nsc.ru**

ИДРИСОВА Валерия Александровна — аспирантка Института математики им. С. Л. Соболева СО РАН, инженер лаборатории алгоритмики Новосибирского государственного университета, г. Новосибирск. E-mail: **vvitkup@yandex.ru**

КОЛОМЕЕЦ Николай Александрович — кандидат физико-математических наук, научный сотрудник Института математики им. С. Л. Соболева СО РАН, ассистент Новосибирского государственного университета, г. Новосибирск.
E-mail: **nkolomeec@gmail.com**

КУЦЕНКО Александр Владимирович — магистрант Новосибирского государственного университета, инженер лаборатории алгоритмики Новосибирского государственного университета, г. Новосибирск. E-mail: **AlexandrKutsenko@bk.ru**

ЛЕОРА Светлана Николаевна — кандидат физико-математических наук, доцент, доцент кафедры общей математики и информатики Санкт-Петербургского государственного университета, г. Санкт-Петербург. E-mail: **leora2008@mail.ru**

МИРЗАБЕКОВ Яхья Мирзабекович — старший преподаватель кафедры дискретной математики и информатики Дагестанского государственного университета, г. Махачкала. E-mail: **yash831@mail.ru**

МОНАРЕВ Виктор Александрович — кандидат физико-математических наук, научный сотрудник Института вычислительных технологий СО РАН, г. Новосибирск. E-mail: **viktor.monarev@gmail.com**

ОБЛАУХОВ Алексей Константинович — магистрант Новосибирского государственного университета, инженер лаборатории алгоритмики Новосибирского государственного университета, г. Новосибирск. E-mail: **oblaukhov@gmail.com**

ПАНКРАТОВА Ирина Анатольевна — кандидат физико-математических наук, доцент, доцент кафедры защиты информации и криптографии Национального исследовательского Томского государственного университета, г. Томск.
E-mail: **pank@isc.tsu.ru**

ПЕСТУНОВ Андрей Игоревич — кандидат физико-математических наук, доцент кафедры информационной безопасности Новосибирского государственного университета экономики и управления, г. Новосибирск. E-mail: **pestunov@gmail.com**

РЫБАЛОВ Александр Николаевич — кандидат физико-математических наук, научный сотрудник научно-исследовательской лаборатории «Информационная безопасность» кафедры комплексной защиты информации Омского государственного технического университета, г. Омск. E-mail: **alexander.rybalov@gmail.com**

СОШИН Данил Андреевич — сотрудник ФГУП «НИИ «Квант», г. Москва.
E-mail: **danil_re@list.ru**

ТАРКОВ Михаил Сергеевич — кандидат технических наук, доцент, старший научный сотрудник Института физики полупроводников им. А. В. Ржанова СО РАН, г. Новосибирск. E-mail: **tarkov@isp.nsc.ru**

ТОКАРЕВА Наталья Николаевна — кандидат физико-математических наук, старший научный сотрудник Института математики им. С. Л. Соболева СО РАН, доцент Новосибирского государственного университета, г. Новосибирск.
E-mail: **tokareva@math.nsc.ru**

ЧЕРЕМУШКИН Александр Васильевич — доктор физико-математических наук, научный консультант ФГУП «НИИ «Квант», г. Москва. E-mail: **avc238@mail.ru**

ШЕЙДАЕВ Вадим Физулиевич — студент факультета ВМК МГУ имени М. В. Ломоносова, г. Москва. E-mail: **enr1g@seclab.cs.msu.su**

ШИХИЕВ Шукур Бабаевич — кандидат физико-математических наук, доцент, доцент кафедры дискретной математики и информатики Дагестанского государственного университета, г. Махачкала. E-mail: **sh_sh_b51@mail.ru**

ШУШУЕВ Георгий Иннокентьевич — аспирант Института математики им. С. Л. Соболева СО РАН, г. Новосибирск. E-mail: **shyshyev@gmail.com**