МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

ПРИКЛАДНАЯ ДИСКРЕТНАЯ МАТЕМАТИКА

Научный журнал

2021 № 53

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

РЕДАКЦИОННАЯ КОЛЛЕГИЯ ЖУРНАЛА
«ПРИКЛАДНАЯ ДИСКРЕТНАЯ МАТЕМАТИКА»

Черемушкин А. В., д-р физ.-мат. наук, чл.-корр. Академии криптографии РФ (главный редактор); Девянин П. Н., д-р техн. наук, чл.-корр. Академии криптографии РФ (зам. гл. редактора); Панкратова И. А., канд. физ.-мат. наук, доц. (отв. секретарь); Агиевич С. В., канд. физ.-мат. наук; Алексеев В. Б., д-р физ.-мат. наук, проф.; Евдокимов А. А., канд. физ.-мат. наук, проф.; Колесникова С. И., д-р техн. наук; Крылов П. А., д-р физ.-мат. наук, проф.; Логачев О. А., д-р физ.-мат. наук, доц.; Мясников А. Г., д-р физ.-мат. наук, проф.; Романьков В.А., д-р физ.-мат. наук, проф.; Сафонов К. В., д-р физ.-мат. наук, проф.; Фомичев В. М., д-р физ.-мат. наук, проф.; Харин Ю. С., д-р физ.-мат. наук, чл.-корр. НАН Беларуси; Чеботарев А. Н., д-р техн. наук, проф.; Шоломов Л. А., д-р физ.-мат. наук, проф.

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

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

Периодичность выхода журнала: 4 номера в год.

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

Подписано к печати 20.09.2021. Формат $60 \times 84\frac{1}{8}$. Усл. п. л. 14,8. Тираж 300 экз.
Заказ № 4789. Цена свободная. Дата выхода в свет 29.09.2021.

Отпечатано на оборудовании
Издательства 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

UDC 512.53 DOI 10.17223/20710410/53/1

EQUATIONS OVER DIRECT POWERS OF ALGEBRAIC STRUCTURES IN RELATIONAL LANGUAGES[1]

A. Shevlyakov

Sobolev Institute of Mathematics SB RAS, Omsk, Russian Federation
Omsk State Technical University, Omsk, Russian Federation

E-mail: art.shevlyakov@gmail.com

For a semigroup S (group G) we study relational equations and describe all semigroups S with equationally Noetherian direct powers. It follows that any group G has equationally Noetherian direct powers if we consider G as an algebraic structure of a certain relational language. Further we specify the results as follows: if a direct power of a finite semigroup S is equationally Noetherian, then the minimal ideal $\mathrm{Ker}(S)$ of S is a rectangular band of groups and $\mathrm{Ker}(S)$ coincides with the set of all reducible elements.

Keywords: *relations, groups, semigroups, direct powers, equationally Noetherian algebraic structures.*

Introduction

Let $\mathcal{A}$ be an algebraic structure of a functional language $\mathcal{L}$ with a universe A. In other words, there are certain functions and constants over $\mathcal{A}$ that correspond to symbols of $\mathcal{L}$. One can define a structure $\mathrm{Pr}(\mathcal{A})$ with the universe A of a pure relational language $\mathcal{L}_{\mathrm{pred}}$ as follows:

$$R_f(x_1,\dots,x_n,y) = \{(x_1,\dots,x_n,y) : f(x_1,\dots,x_n) = y \in \mathcal{A}\};$$
$$R_c(x) = \{x : x = c \in \mathcal{A}\},$$

where functional and constant symbols f, c belong to the language $\mathcal{L}$. Namely, the relation $R_f \in \mathcal{L}_{\mathrm{pred}}$ ($R_c \in \mathcal{L}_{\mathrm{pred}}$) is the graph of a function f (respectively, constant c).

The $\mathcal{L}_{\mathrm{pred}}$-structure $\mathrm{Pr}(\mathcal{A})$ is called the *predicatization* of an $\mathcal{L}$-structure $\mathcal{A}$. In particular, if $\mathcal{A}$ is a group of the language $\mathcal{L}_g = \{\cdot, ^{-1}, 1\}$, then $\mathrm{Pr}(\mathcal{A})$ is an algebraic structure of the language $\mathcal{L}_{g-\mathrm{pred}}$ with the following relations:

$$M(x,y,z) \Leftrightarrow xy = z; \tag{1}$$
$$I(x,y) \Leftrightarrow x = y^{-1}; \tag{2}$$
$$E(x) \Leftrightarrow x = 1. \tag{3}$$

[1]The author was supported by the RSF-grant 18-71-10028 (Theorem 1) and RSF-grant 19-11-00209 (Theorem 4).

Notice that any equation over a group $\mathcal{A}$ may be rewritten in the language $\mathcal{L}_{g-\mathrm{pred}}$ by the introducing new variables. For example, the equation $x^{-1}y^{-1}xy = 1$ has the following correspondence in the relational language $\mathcal{L}_{g-\mathrm{pred}}$:

$$\mathrm{Pr}(\mathbf{S}) = \begin{cases} I(x, x_1), \\ I(y, y_1), \\ M(x_1, y_1, z_1), \\ M(z_1, x, z_2), \\ M(z_2, y, z_3), \\ E(z_3). \end{cases}$$

It is easy to see that the projection of the solution set of $\mathbf{S}$ onto the variables x, y gives the solution set of the initial equation $x^{-1}y^{-1}xy = 1$. More generally, for any finite set of group equations $\mathbf{S}$ in variables X there exists a system $\mathrm{Pr}(\mathbf{S})$ of equations in the language $\mathcal{L}_{g-\mathrm{pred}}$ such that the solution set of $\mathbf{S}$ is the projection of the solution set $\mathrm{Pr}(\mathbf{S})$ onto the variables X. Hence, there arises the following important problem.

Problem. What properties of a finite system $\mathbf{S}$ are determined by the system $\mathrm{Pr}(\mathbf{S})$?

This problem was originally studied in [1], where it was proved the general results for relational structures $\mathrm{Pr}(\mathbf{S})$.

We study equations over direct products of semigroups. Namely, for a finite semigroup S we give necessary and sufficient condition whether the direct power $\Pi\,\mathrm{Pr}(S)$ is equationally Noetherian. It continues the research [2], where we found the necessary and sufficient conditions for the equationally Noetherian property of direct powers of functional algebraic structures (groups, rings, monoids). For example, a group (ring) has equationally Noetherian direct powers in a functional language with constants iff it is abelian (respectively, with zero multiplication).

On the other hand, we prove below that any finite group in the language $\mathcal{L}_{g-\mathrm{pred}}$ has equationally Noetherian direct powers (Corollary 1). Moreover, the similar result holds for the natural generalizations of groups: quasi-groups and loops (Remark 1).

However, the class of semigroups has a nontrivial classification in the relational language. We find two quasi-identities

$$\forall a \forall b \forall \alpha \forall \beta \left((a\alpha = a\beta) \to (b\alpha = b\beta) \right); \tag{4}$$

$$\forall a \forall b \forall \alpha \forall \beta \left((\alpha a = \beta a) \to (\alpha b = \beta b) \right) \tag{5}$$

such that a finite semigroup S satisfies (4), (5) iff any direct power of $\mathrm{Pr}(S)$ is equationally Noetherian (Theorem 1).

In the class of finite semigroups the conditions (4), (5) imply that the minimal ideal (kernel) of a semigroup S is a rectangular band of groups, and the kernel $\mathrm{Ker}(S)$ (the minimal ideal of S) coincides with the ideal of reducible elements of S. If the kernel of a finite semigroup S is a group, then the converse statement also holds (Theorem 4). However, the converse statement is not true in general (Example 1).

1. Basic notions

An algebraic structure of the language $\mathcal{L}_{s-\mathrm{pred}} = \{M^{(3)}\}$ ($\mathcal{L}_{g-\mathrm{pred}} = \{M^{(3)}, I^{(2)}, E^{(1)}\}$) is called the *predicatization* of a semigroup S (group G) if the operations over S (G) corresponds to the relations (1)–(3). The predicatization of a semigroup S (group G) is denoted by $\mathrm{Pr}(S)$ (respectively, $\mathrm{Pr}(G)$).

Following [3], we give the main definitions of algebraic geometry over algebraic structures (below $\mathcal{L} \in \{\mathcal{L}_{s-\mathrm{pred}}, \mathcal{L}_{p-\mathrm{pred}}\}$).

An *equation over $\mathcal{L}$ ($\mathcal{L}$-equation)* is an atomic formula over $\mathcal{L}$. The examples of equations are the following: $M(x,x,x)$, $M(x,y,x)$ ($\mathcal{L}_{s-\mathrm{pred}}$-equations); $M(x,x,y)$, $I(x,y)$, $I(x,x)$, $E(x)$ ($\mathcal{L}_{g-\mathrm{pred}}$-equations).

A *system of $\mathcal{L}$-equations* ($\mathcal{L}$-*system* for shortness) is an arbitrary set of $\mathcal{L}$-equations. Notice that we will consider only systems in a finite set of variables $X = \{x_1, x_2, \ldots, x_n\}$. The set of all solutions of $\mathbf{S}$ in an $\mathcal{L}$-structure $\mathcal{A}$ is denoted by $\mathrm{V}_{\mathcal{A}}(\mathbf{S}) \subseteq \mathcal{A}^n$. A set $Y \subseteq \mathcal{A}^n$ is said to be an *algebraic set over* $\mathcal{A}$ if there exists an $\mathcal{L}$-system $\mathbf{S}$ with $Y = \mathrm{V}_{\mathcal{A}}(\mathbf{S})$. If the solution set of an $\mathcal{L}$-system $\mathbf{S}$ is empty, $\mathbf{S}$ is said to be *inconsistent*. Two $\mathcal{L}$-systems $\mathbf{S}_1, \mathbf{S}_2$ are called *equivalent over an $\mathcal{L}$-structure $\mathcal{A}$* if $\mathrm{V}_{\mathcal{A}}(\mathbf{S}_1) = \mathrm{V}_{\mathcal{A}}(\mathbf{S}_2)$.

An $\mathcal{L}$-structure $\mathcal{A}$ is *$\mathcal{L}$-equationally Noetherian* if any infinite $\mathcal{L}$-system $\mathbf{S}$ is equivalent over $\mathcal{A}$ to a finite subsystem $\mathbf{S}' \subseteq \mathbf{S}$.

Let $\mathcal{A}$ be an $\mathcal{L}$-structure. By $\mathcal{L}(\mathcal{A})$ we denote the language $\mathcal{L} \cup \{a : a \in \mathcal{A}\}$ extended by new constants symbols which correspond to elements of $\mathcal{A}$. The language extension allows us to use constants in equations. The examples of equations in the extended languages are the following: $M(x,y,a)$ ($\mathcal{L}_{s-\mathrm{pred}}(S)$-equation and $a \in S$); $M(a,x,b)$, $I(x,a)$, $E(a)$ ($\mathcal{L}_{g-\mathrm{pred}}(S)$-equations and $a, b \in G$). Obviously, the class of $\mathcal{L}(\mathcal{A})$-equations is wider than the class of $\mathcal{L}$-equations, so an $\mathcal{L}$-equationally Noetherian algebraic structure $\mathcal{A}$ may lose this property in the language $\mathcal{L}(\mathcal{A})$.

One can directly prove that any finite $\mathcal{L}(\mathcal{A})$-structure $\mathcal{A}$ is always $\mathcal{L}(\mathcal{A})$-equationally Noetherian.

Since the algebraic structures $\mathcal{A}$ and $\mathrm{Pr}(\mathcal{A})$ have the same universe, we will write below $\mathrm{V}_{\mathcal{A}}(\mathbf{S})$ ($\mathcal{L}(\mathcal{A})$) instead of $\mathrm{V}_{\mathrm{Pr}(\mathcal{A})}(\mathbf{S})$ (respectively, $\mathcal{L}(\mathrm{Pr}(\mathcal{A}))$).

Let $\mathcal{A}$ be a relational $\mathcal{L}$-structure. The *direct power* $\Pi\mathcal{A} = \prod\limits_{i \in I} \mathcal{A}$ of $\mathcal{A}$ is the set of all sequences $[a_i : i \in I]$ and any relation $R \in \mathcal{L}$ is defined as follows

$$R\big([a_i^{(1)} : i \in I], [a_i^{(2)} : i \in I], \ldots, [a_i^{(n)} : i \in I]\big) \Leftrightarrow R\big(a_i^{(1)}, a_i^{(2)}, \ldots, a_i^{(n)}\big) \text{ for each } i \in I.$$

A map $\pi_k \colon \Pi\mathcal{A} \to \mathcal{A}$ is called the *projection onto the i-th coordinate* if $\pi_k([a_i : i \in I]) = a_k$.

Let $E(X)$ be an $\mathcal{L}(\Pi\mathcal{A})$-equation over a direct power $\Pi\mathcal{A}$. We may rewrite $E(X)$ in the form $E(X, \overrightarrow{\mathbf{C}})$, where $\overrightarrow{\mathbf{C}}$ is an array of constants occurring in the equation $E(X)$. One can introduce the *projection of an equation* onto the i-th coordinate as follows:

$$\pi_i(E(X)) = \pi_i(E(X, \overrightarrow{\mathbf{C}})) = E(X, \pi_i(\overrightarrow{\mathbf{C}})),$$

where $\pi_i(\overrightarrow{\mathbf{C}})$ is an array of the i-th coordinates of the elements from $\overrightarrow{\mathbf{C}}$. For example, the $\mathcal{L}_{s-\mathrm{pred}}(\Pi\mathcal{A})$-equation $M(x, [a_1, a_2, a_3, \ldots], [b_1, b_2, b_3, \ldots])$ has the following projections

$$\begin{gathered} M(x, a_1, b_1), \\ M(x, a_2, b_2), \\ M(x, a_3, b_3), \\ \ldots \end{gathered}$$

Obviously, any projection of an $\mathcal{L}(\Pi\mathcal{A})$-equation is an $\mathcal{L}(\mathcal{A})$-equation.

Let us take an $\mathcal{L}(\Pi\mathcal{A})$-system $\mathbf{S} = \{E_j(X) : j \in J\}$. The i-th projection of $\mathbf{S}$ is the $\mathcal{L}(\mathcal{A})$-system defined by $\pi_i(\mathbf{S}) = \{\pi_i(E_j(X)) : j \in J\}$. The projections of an $\mathcal{L}(\Pi\mathcal{A})$-system $\mathbf{S}$ allow to describe the solution set of $\mathbf{S}$ by

$$\mathrm{V}_{\Pi\mathcal{A}}(\mathbf{S}) = \{[P_i : i \in I] : P_i \in \mathrm{V}_{\mathcal{A}}(\pi_i(\mathbf{S}))\}. \tag{6}$$

In particular, if one of the projections $\pi_i(\mathbf{S})$ is inconsistent, so is $\mathbf{S}$.

The following statement immediately follows from the description (6) of the solution set over a direct powers.

Lemma 1. Let $\mathbf{S} = \{E_j(X) : j \in J\}$ be an $\mathcal{L}(\Pi\mathcal{A})$-system over $\Pi\mathcal{A}$. If one of the projections $\pi_i(\mathbf{S})$ is inconsistent, so is $\mathbf{S}$. Moreover, if $\mathcal{A}$ is $\mathcal{L}(\mathcal{A})$-equationally Noetherian, then an inconsistent $\mathcal{L}(\Pi\mathcal{A})$-system $\mathbf{S}$ is equivalent to a finite subsystem.

Proof. The first assertion directly follows from (6). Suppose $\mathcal{A}$ is $\mathcal{L}$-equationally Noetherian, and $\pi_i(\mathbf{S})$ is inconsistent. Hence, $\pi_i(\mathbf{S})$ is equivalent to its finite inconsistent subsystem $\{\pi_i(E_j(X)) : j \in J'\}$, $|J'| < \infty$, and the finite subsystem $\mathbf{S}' = \{E_j(X) : j \in \in J'\} \subseteq \mathbf{S}$ is also inconsistent. ■

2. Predicatization of semigroups and groups

Theorem 1. Let $\Pr(S)$ be the predicatization of a finite semigroup S. A direct power of $\Pr(S)$ is $\mathcal{L}_{s-\mathrm{pred}}(\Pi S)$-equationally Noetherian iff the quasi-identities (4), (5) hold in S.

Proof. First, we prove the "if" part of the theorem. Suppose S satisfies (4,5) and consider an infinite $\mathcal{L}_{s-\mathrm{pred}}(\Pi S)$-system $\mathbf{S}$. One can represent $\mathbf{S}$ as a finite union of the following systems

$$\mathbf{S} = \bigcup_{1\leqslant i,j\leqslant n} \mathbf{S}_{cij} \bigcup_{1\leqslant i,j\leqslant n} \mathbf{S}_{icj} \bigcup_{1\leqslant i,j\leqslant n} \mathbf{S}_{ijc} \bigcup_{1\leqslant i\leqslant n} \mathbf{S}_{cci} \bigcup_{1\leqslant i\leqslant n} \mathbf{S}_{cic} \bigcup_{1\leqslant i\leqslant n} \mathbf{S}_{icc} \bigcup \mathbf{S}_0, \tag{7}$$

where each equation of $\mathbf{S}_0$ is one of the following types:

1) $x_i = x_j$;
2) $x_i = \mathbf{c}_j$;
3) $\mathbf{c}_i = \mathbf{c}_j$;
4) $M(x_i, x_j, x_k)$;
5) $M(\mathbf{c}_i, \mathbf{c}_j, \mathbf{c}_k)$,

and $\mathbf{S}_{cij} = \{M(\mathbf{c}_k, x_i, x_j) : k \in K\}$, $\mathbf{S}_{icj} = \{M(x_i, \mathbf{c}_k, x_j) : k \in K\}$, $\mathbf{S}_{ijc} = \{M(x_i, x_j, \mathbf{c}_k) : k \in K\}$, $\mathbf{S}_{cci} = \{M(\mathbf{c}_k, \mathbf{d}_k, x_i) : k \in K\}$, $\mathbf{S}_{cic} = \{M(\mathbf{c}_k, x_i, \mathbf{d}_k) : k \in K\}$, $\mathbf{S}_{icc} = = \{M(x_i, \mathbf{c}_k, \mathbf{d}_k) : k \in K\}$ $(\mathbf{c}_k, \mathbf{d}_k \in \Pi \Pr(S))$, where each system above has its own index set K.

Clearly, the system $\mathbf{S}_0$ is equivalent to its finite subsystem. So it is sufficient to prove that the other systems are equivalent to their finite subsystems. According to Lemma 1, we may assume that all systems below are consistent.

Thus, we have the following cases:

1) Let $\mathbf{S}_{icc} = \{M(x_i, \mathbf{c}_k, \mathbf{d}_k) : k \in K\}$ and $M(x_i, \mathbf{c}_1, \mathbf{d}_1)$ be an arbitrary equation of $\mathbf{S}_{icc}$. Since $\mathbf{S}_{icc}$ is consistent, then one can choose $\bar{\alpha} \in \mathrm{V}_{\Pi S}(\mathbf{S}_{icc})$, $\bar{\beta} \in \mathrm{V}_{\Pi Ss}(M(x_i, \mathbf{c}_1, \mathbf{d}_1))$. We have $\bar{\alpha}\mathbf{c}_1 = \bar{\beta}\mathbf{c}_1 = \mathbf{d}_1$. By the quasi-identities (4), (5), $\bar{\alpha}\mathbf{c}_k = \bar{\beta}\mathbf{c}_k$ for any $\mathbf{c}_k$. Hence, $\bar{\beta}$ satisfies all equations from $\mathbf{S}_{icc}$, Thus, $\mathbf{S}_{icc}$ is equivalent to the equation $M(x_i, \mathbf{c}_1, \mathbf{d}_1)$. The proof for the system $\mathbf{S}_{cic}$ is similar.
2) Let $\mathbf{S}_{cci} = \{M(\mathbf{c}_k, \mathbf{d}_k, x_i) : k \in K\}$. Since the system $\mathbf{S}_{cci}$ is consistent, the products $\mathbf{c}_k\mathbf{d}_k$ are equal to each other, hence $\mathbf{c} = \mathbf{c}_k\mathbf{d}_k$ for all $k \in K$. Thus, the whole system $\mathbf{S}_{cci}$ is equivalent to any equation $M(\mathbf{c}_k, \mathbf{d}_k, x_i)$.
3) Let $\mathbf{S}_{icj} = \{M(x_i, \mathbf{c}_k, x_j) : k \in K\}$ (the proof for $\mathbf{S}_{cij}$ is similar). Since $\mathbf{S}_{icj}$ is consistent, there exist a point $(\bar{\alpha}, \bar{\beta}) \in \mathrm{V}_{\Pi S}(\mathbf{S}_{icj})$ and the equalities $\bar{\alpha}\mathbf{c}_k = \bar{\alpha}\mathbf{c}_l = \bar{\beta}$ hold for any $k, l \in K$. By (4), (5), for any $\bar{\gamma} \in \Pi S$ it holds $\bar{\gamma}\mathbf{c}_k = \bar{\gamma}\mathbf{c}_l$. Thus, the solution set of $\mathbf{S}_{icj}$ is $Y = \{(\bar{\gamma}, \bar{\gamma}\mathbf{c}_{k_0}) \mid \bar{\gamma} \in \Pi S\}$ for a fixed $k_0 \in K$. Thus, $\mathbf{S}_{icj}$ is equivalent to the equation $M(x_i, \mathbf{c}_{k_0}, x_j)$.

4) Let $\mathbf{S}_{ijc} = \{M(x_i, x_j, \mathbf{c}_k) : k \in K\}$. Since the system $\mathbf{S}_{ijc}$ is consistent, the elements $\mathbf{c}_k$ $(k \in K)$ are equal to each other. Hence, the system $\mathbf{S}_{ijc}$ consists of the same equations. Thus, $\mathbf{S}_{ijc}$ is equivalent to any equation $M(x_i, x_j, \mathbf{c}_k)$.

Now, we prove the "only if" part of the theorem. Suppose the quasi-identity (4) does not hold in S (for the formula (5) the proof is similar). It follows there exist elements a, b, α, β such that $a\alpha = a\beta = c$, $b\alpha \neq b\beta$. Let us consider the system

$$\mathbf{S} = \{M(\mathbf{a}_n, x, \mathbf{c}_n) : n \in \mathbb{N}\},$$

where

$$\mathbf{a}_n = [\underbrace{b, \ldots, b}_{n \text{ times}}, a, a, \ldots], \quad \mathbf{c}_n = [\underbrace{b\beta, \ldots, b\beta}_{n \text{ times}}, c, c, \ldots].$$

One can directly check that the point

$$\mathbf{a} = [\underbrace{\beta, \ldots, \beta}_{n \text{ times}}, \alpha, \alpha, \ldots]$$

satisfies the first n equations of $\mathbf{S}$ (since we obtain the true equalities $a\beta = c$ or $b\beta = b\beta$). However the $(n+1)$-th equation of $\mathbf{S}$ gives $\mathbf{a}_{n+1}\mathbf{a} \neq \mathbf{c}_{n+1}$, since its $(n+1)$-th projection defines the equation $bx = b\beta$, but $b\alpha \neq b\beta$. Thus, $\mathbf{S}$ is not equivalent to any finite subsystem. ■

Corollary 1. Let $\mathrm{Pr}(G)$ be the predicatization of a finite group G. Then any direct power of $\mathrm{Pr}(G)$ is $\mathcal{L}_{g-\mathrm{pred}}(\Pi G)$-equationally Noetherian.

Proof. Since the equality $a\alpha = a\beta$ $(\alpha a = \beta a)$ implies $\alpha = \beta$ in any group, the quasi-identities (4), (5) obviously hold in G. Thus, any infinite system of the form $\{M(*, *, *) : i \in I\}$ is equivalent to a finite subsystem.

One can directly prove that for any finite group G the infinite systems of the form $\{I(*, *) : i \in I\}$ $(\{E(*) : i \in I\})$ are also equivalent to their finite subsystems over ΠG.

Thus, any system of $\mathcal{L}_{g-\mathrm{pred}}(\Pi G)$-equations is equivalent over ΠG to its finite subsystem. ■

Remark 1. The Corollary 1 also holds for finite quasi-groups. Notice that a quasi-group is a non-associative generalization of a group. Any quasi-group admits the analogue of divisibility, hence the quasi-identities (4), (5) obviously hold in any quasi-group. Thus, any direct power of a quasi-group G is $\mathcal{L}_{s-\mathrm{pred}}(\Pi G)$-equationally Noetherian (here we consider quasi-groups and loops in the language $\mathcal{L}_{s-\mathrm{pred}}$, since not any quasi-group admits the relations $I(x, y)$ and $E(x)$).

Below we study finite semigroups S that satisfy Theorem 1.

A subset $I \subseteq S$ is called a *left* (*right*) *ideal* if for any $s \in S$, $a \in I$ it holds $sa \in I$ $(as \in I)$. An ideal which is right and left simultaneously is said to be *two-sided* (or an *ideal* for shortness).

A semigroup S with a unique ideal $I = S$ is called *simple*. Let us remind the classical Sushkevich — Rees theorem for finite simple semigroups.

Theorem 2. For any finite simple semigroup S there exist a finite group G and finite sets I, Λ such that S is isomorphic to the set of triples (λ, g, i), $g \in G$, $\lambda \in \Lambda$, $i \in I$. The multiplication over the triples (λ, g, i) is defined by

$$(\lambda, g, i)(\mu, h, j) = (\lambda, gp_{i\mu}h, j),$$

where $p_{i\mu} \in G$ is an element of a matrix $\mathbf{P}$ such that

1) $\mathbf{P}$ consists of $|I|$ rows and $|\Lambda|$ columns;
2) the elements of the first row and the first column equal $1 \in G$ (i.e., $\mathbf{P}$ is *normalized*).

Following Theorem 2, we denote any finite simple semigroup S by $S = (G, \mathbf{P}, \Lambda, I)$.

The minimal ideal of a semigroup S is called a *kernel* and denoted by $\mathrm{Ker}(S)$ (any finite semigroup always has a unique kernel, and the kernel is always simple, i.e., $\mathrm{Ker}(S)$ satisfies Theorem 2). Obviously, if $S = \mathrm{Ker}(S)$, then the semigroup is simple. If $\mathrm{Ker}(S)$ is a group, then S is said to be a *homogroup*. The next theorem contains the necessary information about homogroups.

Theorem 3 [4]. In a homogroup S the identity element e of the kernel $\mathrm{Ker}(S)$ is idempotent ($e^2 = e$) and belongs to the center of S (i.e., e commutes with any $s \in S$).

A semigroup S is called a *rectangular band of groups* if $S = (G, \mathbf{P}, \Lambda, I)$ and $p_{i\lambda} = 1$ for any $i \in I$, $\lambda \in \Lambda$.

Lemma 2. Suppose a finite simple semigroup S satisfies (4), (5). Then S is a rectangular band of groups.

Proof. By Theorem 2, $S = (G, \mathbf{P}, \Lambda, I)$ for some finite group G, matrix $\mathbf{P}$ and finite sets of indexes Λ, I.

Assume that $|\Lambda| > 1$ and $p_{i\lambda} \neq 1$ for some i, λ.

Let $a = (1,1,1)$, $\alpha = (\lambda,1,1)$, $\beta = (1,1,1)$ and hence

$$a\alpha = (1,1,1)(\lambda,1,1) = (1,1,1) = (1,1,1)(1,1,1) = a\beta. \tag{8}$$

However, for $b = (1,1,i)$ we have

$$b\alpha = (1,1,i)(\lambda,1,1) = (1,p_{i\lambda},1) \neq (1,1,1) = (1,1,i)(1,1,1) = b\beta. \tag{9}$$

Thus, the equalities (8), (9) contradict (4), (5). ■

An element s of a semigroup S is called *reducible* if there exist $a, b \in S$ with $s = ab$. Clearly, the set of all reducible elements $\mathrm{Red}(S)$ is an ideal of a semigroup S.

Lemma 3. Let S be a finite semigroup satisfying (4), (5). Then $\mathrm{Ker}(S)$ is the set of all reducible elements.

Proof. Since the kernel $\mathrm{Ker}(S)$ is simple, Theorem 2 gives $\mathrm{Ker}(S) = (G, \mathbf{P}, \Lambda, I)$ for some finite $G, \mathbf{P}, \Lambda, I$. Let $b \in S$. We have $(\lambda, g, i)b = (\lambda, g, i)(1,1,i)b = (\lambda, g, i)r$, where $r = (1,1,i)b \in \mathrm{Ker}(S)$. By (4), we obtain $ab = ar$ for any $a \in S$. Since $ar \in \mathrm{Ker}(S)$, so is ab. Thus, any product of elements belongs to $\mathrm{Ker}(S)$, hence $\mathrm{Red}(S) = \mathrm{Ker}(S)$. ■

Theorem 4. If $\mathrm{Ker}(S) = \mathrm{Red}(S)$ for a finite homogroup S, then S satisfies (4), (5) or, equivalently, ΠS is $\mathcal{L}_{s-\mathrm{pred}}(\Pi S)$-equationally Noetherian.

Proof. Let us take a, b, α, β such that $a\alpha = a\beta$, and e be the identity of $\mathrm{Ker}(S)$. We have

$$\begin{aligned}
a\alpha &= a\beta \mid \cdot e,\\
ea\alpha &= ea\beta,\\
(ea)\alpha &= (ea)\beta \mid \cdot(ea)^{-1} \text{ since } ea \text{ belongs to the group } \mathrm{Ker}(S),\\
e\alpha &= e\beta \mid e \text{ is a central element},\\
\alpha e &= \beta e.
\end{aligned}$$

We have (below we use $b\alpha, b\beta \in \mathrm{Ker}(S) = \mathrm{Red}(S)$):

$$b\alpha = (b\alpha)e = b(\alpha e) = b(\beta e) = (b\beta)e = b\beta.$$

Thus, the quasi-identity (4) holds for S. The proof for the quasi-identity (5) is similar. ■

One can directly check that for a rectangular band of groups $S = (G, \mathbf{P}, \Lambda, I)$ the analogue of Theorem 4 also holds.

Thus, there arises the following question.

Question. Suppose the kernel $\mathrm{Ker}(S)$ of a finite semigroup S satisfies the following conditions:

1) $\mathrm{Ker}(S) = \mathrm{Red}(S)$;
2) $\mathrm{Ker}(S)$ is a rectangular band of groups.

Does S satisfy the quasi-identities (4), (5)?

Example 1. The answer for the last question is negative. Let us consider a semigroup S with the following multiplication table:

$\cdot$	a	b	z_1	z_2	z_3	z_4
a	z_4	z_4	z_2	z_4	z_4	z_4
b	z_4	z_4	z_3	z_4	z_4	z_4
z_1	z_1	z_1	z_1	z_1	z_1	z_1
z_2	z_2	z_2	z_2	z_2	z_2	z_2
z_3	z_3	z_3	z_3	z_3	z_3	z_3
z_4	z_4	z_4	z_4	z_4	z_4	z_4

This Table defines an associative binary operation (we checked it by a computer).

One can directly compute that $\mathrm{Ker}(S) = \mathrm{Red}(S) = \{z_1, z_2, z_3, z_4\}$. Since the elements z_i are left zeros, we have $\mathrm{Ker}(S) = (G, \mathbf{P}, \Lambda, I)$, where $G = \{1\}$, $\mathbf{P} = (1, 1, 1, 1)$, $\Lambda = \{1, 2, 3, 4\}$, $I = \{1\}$. However, the quasi-identity (5) does not hold in S, since $az_1 = bz_1$, but $az_0 \neq bz_0$.

REFERENCES

1. *Shevlyakov A. N.* Algebraic geometry over groups in predicate language. Herald of Omsk University, 2018, vol. 24, no. 4, pp. 60–63.
2. *Shevlyakov A. N. and Shahryari M.* Direct products, varieties, and compactness conditions. Groups Complexity Cryptology, 2017, vol. 9, no. 2, pp. 159–166.
3. *Daniyarova E. Yu., Myasnikov A. G., and Remeslennikov V. N.* Algebraic geometry over algebraic structures, II: Foundations. J. Math. Sci., 2012, vol. 183, pp. 389–416.
4. *Lyapin E. S.* Semigroups. Translations Math. Monographs, Amer. Math. Soc., 1974, vol. 3, 519 p.

МАТЕМАТИЧЕСКИЕ МЕТОДЫ КРИПТОГРАФИИ

УДК 519.23

СПЕКТРАЛЬНЫЙ ВЕРОЯТНОСТНО-СТАТИСТИЧЕСКИЙ АНАЛИЗ МАРКОВСКИХ ШИФРОВ

О. В. Денисов

ООО «Инновационные телекоммуникационные технологии», г. Москва, Россия

Изучаются вероятностные модели блочных шифрсистем, в которых случайные раундовые ключи независимы и одинаково распределены. Они называются марковскими шифрами, если последовательность раундовых разностей образует простую однородную цепь Маркова. Описаны (при условиях доминирования второго собственного значения матрицы P вероятностей переходов разностей и одной координаты соответствующих собственных векторов) элементы и строки матриц P^R вероятностей переходов разностей за R раундов, наиболее удалённые от равновероятных значений при всех достаточно больших R. Предложенный в 2016 г. автором спектральный критерий для проверки гипотез о случайных подстановках применён для построения и расчёта атак различения в модели независимых двублочных текстов и в новой модели независимых полных кодовых книг.

Ключевые слова: *марковские блочные шифры, атака различения, спектр матриц, переходные вероятности разностей, второе доминирующее собственное значение, независимые полные кодовые книги.*

DOI 10.17223/20710410/53/2

SPECTRAL PROBABILISTIC AND STATISTICAL ANALYSIS OF MARKOV CIPHERS

O. V. Denisov

Innovative Telecommunication Technologies, LLC, Moscow, Russia

E-mail: denisovOleg@yandex.ru

Let an Abelian group $(\mathcal{X}, +)$ be the alphabet of R-round Markov block cipher with matrix P of transition probabilities of differentials; matrix size equals $M = |\mathcal{X}'|$, $\mathcal{X}' = \mathcal{X} \setminus \{0\}$. Suppose spectrum of P satisfies the condition $\lambda_1 = 1 > |\lambda_2| > |\lambda_3| \geqslant \geqslant \ldots \geqslant \lambda_M$.

1. Extremal transition probabilities $p_{ab}(R)$ and rows P^R for a large number of rounds. Let P be diagonalizable: $BPC = D = \operatorname{diag}(1, \lambda_2, \ldots, \lambda_M)$, $B = C^{-1}$, and there exist $a, b \in \mathcal{X}'$ such that $|C_{a2}| > |C_{i2}|$, $|B_{2b}| > |B_{jb}|$ for all $i \neq a$, $j \neq b$. Then $\arg\max\limits_{(i,j)\in\mathcal{X}'\times\mathcal{X}'} |p_{ij}(R) - \dfrac{1}{M}| = (a, b)$ and $\arg\max\limits_{i\in\mathcal{X}'} |\mathbf{P}_i^{(R)} - \dfrac{1}{M}\mathbf{1}| = a$ for all sufficiently large R, $p_{ab}(R) - \dfrac{1}{M} \sim C_{a2}B_{2b}\lambda_2^R$ and $|\mathbf{P}_a(R) - \dfrac{1}{M}\mathbf{1}| \sim |C_{a2}|\,|\mathbf{B}_2|\,|\lambda_2|^R$ as $R \to \infty$.

2. Distinguishing attack by independent full codebooks. Let the cipher with alphabet $\mathcal{X} = \mathbb{Z}_2^n$ be Markovian (provided random uniformly distributed set of round keys

$\mathbf{k} \sim U(\mathscr{K}^R)$) with matrix P, $z_i = z_i(\mathbf{k})$ be the result of block $i \in \mathcal{X}$ transformation either by cipher (hypothesis H_2) or random uniformly distributed substitution $z(\mathbf{k})$ (hypothesis H_1). Let (λ_2, u) or (λ_2, v) be left or right eigenpair of P, $|u| = |v| = 1$, $\mu_2(R) = \mathbf{u}P^R v^{\downarrow} \neq 0$, $S(\mathbf{k}) = M \sum\limits_{\{i,j\} \subset \mathcal{X}} u_{j-i} v_{z_j - z_i}$. We prove that mean and variance of statistic $S(\mathbf{k})$ equal 0 and $M^2 \dfrac{M+1}{2(M-2)}$ respectively under hypothesis H_1. If sets $\mathbf{k}(1), \ldots, \mathbf{k}(N_b) \sim U(\mathscr{K}^R)$ are independent, $N_b \to \infty$, then for all $0 < \alpha < 1$ criterion $d:\ S'(N_b)\ \mathrm{sign}(\mu_2(R)) > \varkappa_{1-\alpha} \sqrt{\dfrac{NM}{M-2}} \Longrightarrow H_2$, where $N = \dbinom{M+1}{2} N_b$, has error probability $\alpha_1(d) \to \alpha$. We show that $\alpha_2(d) \approx \beta$ for large values of R and $N_b \approx \dfrac{2(\varkappa_{1-\alpha} + \varkappa_{1-\beta})^2}{(2^n \mu_2(R))^2}$.

Keywords: *Markov block ciphers, distinguishing attack, matrix spectrum, transition probabilities of differentials, second dominant eigenvalue, independent full codebooks.*

Введение

Пусть $\mathcal{X}$ — входной и выходной алфавит блочного шифра, $(\mathcal{X}, +)$ — абелева группа с нейтральным элементом 0. Как принято в дифференциальном (разностном) анализе, величины вида $x^* - x$ будем называть *разностями* блоков $x, x^* \in \mathcal{X}$. Через $\mathcal{X}' = \mathcal{X} \setminus \{0\}$ обозначим множество ненулевых блоков, $M = |\mathcal{X}| - 1$ — его мощность, через $S(\mathcal{X})$ — множество всех подстановок на множестве $\mathcal{X}$.

Рассмотрим итеративный R-раундовый шифр, в котором промежуточные раундовые блоки формируются по правилу

$$x^r = f(x^{r-1}, k^r), \quad 1 \leqslant r \leqslant R,$$

где $x = x^0 \in \mathcal{X}$ — входной блок; k^r — ключ r-го раунда из некоторого множества $\mathscr{K}$ раундовых ключей; f — функция шифрования на одном раунде; $y = x^R \in \mathcal{X}$ — выходной блок.

Рассмотрим также вероятностную модель шифрования, в которой шифруется случайно выбираемый двублочный открытый текст $(X, X^*) \in \mathcal{X}^2$, случайно выбираемые раундовые ключи $K^1, \ldots, K^R \in \mathscr{K}$ не зависят от текста, независимы между собой и одинаково распределены. Заглавные буквы здесь и далее обозначают случайные блоки и ключи.

Тогда последовательность (X^r, X^{*r}), $r = 0, \ldots, R$, пар раундовых блоков является простой однородной цепью Маркова (далее рассматриваем только такие цепи). Но не всякая функция от последовательности состояний цепи Маркова (и, в частности, разность компонент пар) образует цепь Маркова. Поэтому содержательно следующее определение [1 – 3]: введённая модель шифрования называется *марковским шифром* (относительно операции + при заданных распределениях входной пары (X, X^*) и раундового ключа K^1), если последовательность случайных *раундовых разностей*

$$\Delta X = X^* - X,\ \Delta X^1 = X^{*1} - X(1),\ \ldots,\ \Delta Y = \Delta X^R = X^{*R} - X^R \tag{1}$$

образует цепь Маркова.

Известно [1; 3, теорема 1], что матрица P вероятностей переходов такой цепи является дважды стохастической. Поэтому цепь не имеет несущественных состояний [4]. Будем считать, что цепь Маркова (1) является эргодической, т. е. имеет один класс существенных состояний, и он ациклический. Такое предположение практически всегда

выполнено, поскольку в противном случае матрица $\|p_{ij}(R)\|_{i,j\in\mathcal{X}'} = P^R$ вероятностей переходов за R раундов содержит нули для любого $R \geqslant 1$. Тогда в силу дважды стохастичности

$$\lim_{R\to\infty} P^R = \mathbb{U},$$

где $\mathbb{U} = \dfrac{1}{M} 1^{\downarrow}\mathbf{1}$ — *равномерная матрица* размера M; $\mathbf{1}$ и $1^{\downarrow}$ — вектор-строка и вектор-столбец, состоящие из всех единиц.

Для матрицы A строку с номером i, столбец с номером j и элемент на месте i, j будем обозначать $\mathbf{A}_i$, $A_j^{\downarrow}$ и A_{ij} соответственно. Единичной матрицей будем называть матрицу, являющуюся нейтральным элементом относительно умножения:

$$E = \operatorname{diag}(1, \ldots, 1) = \sum_{1\leqslant i\leqslant M} E_{i,i}.$$

Через $E_{i,j} = e_i^{\downarrow}\mathbf{e}_j$ обозначаем матрицу, единственный ненулевой элемент которой равен 1 и находится на месте i, j; e_k — k-й вектор стандартного базиса $\mathbb{R}^M$. Для вектора $\mathbf{u}$ будем обозначать через $|\mathbf{u}|$ его евклидову норму.

Пусть $\lambda_1 = 1, \lambda_2, \ldots, \lambda_M \in \mathbb{C}$ — *спектр* матрицы переходных вероятностей разностей (МПВР) P, т. е. набор всех её различных собственных чисел (комплексных корней её характеристического многочлена) с учётом их кратностей. Известно [5, с. 266], что спектральный радиус стохастической матрицы равен 1. Поэтому можем считать, что

$$\lambda_1 = 1 \geqslant |\lambda_2| \geqslant \ldots \geqslant |\lambda_M|.$$

Тогда с учётом отсутствия несущественных состояний предположение об эргодичности цепи эквивалентно [6, теорема 6] тому, что кратность корня 1 равна 1, т. е. $1 > |\lambda_2|$.

В п. 1 рассматривается ситуация диагонализуемой матрицы P при условиях доминирования собственного значения λ_2 над остальными и модуля одной из компонент левого и правого собственных векторов (соответствующих значению λ_2) над остальными. Доказано, что тогда для всех достаточно больших R наибольшее значение отклонения от $1/M$ имеет элемент матрицы, координаты которого соответствуют компонентам $\mathbf{B}_2$, $C_2^{\downarrow}$ с наибольшими модулями; для значения отклонения получена асимптотическая формула. Аналогичный результат получен для строк матрицы P^R, наиболее удалённых от равномерного распределения.

В п. 2 и 3 изучаются спектральные атаки различения в двух моделях наблюдений. *Атакой различения на R раундов* шифрсистемы называют статистический критерий для проверки гипотезы H_2 о том, что при шифровании применялись R раундов данной шифрсистемы, против гипотезы H_1 о том, что использовалась случайная подстановка. Для марковских шифров традиционно считается, что МПВР Q равна $\mathbb{U}$ при гипотезе H_1 и P^R — при H_2. Но при шифровании случайно выбранной пары блоков (X, X^*) на фиксированном наборе $\mathbf{k} = (k^1, \ldots, k^R)$ раундовых ключей последовательность (1) раундовых разностей не образует цепь Маркова в общем случае, и МПВР за R раундов не равна P^R. Поэтому возникает вопрос о существовании модели наблюдений, в которой разностная атака различения будет приводить в точности к такой статистической модели.

Автором в [7] предложена такая модель наблюдений — независимые «двублочные тексты». Условия модели отличаются от традиционных и фактически означают, что при гипотезе H_2 каждая пара входных блоков шифруется на своём случайно выбираемом ключе $\mathbf{K}$, а при H_1 — на своей случайно равновероятно выбираемой подстановке.

Это близко к ситуации, когда криптоаналитик наблюдает много коротких открытых сообщений и результатов их шифрования. Тогда непосредственно с помощью атаки различения может проверяться гипотеза о том, что шифрование осуществляется заданной системой. В этой модели без использования эвристических предположений получены оценки вероятностей ошибок атак различения, основанных на разностных и мультиразностных статистиках. В частности, для ситуации фиксированной входной разности и сближения гипотез рассчитаны асимптотические вероятности ошибок критерия отношения правдоподобий [8], основанного на частотах всех выходных разностей.

В п. 2 в модели независимых двублочных текстов со случайной ненулевой равновероятной входной разностью на основе [4] построен спектральный критерий. В п. 3 введена вторая модель наблюдений, развивающая модель [8], — случайные полные кодовые книги, в которой также построена и рассчитана аналогичная атака.

1. Экстремальные вероятности переходов за большое число раундов

Одной из важных задач классического разностного анализа является поиск *экстремальных вероятностей*, под которыми будем понимать вероятности с наибольшим отклонением $|p_{ij}(R) - 1/M|$. При небольших R оно достигается обычно на элементах с максимальным значением $p_{ij}(R)$, для поиска которых в криптографической литературе предлагаются в основном эвристические алгоритмы. Мы будем использовать методы спектрального матричного анализа [5] для описания поведения таких вероятностей, предполагая для простоты, что P диагонализуема над $\mathbb{C}$, т. е. существует такая невырожденная матрица C, что

$$BPC = D = \operatorname{diag}(1, \lambda_2, \ldots, \lambda_M), \quad B = C^{-1}. \tag{2}$$

Усилим условие эргодичности цепи условием доминирования собственного значения λ_2:

$$\lambda_1 = 1 > |\lambda_2| > |\lambda_3| \tag{3}$$

и введём условия доминирования модуля одной компоненты собственных векторов $C_2^{\downarrow}$ и $\mathbf{B}_2$ матрицы P, соответствующих λ_2:

$$\exists a \in \mathcal{X}' \, \forall i \neq a \; (|C_{a2}| > |C_{i2}|); \tag{4}$$

$$\exists b \in \mathcal{X}' \, \forall j \neq b \; (|B_{2b}| > |B_{jb}|). \tag{5}$$

Из условия (3) вытекает, в частности, что спектр не содержит числа, сопряжённого к λ_2, поэтому λ_2 вещественно, как и соответствующие ему собственные векторы $\mathbf{B}_2$, $C_2^{\downarrow}$.

Теорема 1. Пусть выполнены условие (2) диагонализуемости МПВР P и условие (3) доминирования $\lambda_{1,2}$. Тогда существуют такие R_0, R_1, что:

1) при условиях (4) и (5) доминирования модуля одной компоненты

$$\operatorname*{arg\,max}_{(i,j)\in\mathcal{X}'\times\mathcal{X}'} \left| p_{ij}(R) - \frac{1}{M} \right| = (a, b), \quad R \geqslant R_0,$$

$$p_{ab}(R) - \frac{1}{M} \sim C_{a2} B_{2b} \lambda_2^R, \quad R \to \infty; \tag{6}$$

2) при условии (4) для строк $\mathbf{P}_i^{(R)}$ матрицы P^R имеем

$$\arg\max_{i\in\mathcal{X}'}\left|\mathbf{P}_i^{(R)}-\frac{1}{M}\mathbf{1}\right|=a,\quad R\geqslant R_1,$$
$$\left|\mathbf{P}_a^{(R)}-\frac{1}{M}\mathbf{1}\right|\sim|C_{a2}|\,|\mathbf{B}_2|\,|\lambda_2|^R,\quad R\to\infty. \tag{7}$$

Доказательство. В силу равенств $PC=CD$ и $BP=DB$, $D=\operatorname{diag}(\cdot)$, векторы $C_i^{\downarrow}$ и $\mathbf{B}_i$ — собственные, соответствующие λ_i. При $i=1,2$ эти векторы определены однозначно с точностью до мультипликативной константы, поскольку алгебраическая и геометрическая кратности собственных значений λ_1, λ_2 равны 1. С учётом дважды стохастичности матрицы это означает, что $C_1^{\downarrow}$ и $\mathbf{B}_1$ параллельны вектору $1^{\downarrow}$.

Условие $BC=E$ равносильно тому, что $\mathbf{B}_iC_j^{\downarrow}=\mathbb{I}\{i=j\}$, $i,j\in\mathcal{X}'$. Оно сохранится, если одновременно любую строку $\mathbf{B}_i$ умножить, а столбец $C_i^{\downarrow}$ с тем же номером разделить на произвольное число $\lambda\neq0$. Поэтому без ограничения общности считаем, что $(C_1^{\downarrow})^{\top}=\mathbf{B}_1=\dfrac{1}{\sqrt{M}}\mathbf{1}$. Тогда, согласно теореме [5, с. 64] о спектральном разложении, получаем

$$P^R=CD^RB=\sum_{1\leqslant k\leqslant M}\lambda_k^RC_k^{\downarrow}\mathbf{B}_k=\mathbb{U}+\lambda_2^RG_2+\ldots+\lambda_M^RG_M,\ G_k=C_k^{\downarrow}\mathbf{B}_k. \tag{8}$$

1. Докажем сначала формулу (6). Из (8) для всех $i,j\in\mathcal{X}'$ находим

$$p_{ij}(R)=\mathbf{e}_iP^Re_j^{\downarrow}=\sum_{1\leqslant k\leqslant M}\lambda_k^RC_{ik}B_{kj}=\frac{1}{M}+\lambda_2^R\big(C_{i2}B_{2j}+\varepsilon_{ij}(R)\big),$$
$$\text{где}\quad\varepsilon_{ij}(R)=\sum_{3\leqslant k\leqslant M}C_{ik}B_{kj}(\lambda_k/\lambda_2)^R,$$

и $|\varepsilon_{ij}(R)|\leqslant|\lambda_3/\lambda_2|^R\sum\limits_{3\leqslant k\leqslant M}|C_{ik}B_{kj}|\to0$ при $R\to\infty$. Поэтому $\left(p_{ab}(R)-1/M\right)/\lambda_2^R\to$ $\to C_{a2}B_{2b}\neq0$, что равносильно (6).

Из полученных асимптотических представлений и условий (4), (5) следует, что при всех $(i,j)\neq(a,b)$

$$\left|\frac{p_{ij}(R)-1/M}{p_{ab}(R)-1/M}\right|=\left|\frac{C_{i2}B_{2j}+\varepsilon_{ij}(R)}{C_{a2}B_{2b}+\varepsilon_{ab}(R)}\right|\to\left|\frac{C_{i2}}{C_{a2}}\right|\left|\frac{B_{2j}}{B_{2b}}\right|<1,$$

поэтому, начиная с некоторого R_0, модуль числителя меньше модуля знаменателя.

2. Пункт 2 доказывается аналогично; установим сначала формулу (7). Из (8) для всех $i\in\mathcal{X}'$ находим

$$\mathbf{P}_i^{(R)}=\mathbf{e}_iP^R=\sum_{1\leqslant k\leqslant M}\lambda_k^RC_{ik}\mathbf{B}_k=\frac{1}{M}\mathbf{1}+\lambda_2^R\big(C_{i2}\mathbf{B}_2+\varepsilon_i(R)\big),$$
$$\text{где}\quad\varepsilon_i(R)=(\varepsilon_{ij}(R),j\in\mathcal{X}')=\sum_{3\leqslant k\leqslant M}C_{ik}\mathbf{B}_k(\lambda_k/\lambda_2)^R,$$

и при $R\to\infty$

$$|\varepsilon_i(R)|\leqslant|\lambda_3/\lambda_2|^R\sum_{3\leqslant k\leqslant M}|C_{ik}||\mathbf{B}_k|\to0.$$

Здесь для последовательности векторов $\mathbf{u}(R)$ пишем $\mathbf{u}(R)=o(1)$, если $|\mathbf{u}(R)|=o(1)$. Так как

$$\big||\mathbf{u}+o(1)|-|\mathbf{u}|\big|\leqslant|\mathbf{u}+o(1)-\mathbf{u}|=o(1),$$

то $|\mathbf{u}+o(1)|=|\mathbf{u}|+o(1)$, и тогда

$$\left|\mathbf{P}_a^{(R)}-\frac{1}{M}\mathbf{1}\right|=|\lambda_2|^R(|C_{a2}|\,|\mathbf{B}_2|+o(1)),$$

что с учётом неравенства $|C_{a2}|\,|\mathbf{B}_2|>0$ равносильно (7).

Отсюда и из условия (4) следует, что при всех $i\neq a$

$$\frac{\left|\mathbf{P}_i^{(R)}-\frac{1}{M}\mathbf{1}\right|}{\left|\mathbf{P}_a^{(R)}-\frac{1}{M}\mathbf{1}\right|}=\frac{|C_{i2}\mathbf{B}_2+o(1)|}{|C_{a2}\mathbf{B}_2+o(1)|}\to\frac{|C_{i2}|\,|\mathbf{B}_2|}{|C_{a2}|\,|\mathbf{B}_2|}<1,$$

поэтому, начиная с некоторого R_1, модуль числителя меньше модуля знаменателя.

Теорема 1 доказана. ■

Фактически теорема 1 даёт условия, достаточные для единственности наиболее неравномерного элемента (соответственно строки) матрицы P^R при росте R, а также для возникновения *эффекта стабилизации позиции* таких элемента и строки. Из п. 1 теоремы 1 легко получить следующие достаточные условия стабилизации позиции максимальных элементов P^R.

Следствие 1. Пусть выполнены условия (2)–(5) и ограничение

$$(C_{a2}B_{2b}>0 \text{ при } \lambda_2>0) \text{ либо } \left((-1)^R=\operatorname{sign}(C_{a2}B_{2b}) \text{ при } \lambda_2<0\right).$$

Тогда элемент $p_{ab}(R)$ — максимальный в матрице P^R для всех достаточно больших R.

Доказательство. Из доказательства п. 1 теоремы 1 следует, что $\operatorname{sign}(p_{ab}(R)-1/M)=\operatorname{sign}(C_{a2}B_{2b}\lambda_2^R)$ для всех достаточно больших R. При каждом из условий ограничения этот знак равен 1, поэтому с учётом доказательства п. 1 теоремы 1

$$p_{ab}(R)-1/M=|p_{ab}(R)-1/M|>|p_{ij}(R)-1/M|\geqslant p_{ij}(R)-1/M$$

для всех $(i,j)\neq(a,b)$. ■

Возможно, условие (2) диагонализуемости в теореме 1 можно ослабить, преодолевая возникающие при этом трудности с помощью более общих блочно-диагональных матричных представлений типа [9, с. 184] и аппарата матричных норм.

Условие (3) доминирования λ_2 представляется в этом смысле более существенным: при отказе от него может появиться, например, сопряжённая доминирующая пара $|\lambda_{2,3}|>\lambda_4$, которая даст осцилляцию множеств координат с экстремальными значениями. Простейшим примером этого является циркулянт $P=T=\left(\begin{smallmatrix}0&1&0\\0&0&1\\1&0&0\end{smallmatrix}\right)$ — дважды стохастическая матрица, соответствующая полноцикловой подстановке. Как известно [5, с. 245], его спектр совпадает с группой Γ_n корней из 1 степени n. Здесь $n=3$ и спектр $\operatorname{sp}(T)=\{1,\gamma,\gamma^*\}$, $\gamma=e^{2\pi i/3}$. Так как $T^2=\left(\begin{smallmatrix}0&0&1\\1&0&0\\0&1&0\end{smallmatrix}\right)$, $T^3=E$, то при нумерации позиций в матрице элементами группы вычетов $(\mathbb{Z}_n,\oplus)$ получаем

$$p_{ij}(R)-\frac{1}{3}=\begin{cases}2/3 & \text{при } j\equiv i\oplus R \pmod 3,\\ -1/3 & \text{иначе.}\end{cases}$$

Таким образом, соответствующая экстремальным позициям трансверсаль будет изменяться периодически с периодом 3.

Развивая эту идею, построим пример, в котором соответствующая матрице цепь Маркова будет эргодической.

Пример 1. При произвольном n элементы циркулянта, соответствующего вектору $(a_0,\dots,a_{n-1})$, задаются равенством $p_{ij}=a_{j\ominus i}$, $i,j\in\mathbb{Z}_n$, $\ominus$ — вычитание по модулю n. Рассмотрим параметрическое семейство дважды стохастических матриц $P(p)$ — циркулянтов вектора $(p,1-p,0,\dots,0)$, $0\leqslant p\leqslant 1$.

Заметим, что $P(p)=pE+(1-p)T$ — многочлен от циркулянта T, соответствующего полноцикловой подстановке. Спектр матрицы $P(p)$ есть результат применения этого многочлена к спектру T [5, с. 245], поэтому состоит из чисел $z_k(p)=p+(1-p)\gamma^k$, $k\in\mathbb{Z}_n$, $\gamma=e^{2\pi i/n}$. Числу $z_k(p)$ соответствует правый собственный вектор $v_k=(\gamma^{jk}\big|\, j\in\mathbb{Z}_n)$ и левый $v_k^*=v_{n-1-k}$, $k\in\mathbb{Z}_n$. Следовательно, $G_{k+1}=\dfrac{1}{\sqrt{n}}v_k^{\downarrow}\dfrac{1}{\sqrt{n}}\mathbf{v}_k^*$ в разложении (8).

Геометрически описать изменение спектра можно так: при $p=0$ спектр $\operatorname{sp}(P(0))=\operatorname{sp}(T)=\Gamma_n$ является правильным многоугольником, вписанным в единичную окружность. Так как равенства

$$z_k(p)-1=(1-p)(\gamma^k-1),\ k\in\mathbb{Z}_n,\ 0\leqslant p\leqslant 1,$$

означают гомотетию с центром в вершине $z_0(p)=1=\lambda_1$ и коэффициентом подобия $1-p$ (векторы, соединяющие 1 и $z_k(p)$, пропорциональны векторам, соединяющим 1 и γ^k), то при увеличении p этот многоугольник уменьшается (длина сторон равна $(1-p)|\gamma-1|$), оставаясь подобным исходному, и при $p=1$ вырождается в точку 1. Таким образом, при $0<p<1$ все точки, кроме λ_1, лежат внутри единичного круга, $|\lambda_2|<1$, поэтому цепь эргодическая [6, теорема 6].

При $n=3$ найдем p, при котором сопряжённая пара будет лежать на мнимой оси. Из условия $\operatorname{Re}z_1(p)=p+(1-p)\cos(2\pi/3)=0$ получаем $p=1/3$, $P=\frac{1}{3}\left(\begin{smallmatrix}1&2&0\\0&1&2\\2&0&1\end{smallmatrix}\right)$, $\lambda_{2,3}=\pm i/\sqrt{3}$.

Так как здесь

$$G_2=\frac{1}{\sqrt{3}}v_1^{\downarrow}\frac{1}{\sqrt{3}}\mathbf{v}_2=\frac{1}{3}\begin{pmatrix}1\\ \gamma\\ \gamma^2\end{pmatrix}\begin{pmatrix}1 & \gamma^2 & \gamma\end{pmatrix}=\frac{1}{3}\begin{pmatrix}1 & \gamma^2 & \gamma\\ \gamma & 1 & \gamma^2\\ \gamma^2 & \gamma & 1\end{pmatrix}$$

и $G_3=\dfrac{1}{3}v_2^{\downarrow}\mathbf{v}_1=G_2^*$, то, согласно разложению (8),

$$P^R-\mathbb{U}=\lambda_2^RG_2+\lambda_3^RG_3=\lambda_2^RG_2+(\lambda_2^*)^RG_2^*=2\operatorname{Re}(\lambda_2^RG_2)=\frac{2}{3^{R/2}}\operatorname{Re}(i^RG_2).$$

Итак, конфигурация мест максимальных элементов P^R будет изменяться периодически с периодом 4, а экстремальных — с периодом 2. Действительно, это наименьшие периоды последовательностей $\operatorname{Re}(i^Rz)$, $0\neq z\in\mathbb{C}$, и $|\operatorname{Re}(i^Rz)|$, $\arg z\notin\{\pm\pi/4,\pm3\pi/4\}$, $R=1,2,\dots$ Поэтому эффекты стабилизации позиций максимальных и экстремальных элементов здесь отсутствуют.

2. Спектральные атаки различения на марковские шифры по двублочным текстам

2.1. Несмещённая оценка МПВР и билинейная функция от неё

Наблюдаются входные разности $\Delta X(t)=X^*(t)-X(t)$ случайных независимых двублочных открытых текстов $(X(t),X^*(t))$ и выходные разности $\Delta Y(t)=Y^*(t)-Y(t)$ соответствующих им шифртекстов $(Y(t),Y^*(t))$, причём

$$\text{пары } (X(t),X^*(t)) \text{ независимы},\ \Delta X(t)\sim U(\mathcal{X}'),\ 1\leqslant t\leqslant N, \tag{9}$$

т. е. разности $\Delta X(1), \ldots, \Delta X(N)$ независимы и равномерно распределены на $\mathcal{X}'$.

Гипотеза H_1 заключается в том, что шифртексты получены в результате применения подстановок, выбираемых независимо равновероятно из $S(\mathcal{X})$. Гипотеза H_2 заключается в том, что шифртексты получены R-раундовым применением марковского шифра с дважды стохастической МПВ ненулевых разностей P за 1 раунд. Обозначим через Q матрицу вероятностей переходов цепи из раундовых разностей за R шагов, и тогда задача сводится к построению критерия для проверки гипотезы $Q = \mathbb{U}$ против гипотезы $Q = P^R$. Такая постановка соответствует условиям работы [4], и можно воспользоваться построенным в ней спектральным критерием. Далее через P_i, E_i, D_i, $\alpha_i(N)$ обозначаем соответственно вероятностное распределение, математическое ожидание и дисперсию статистик, вероятности ошибок критериев при гипотезе H_i, $i = 1, 2$.

Рассмотрим матричнозначную статистику

$$\hat{Q} = \frac{M}{N} \sum_{1 \leqslant t \leqslant N} E_{\Delta X(t), \Delta Y(t)}.$$

Согласно модели наблюдений (9), она является суммой независимых одинаково распределённых как случайная матрица $E_{x,y}$ матриц, где распределение случайной пары (x, y) задаётся условием

$$x \sim U(\mathcal{X}'),\ \mathsf{P}\{y = j \mid x = i\} = Q_{ij}. \tag{10}$$

Поэтому математическое ожидание статистики равно

$$\mathsf{E}\hat{Q} = M\mathsf{E}E_{x,y} = M\|\,\mathsf{P}\{x = i, y = j\}\|_{i,j \in \mathcal{X}'} = M\|\,\mathsf{P}\{x = i\}Q_{ij}\|_{i,j \in \mathcal{X}'} = Q,$$

то есть $\hat{Q}$ является несмещённой оценкой для Q при каждой гипотезе.

На основе принципа биортогональности [9] проще докажем следующее обобщение теоремы 1 [4].

Лемма 1. Пусть P — дважды стохастическая матрица размера M, $u, v \in \mathbb{R}^M$ — векторы длины 1. Тогда:

1) если (λ, u) или (λ, v) — собственная пара P, $\lambda \neq 1$, то $\mathbf{u}\,\mathbb{U}v^{\downarrow} = 0$;
2) если (λ, u) — левая или (λ, v) — правая собственная пара P, то $\mathbf{u}P^R v^{\downarrow} = \lambda^R \cos(u, v)$;
3) если u, v — левый и правый собственные векторы для значения λ кратности 1, то $\cos(u, v) \neq 0$.

Доказательство.

1) Пусть v — правый собственный вектор (ситуации, когда v — левый собственный или u — собственный, рассматриваются аналогично). Напомним *принцип биортогональности* [9, с. 78]: если $(\lambda_i, \mathbf{x})$ и $(\lambda_j, v^{\downarrow})$ — собственные пары вещественной матрицы A и $\lambda_i \neq \lambda_j$, то $x \perp v$. Действительно, вычитая из равенства $(\mathbf{x}A)v^{\downarrow} = \lambda_i \mathbf{x}v^{\downarrow}$ равенство $\mathbf{x}(Av^{\downarrow}) = \mathbf{x}\lambda_j v^{\downarrow}$, получаем $(\lambda_i - \lambda_j)\langle x, v\rangle = 0$, откуда $\langle x, v\rangle = 0$.

Осталось заметить, что из дважды стохастичности P следует $\mathbf{1}P = \mathbf{1}$, т. е. $(1, \mathbf{1})$ — собственная пара, $\lambda_1 = 1 \neq \lambda$, поэтому

$$\langle \mathbf{1}, \mathbf{v}\rangle = 0, \quad \mathbb{U}v^{\downarrow} = \tfrac{1}{M}1^{\downarrow}\mathbf{1}v^{\downarrow} = \tfrac{1}{M}1^{\downarrow}0 = 0^{\downarrow}.$$

2) Пусть v — правый собственный вектор (ситуация, когда u — левый собственный, рассматривается аналогично). Из определения собственного вектора получаем

$$\mathbf{u}(P^R v^{\downarrow}) = \mathbf{u}\lambda^R v^{\downarrow} = \lambda^R \langle u, v\rangle.$$

Заметим, что $\langle u, v\rangle = \cos(u, v)$ согласно условиям нормированности векторов.

3) Если u, v — левый и правый собственные, соответствующие простому значению λ, то $\langle u, v\rangle \neq 0$ по лемме о неортогональности [9, с. 412].

Лемма 1 доказана. ■

Пусть $\lambda \in \mathbb{R} \setminus \{0, 1\}$ — вещественное собственное значение P. Зафиксируем произвольно пару векторов $(u, v) \in \mathbb{R}^M$, удовлетворяющую условиям п. 1 леммы 1. Она определяет скалярную статистику

$$S(N) = N\mathbf{u}\hat{Q}v^{\downarrow} = N\mathbf{u}\left(\frac{M}{N}\sum_{1\leqslant t\leqslant N} E_{\Delta X(t),\Delta Y(t)}\right)v^{\downarrow} = M\sum_{1\leqslant t\leqslant N} u_{\Delta X(t)}v_{\Delta Y(t)}. \tag{11}$$

Согласно модели наблюдений (9), $S(N)$ — сумма независимых случайных величин, имеющих такое же распределение, как случайная величина

$$\tau = Mu_x v_y,$$

где распределение (x, y) задано условием (10). Из несмещённости статистики $\hat{Q}$ и п. 1 леммы 1 с учётом марковости шифра находим

$$\mathsf{E}S(N) = N\mathbf{u}Qv^{\downarrow} = \begin{cases} 0 & \text{при } H_1, \\ N\mu_2(R) & \text{при } H_2, \text{ где } \mu_2(R) = \mathbf{u}P^R v^{\downarrow}. \end{cases} \tag{12}$$

Далее считаем $\mu_2(R) \neq 0$. Это выполнено, например, при условии п. 3 леммы 1.

2.2. Критерий и его расчёт

Из (11) и (12) следует, что $\mathsf{E}_i\tau = \mathbb{I}\{i = 2\}\mu_2(R)$, $i = 1, 2$. Дисперсия случайной величины τ в силу независимости слагаемых в правой части (11) равна $\mathsf{D}_i S(N)/N$ при гипотезе H_i, это значение обозначим через σ_i^2, $i = 1, 2$. Выражения (12) для среднего и центральная предельная теорема позволяют описать поведение статистики $S(N)$ при больших N следующим образом. При гипотезе H_1 с большой вероятностью значение $S(N)$ лежит в некоторой окрестности нуля, радиус которой определяется величиной σ_1. При гипотезе H_2 с большой вероятностью $S(N)$ имеет знак, совпадающий со знаком величины $\mu_2(R)$. Поэтому для заданного значения $\alpha \in (0, 1)$ рассматриваем односторонние критерии вида

$$d: \; S(N)\,\operatorname{sign}(\mu_2(R)) > \varkappa_{1-\alpha}\sqrt{N} \Longrightarrow H_2. \tag{13}$$

Здесь и далее $\varkappa_\gamma$ — квантиль уровня γ стандартного нормального распределения $\mathcal{N}(0, 1)$.

Будем обозначать через $v^2 = (v_1^2, \ldots, v_M^2)$ векторы, полученные возведением в квадрат каждой компоненты исходного вектора. С учётом леммы 1 аналогично теоремам 2 и 3 из [4] доказывается следующая

Теорема 2. Пусть в модели наблюдений (9) при гипотезе H_2 шифр является марковским с МПВР P,

$$\begin{gathered} (\lambda_2, u) \text{ или } (\lambda_2, v) \text{ — собственная пара } P, \\ \lambda_2 \in \mathbb{R},\ |\lambda_2| \neq 0, 1,\ |u| = |v| = 1,\ \mu_2(R) = \mathbf{u}P^R v^{\downarrow} \neq 0. \end{gathered} \tag{14}$$

Тогда:

1) $\sigma_1^2 = 1,\ \sigma_2^2(R) = M\mathbf{u}^2 P^R v^{2\downarrow} - \mu_2^2(R)$;
2) при $N \to \infty$ критерий (13) имеет асимптотический размер α и

$$\alpha_2(d) = \Phi\left(-\frac{\sqrt{N}|\mu_2(R)| - \varkappa_{1-\alpha}}{\sigma_2(R)}\right) + O\left(\frac{1}{\sqrt{N}}\right).$$

Доказательство.

1) При обеих гипотезах из равенства $\mathsf{P}\{x = i, y = j\} = \dfrac{1}{M}Q_{ij}$ имеем

$$\mathsf{E}\tau^2 = M^2\mathsf{E}(u_x v_y)^2 = M^2 \sum_{i,j\in\mathcal{X}'} \mathsf{P}\{x = i, y = j\} u_i^2 v_j^2 = M \sum_{i,j\in\mathcal{X}'} u_i^2 Q_{ij} v_j^2 = M\mathbf{u}^2 Q v^{2\downarrow}.$$

Отсюда с учётом (12) получаем $\sigma_2^2 = \mathsf{E}\tau^2 - \mu^2$, как и указано в теореме, $\mu = \mu_2(R)$.

При H_1 подставляем $Q = \mathbb{U}$:

$$\sigma_1^2 = M\mathbf{u}^2 \frac{1}{M} 1^{\downarrow}\mathbf{1}v^{2\downarrow} = (\mathbf{u}^2 1^{\downarrow})(\mathbf{1}v^{2\downarrow}) = |u|^2\,|v|^2 = 1.$$

2) При гипотезе H_1, согласно центральной предельной теореме для сумм независимых одинаково распределенных случайных величин, с учётом (12) и равенства $\sigma_1^2 = 1$ имеем

$$\tilde{S}(N) = \frac{S(N)}{\sqrt{N}} \xrightarrow{w} \mathcal{N}(0,1), \quad N \to \infty.$$

Тогда с учётом равенства $\varkappa_{1-\alpha} = -\varkappa_\alpha$ при $\mu > 0$

$$\alpha_1(d) = \mathsf{P}_1\{\tilde{S}(N) > \varkappa_{1-\alpha}\} \to 1 - \Phi(\varkappa_{1-\alpha}) = \alpha.$$

При $\mu < 0$ аналогично

$$\alpha_1(d) = \mathsf{P}_1\{-\tilde{S}(N) > \varkappa_{1-\alpha}\} = \mathsf{P}_1\{\tilde{S}(N) < -\varkappa_{1-\alpha}\} = \mathsf{P}_1\{\tilde{S}(N) < \varkappa_\alpha\} \to \Phi(\varkappa_\alpha) = \alpha.$$

При гипотезе H_2 имеем $\tilde{S}(N) = \dfrac{S(N) - N\mu}{\sigma_2\sqrt{N}} \xrightarrow{w} \mathcal{N}(0,1)$. Тогда в случае $\mu > 0$

$$\alpha_2(d) = \mathsf{P}_2\{S(N) \leqslant \varkappa_{1-\alpha}\sqrt{N}\} = \mathsf{P}_2\left\{\tilde{S}(N) \leqslant \frac{1}{\sigma_2\sqrt{N}}\left(\varkappa_{1-\alpha}\sqrt{N}\sigma_2 - N\mu\right)\right\},$$

что при $N \to \infty$ по теореме Берри — Эссеена [10, с. 449] отличается от значения $\Phi\left(-\dfrac{1}{\sigma_2}(\sqrt{N}\mu - \varkappa_{1-\alpha})\right)$ на величину $O(1/\sqrt{N})$. Отсюда вытекает указанное в условии теоремы выражение для $\alpha_2(d)$. При $\mu < 0$ расчёт $\alpha_2(d)$ проводится аналогично.

Теорема 2 доказана. ■

Приравнивая первое слагаемое в выражении для $\alpha_2(d)$ теоремы 2 к $\beta \in (0,1)$, получаем приближённую формулу для объёма материала, достаточного для различения гипотез с заданными вероятностями ошибок:

$$N_1^*(\alpha,\beta,R) = \frac{(\varkappa_{1-\alpha} + \varkappa_{1-\beta}\sigma_2(R))^2}{\mu_2^2(R)}.$$

Главным членом здесь является знаменатель, поскольку $P^R \to \mathbb{U}$ при $R \to \infty$, $\sigma_2^2(R) \to 1$. Ниже показано, что при ограничении парами векторов, удовлетворяющими

условиям п. 2 леммы 1, максимальное по парам значение модуля $\mu_2(R)$ равно $|\lambda_2|^R$ — см. (17). При таком выборе пары значение $N_1^*(\cdot)$ увеличивается примерно в $|\lambda_2|^{-2}$ раз при увеличении числа R раундов шифрования на 1.

Значение второго наибольшего по модулю собственного числа λ_2 МПВР как характеристики устойчивости марковского шифра к разностным атакам впервые рассмотрел X. Lai в диссертации [11]. После 24-летнего перерыва Lai продолжил исследование с соавторами: в [12] рассмотрены два способа получения верхних оценок $|\lambda_2|$, а также их применения к анализу мини-моделей шифрсистемы IDEA с длинами блока 8 и 16 бит. Получена следующая оценка [12, теорема 2]: шифрсистема с длиной блока n является «практически стойкой к разностному криптоанализу», если $|\lambda_2|^{R-2} < 2^{-(n-1)}$ (заметим, что в оригинале пропущен знак модуля у λ_2).

Эта оценка имеет эвристический характер, поскольку понятие «практической стойкости» строго не определено. Спектральный критерий (13) и его расчёт в теореме 2 позволяют дать более строгую и точную характеристику спектрально-разностной атаки — формулу для объёма материала в зависимости от вероятностей ошибок обоих родов. Она показывает, что в модели независимых двублочных текстов при любом числе раундов на достаточном объёме материала может быть построена атака различения со сколь угодно малыми вероятностями ошибок.

2.3. Оптимизация выбора пары векторов

1. С некоторой точки зрения спектральная атака есть обобщение традиционной. Действительно, при традиционном подходе для отличения гипотезы $Q = P^R$ от гипотезы $Q = \mathbb{U}$ используется статистическая оценка одного (желательно наибольшего по модулю) выделенного элемента матрицы $P^R - \mathbb{U}$:

$$p_{ab}(R) - 1/M = \mathbf{e}_a(P^R - \mathbb{U})e_b^{\downarrow}. \tag{15}$$

Правая часть равенства (15) может рассматриваться как значение билинейной формы из семейства

$$\left\{\mathbf{u}\,(P^R - \mathbb{U})v^{\downarrow} : u, v \in \mathbb{R}^M,\ |u| = |v| = 1\right\}. \tag{16}$$

Спектральный критерий основан на разнице средних значений статистики при разных гипотезах, которую при нормировании также можно записать в виде (16):

$$\frac{1}{N}(\mathsf{E}_2 S(N) - \mathsf{E}_1 S(N)) = \mathbf{u}(P^R - \mathbb{U})v^{\downarrow} = \mu_2(R).$$

Здесь последнее равенство выполнено для пар, удовлетворяющих условиям п. 1 леммы 1. Поэтому логично поставить вопрос о поиске пары (u, v), при которой модуль билинейной формы (16) максимален.

2. Пусть выполнено условие (3) доминирования λ_2.

2*а*) При ограничении парами, удовлетворяющими условиям п. 2 леммы 1, из неравенства Коши — Буняковского получаем, что модуль $\mu_2(R)$ максимизируется при выборе

$$u = v \in \{\mathbf{B}_2/|\mathbf{B}_2|, C_2^{\downarrow}/|C_2^{\downarrow}|\},\ \text{при этом } \mu_2(R) = \lambda_2^R, \tag{17}$$

где $\mathbf{B}_2$, $C_2^{\downarrow}$ — левый и правый собственные векторы матрицы P. Выбор (17) можно назвать *одновекторным построением* спектральной статистики. Из простоты λ_2 вытекает, что таких оптимальных пар (с точностью до знака векторов) будет не более двух.

2*б*) В ситуации выбора произвольной нормированной пары векторов можно предложить следующий путь асимптотически оптимального выбора пары векторов при

условии (2) диагонализуемости P над $\mathbb{C}$. Из спектрального разложения (8) следует, что при больших R главный член значений билинейных форм (16) равен

$$\lambda_2^R \mathbf{u}\, C_2^{\downarrow} \mathbf{B}_2 v^{\downarrow} = \lambda_2^R \langle \mathbf{u}, C_2^{\downarrow} \rangle \langle \mathbf{B}_2 v^{\downarrow} \rangle = \lambda_2^R \cos(u, C_2^{\downarrow}) |C_2^{\downarrow}| \cos(v, \mathbf{B}_2) |\mathbf{B}_2|,$$

что по модулю не превосходит $|\mathbf{B}_2||C_2^{\downarrow}||\lambda_2|^R$. При этом равенство достигается тогда и только тогда, когда u параллелен $C_2^{\downarrow}$, v — параллелен $\mathbf{B}_2$. Это условие обеспечивается выбором

$$u = C_2^{\downarrow}/|C_2^{\downarrow}|,\ v = \mathbf{B}_2/|\mathbf{B}_2|, \quad \text{тогда } \mu_2(R) \approx |\mathbf{B}_2||C_2^{\downarrow}|\lambda_2^R. \tag{18}$$

Порядок последней величины совпадает с порядком значения $|\lambda_2|^R$ из (17).

Такое использование правого собственного вектора матрицы в качестве левого вектора билинейной формы (и наоборот) несколько неожиданно, но, согласно неравенству Коши — Буняковского, значение коэффициента

$$|\mathbf{B}_2||C_2^{\downarrow}| \geqslant \langle \mathbf{B}_2, C_2^{\downarrow} \rangle = 1$$

не меньше 1. Поэтому, возможно, при больших R именно пара векторов (18) будет близка к оптимальной.

3. Спектральные атаки по независимым кодовым книгам

Условие наличия у аналитика полной кодовой книги малоразмерного блочного шифра рассмотрено в [8]. Как представляется, оно является достаточно естественным (например, использовалось блочное шифрование в режиме счётчика) и удобным для проведения экспериментов. В [8] предложена, рассчитана (для 50 % атакуемых ключей) и осуществлена атака, основанная на критерии отношения правдоподобий, на последний раунд Present$(16, R)$ для $7 \leqslant R \leqslant 9$ по одной кодовой книге.

Покажем, что спектральный метод позволяет рассматривать более общую ситуацию наличия нескольких случайных полных кодовых книг.

Будем считать, что ненулевые элементы группы $(\mathcal{X}, +)$ пронумерованы числами от 1 до M, и отождествлять элементы с числами при нумерации координат, описании области суммирования и т. д.

Пусть $\mathscr{K}$ — множество значений одного раундового ключа, $B = B(\mathbf{k}) = (z_0, \ldots, z_M)$ — книга, соответствующая набору $\mathbf{k} \in \mathscr{K}^R$ раундовых ключей, т. е. $z_i = F_R(i, \mathbf{k})$ — результат R-раундового шифрования блока $i \in \mathcal{X}$. Книге соответствует матрица переходных вероятностей разностей

$$P(\mathbf{k}) = \|\mathsf{P}\{F_R(X + a, \mathbf{k}) - F_R(X, \mathbf{k}) = b\}\|_{a,b \in \mathcal{X}'},\ X \sim U(\mathcal{X}).$$

В следующей лемме (достаточно простой, но имеющей важный методический смысл) утверждается, что в модели с выбором случайного набора $\mathbf{K}$ раундовых ключей матрица $P^{(R)}$ переходных вероятностей разностей шифра (за R раундов) есть результат усреднения матриц $P(\mathbf{k})$. Элементы МПВР равны при $a, b \in \mathcal{X}'$:

$$P_{ab}^{(R)} = \mathsf{P}\{F_R(X + a, \mathbf{K}) - F_R(X, \mathbf{K}) = b\}, \quad X \sim U(\mathcal{X}).$$

Лемма 2. Пусть случайный набор $\mathbf{K}$ выбирается из $\mathscr{K}^R$ независимо от входного блока X. Тогда $\mathsf{E}P(\mathbf{K}) = P^{(R)}$. Если при этом компоненты $\mathbf{K}$ независимы и одинаково распределены так, что шифр является марковским с матрицей P переходных вероятностей разностей за 1 раунд, то $\mathsf{E}P(\mathbf{K}) = P^R$.

Доказательство. Обозначим $\xi(x,\mathbf{k}) = \mathbb{I}\{F_R(x+a,\mathbf{k}) - F_R(x,\mathbf{k}) = b\}$ при фиксированных $a, b \neq 0$. С учётом независимости $\mathbf{K}$ и X имеем

$$P_{ab}^{(R)} = \sum_{\mathbf{k}\in\mathscr{K}^R, x\in\mathcal{X}} \mathsf{P}\{\mathbf{K}=\mathbf{k}, X=x\}\xi(x,\mathbf{k}) = \sum_{\mathbf{k}\in\mathscr{K}^R} \mathsf{P}\{\mathbf{K}=\mathbf{k}\} \sum\nolimits_{x\in\mathcal{X}} \mathsf{P}\{X=x\}\xi(x,\mathbf{k}) =$$
$$= \sum_{\mathbf{k}\in\mathscr{K}^R} \mathsf{P}\{\mathbf{K}=\mathbf{k}\}P(\mathbf{k})_{ab} = \mathsf{E}P(\mathbf{K})_{ab},$$

что требовалось доказать. ■

Замечание 1. В [13, с. 264] доказан аналогичный лемме 2 результат о произведениях R элементов матриц; вероятность перехода разностей за R раундов является суммой таких произведений.

Замечание 2. Из доказательства леммы 2 видно, что она справедлива и для неравномерного распределения X.

Для фиксированных $u, v \in \mathbb{R}^M$ аналогично статистике (11) из модели двублочных текстов при суммировании по всем возможным $N_0 = \binom{M+1}{2}$ парам разностей книги строим статистику

$$S(\mathbf{k}) = N_0 \mathbf{u} P(\mathbf{k}) v^{\downarrow} = M \sum_{a,b\in\mathcal{X}'} u_a \frac{M+1}{2} P(\mathbf{k})_{ab} v_b,$$

которую назовём *книжным вкладом*.

Ограничимся далее ситуацией $(\mathcal{X}, +) = (\mathbb{Z}_2^n, \oplus)$. Тогда для пар входных блоков $(i, i+a)$, $(i+a, i)$ разности совпадают,

$$\frac{M+1}{2} P(\mathbf{k})_{ab} = \frac{1}{2} \sum_{0\leqslant i\leqslant M} \mathbb{I}\{F_R(i+a,\mathbf{k}) - F_R(i,\mathbf{k}) = b\} =$$
$$= \sum_{i<j} \mathbb{I}\{j = i+a, F_R(j,\mathbf{k}) - F_R(i,\mathbf{k}) = b\} = \sum_{i<j} \mathbb{I}\{a = j-i\}\mathbb{I}\{b = z_j - z_i\},$$

и получаем

$$S(\mathbf{k}) = M \sum_{a,b} \sum_{i<j} \mathbb{I}\{a = j-i\}\mathbb{I}\{b = z_j - z_i\} u_a v_b = M \sum_{0\leqslant i<j\leqslant M} u_{j-i} v_{z_j - z_i}. \tag{19}$$

При условиях леммы 2 книжный вклад является случайной величиной, среднее значение которой при гипотезе H_2 находим из леммы 2 с учётом свойства линейности математического ожидания случайной матрицы

$$\mathsf{E}_2 S(\mathbf{k}) = N_0 \mathbf{u} P^R v^{\downarrow} = N_0 \mu_2(R),$$

что совпадает с выражением $N\mu_2(R)$ в (12) при $N = N_0$. Объяснить это совпадение можно так: при условиях теоремы 2 независимые слагаемые суммы $S(N)$ имеют такое же распределение, как случайная величина $\tau = M u_{\Delta X} v_{\Delta Y}$ при случайном равновероятном выборе разности ΔX. Но при случайном равновероятном выборе пары мест $\{i, j\}$ в книге B распределения случайных величин $u_{j-i} v_{z_j - z_i}$ и τ совпадают.

Если будем предполагать, как в [8], что распределение $S(\mathbf{k})$ близко к нормальному, то получим, что среднее значение $\mathsf{E}_2 S(\mathbf{k})$ близко к медиане распределения. Поэтому примерно для 50 % ключей модули величин $S(\mathbf{k})$ не меньше $|\mathsf{E}_2 S(\mathbf{k})|$. Тогда односторонний критерий

$$d: \; S(\mathbf{K}) \operatorname{sign}(\mu_2(R)) > \varkappa_{1-\alpha}\sqrt{N_0} \Longrightarrow H_2$$

размера, близкого к α, будет выявлять суммарный шифр не менее чем для примерно 50 % ключей (т. е. $\alpha_2(d) \leqslant \beta \approx 0{,}5$) при условии $\varkappa_{1-\alpha}\sqrt{N_0} \leqslant N_0|\mu_2(R)|$. Оно близко к условию $|\mathcal{X}| \geqslant \dfrac{\varkappa_{1-\alpha}}{|\mu_2(R)|}$.

Обобщая модель [8], будем предполагать, что имеется N_b случайных независимых кодовых книг $B(t) = (z_0^t, \ldots, z_M^t)$, $1 \leqslant t \leqslant N_b$. Гипотеза H_1 заключается в том, что они получены как нижние строки N_b независимых подстановок с равновероятным распределением на $S(\mathcal{X})$. Гипотеза H_2 заключается в том, что книги соответствуют R-раундовому шифрованию при независимом выборе случайных равновероятных наборов $\mathbf{K}(t) \sim U(\mathscr{K}^R)$ раундовых ключей, $1 \leqslant t \leqslant N_b$.

Обозначим далее через $P(t)$ МПВР, соответствующую книге $B(t)$, $1 \leqslant t \leqslant N_b$, и перейдём к рассмотрению статистики

$$S'(N_b) = S_1 + \ldots + S_{N_b}, \quad S_t = N_0 \mathbf{u} P(t) v^{\downarrow}. \tag{20}$$

Временная сложность её вычисления пропорциональна $|\mathcal{X}|^2 N_b$. Тогда с учётом леммы 2 при условиях п. 1 леммы 1, согласно (12),

$$\mathsf{E}_i S'(N_b) = N_b N_0 \mu_2(R)\, \mathbb{I}\{i = 2\}. \tag{21}$$

3.1. Дисперсия книжного вклада при гипотезе H_1

В теореме 2 при обеих гипотезах найдены дисперсии $N\sigma_i^2$ случайной величины $S(N)$ как суммы независимых случайных величин. Статистика $S(\mathbf{k})$ книжного вклада является суммой зависимых случайных величин в силу зависимости выбора пар входных блоков, что значительно усложняет вычисление её дисперсии.

Это удаётся сделать при гипотезе H_1 для группы $\mathcal{X} = \mathbb{Z}_2^n$. Здесь возникают следующие интересные вероятностные конструкции.

Обозначим: $(M)_k = M(M-1)\ldots(M-k+1)$ — факториальная степень числа M для $k \geqslant 1$.

Лемма 3. Пусть случайная величина ξ задана на пространстве элементарных событий Ω мощности $M \geqslant 2$ с классической вероятностью, $\mathsf{E}\xi = 0$. Тогда при случайном выборе $\omega = (\omega_1, \omega_2)$ из Ω^2 для случайной величины $\eta(\omega) = \xi(\omega_1)\xi(\omega_2)$ имеем:

1) $\mathsf{E}\eta = -\dfrac{\mathsf{D}\xi}{M-1}$, если ω_1, ω_2 извлекаются из Ω равновероятно без возвращения;
2) $\mathsf{E}\eta = \dfrac{2\mathsf{D}\xi}{(M-1)(M-2)}$, если ω извлекается из Ω^2 в соответствии с мерой $\mathsf{P}\{\omega\} = \dfrac{M-3+2\mathbb{I}\{\omega_1=\omega_2\}}{(M)_3}$.

Доказательство. В вероятностных схемах обоих пунктов компоненты ω распределены равномерно на Ω в силу симметрии исходов.

1. В схеме п. 1

$$\mathsf{E}\eta = \frac{1}{(M)_2}\sum_{\omega_1\neq\omega_2}\xi(\omega_1)\xi(\omega_2) = \frac{1}{(M)_2}\Big(\sum_{\omega_1}\xi(\omega_1)\sum_{\omega_2}\xi(\omega_2) - \sum_{\omega_1}\xi^2(\omega_1)\Big) = -\frac{M\mathsf{E}\xi^2}{(M)_2} = -\frac{\mathsf{D}\xi}{M-1},$$

поскольку $\sum\limits_{\omega_1}\xi(\omega_1) = M\mathsf{E}\xi = 0$.

2. В схеме п. 2 имеем $\mathsf{P}\{\omega_1 = \omega_2\} = M\dfrac{M-1}{(M)_3} = \dfrac{1}{M-2}$, условное математическое ожидание случайной величины η при условии события $\{\omega_1 = \omega_2\}$ равно

$$\mathsf{E}(\eta \,|\, \omega_1 = \omega_2) = \mathsf{E}\xi^2(\omega) = \mathsf{D}\xi.$$

Условие $\{\omega_1 \neq \omega_2\}$ порождает равномерную условную меру на множестве выборок объёма 2 без возвращения, поэтому, согласно п. 1, $\mathsf{E}(\eta \,|\, \omega_1 \neq \omega_2) = -\dfrac{\mathsf{D}\xi}{M-1}$. Тогда по формуле полного математического ожидания [10, с. 68]

$$\mathsf{E}\eta(\omega) = \mathsf{P}\{\omega_1 = \omega_2\}\mathsf{E}(\eta \,|\, \omega_1 = \omega_2) + \mathsf{P}\{\omega_1 \neq \omega_2\}\mathsf{E}(\eta \,|\, \omega_1 \neq \omega_2) =$$
$$= \frac{1}{M-2}\mathsf{D}\xi + \frac{M-3}{M-2}\,\frac{(-\mathsf{D}\xi)}{M-1} = \frac{\mathsf{D}\xi}{M-2}\left(1 - \frac{M-3}{M-1}\right) = \frac{2\mathsf{D}\xi}{(M-1)(M-2)}.$$

Лемма 3 доказана. ■

Пример 2. Пусть $\xi(\omega) = \omega$. Тогда: а) при $\Omega = \{-1, 1\}$ имеем $\mathsf{D}\xi = 1$, и в схеме п. 1 $\eta \equiv -1$, $\mathsf{E}\eta = -1 = -\dfrac{\mathsf{D}\xi}{2-1}$; б) при $\Omega = \{-1, 0, 1\}$ имеем $\mathsf{D}\xi = 2/3$, и в схеме п. 1 $\eta \sim \begin{pmatrix} 0 & -1 \\ 2/3 & 1/3 \end{pmatrix}$, $\mathsf{E}\eta = -\dfrac{1}{3} = -\dfrac{\mathsf{D}\xi}{3-1}$. В схеме п. 2 имеем $\mathsf{P}\{\omega_1 = \omega_2\} = 1$, поэтому η имеет распределение Бернулли с вероятностью успеха $\dfrac{2}{3} = \mathsf{E}\eta = \dfrac{2\mathsf{D}\xi}{3-1}$, что тоже согласуется с формулой леммы 3.

Замечание 3. Если при условии леммы 3 элементы ω_1, ω_2 выбираются независимо равновероятно из Ω, то $\mathsf{E}\eta = \operatorname{cov}(\xi(\omega_1), \xi(\omega_2)) = 0$. В схеме п. 1 вероятность диагонали нулевая (понижена по сравнению с независимым выбором), поэтому случайные величины $\xi(\omega_1)$, $\xi(\omega_2)$ отрицательно коррелированы. В схеме п. 2 вероятность диагонали повышена по сравнению с независимым выбором: $\dfrac{1}{M-2} > \dfrac{1}{M}$, поэтому $\xi(\omega_1)$, $\xi(\omega_2)$ положительно коррелированы.

Изучим отдельно линейно связанную с книжным вкладом (19) статистику

$$S = \sum_{0 \leqslant i < j \leqslant M} u_{j-i} v_{z_j - z_i}.$$

Теорема 3. Пусть подстановка z с нижней строкой $(z_0, \ldots, z_M)$ выбирается случайно равновероятно из $S(\mathcal{X})$, векторы $u, v \in \mathbb{R}^M$ евклидовой длины 1 ортогональны вектору $\mathbf{1}$. Тогда $\mathsf{E}S = 0$, а при $(\mathcal{X}, +) = (\mathbb{Z}_2^n, \oplus)$, $n \geqslant 2$

$$\mathsf{D}S = \frac{M+1}{2(M-2)} = \frac{2^{n-1}}{2^n - 3}.$$

Доказательство. Для различных $i, j \in \mathcal{X}$ положим $\xi = \xi(i,j) = z_j - z_i$. Легко видеть, что эта случайная величина равномерно распределена на $\mathcal{X}'$. Поэтому

$$\mathsf{E}v_\xi = \sum_{a \in \mathcal{X}'} \frac{1}{M} v_a = \frac{1}{M}\langle v, \mathbf{1}\rangle = 0,\ \mathsf{E}v_\xi^2 = \sum_{a \in \mathcal{X}'} \frac{1}{M} v_a^2 = \frac{1}{M}|v|^2 = \frac{1}{M}, \tag{22}$$

и $\mathsf{E}S = \sum\limits_{i<j} u_{j-i}\mathsf{E}v_{\xi(i,j)} = 0$. Вычислим $\mathsf{D}S = \mathsf{E}S^2$. Представим

$$S^2 = \sum_{i<j} u_{j-i} v_{\xi(i,j)} \sum_{k<l} u_{l-k} v_{\xi(k,l)} = \Sigma_2 + \Sigma_1 + \Sigma_0, \tag{23}$$

где Σ_m — сумма случайных величин $u_{j-i}u_{l-k}v_{\xi(i,j)}v_{\xi(k,l)}$ по всем таким парам пар $(i,j), (k,l)$, что

$$0 \leqslant i < j \leqslant M,\ 0 \leqslant k < l \leqslant M,\ |\{i,j\} \cap \{k,l\}| = m, \tag{24}$$

$m = 0, 1, 2$. Вычислим средние значения этих сумм.

1. Легко видеть, что $\Sigma_2 = \sum\limits_{i<j} u_{j-i}^2 v_{\xi(i,j)}^2$, и с учётом (22)

$$\mathsf{E}\Sigma_2 = \sum_{i<j} u_{j-i}^2 \frac{1}{M} = \frac{1}{M}\sum_{i<j} u_{i+j}^2 = \frac{1}{2M}\sum_{i\neq j} u_{i+j}^2,$$

где во втором переходе использовано равенство $-i = i$ для $i \in \mathbb{Z}_2^n$. Для любого фиксированного $a \in \mathcal{X}'$ уравнение $i + j = a$ имеет ровно $M + 1$ решений относительно $(i, j) \in \mathcal{X}^2$, поэтому

$$\mathsf{E}\Sigma_2 = \frac{M+1}{2M}\sum_{a\in\mathcal{X}'} u_a^2 = \frac{M+1}{2M}. \tag{25}$$

2. Для вычисления $\mathsf{E}\Sigma_1$ найдём совместное распределение случайных величин $\xi = \xi(i, j)$ и $\eta = \xi(k, l)$. С учётом симметрии, не ограничивая общности, считаем $\xi = z_0 + z_1$, $\eta = z_0 + z_2$. Так как $z_1 \neq z_2$, то $\mathsf{P}\{\xi = \eta\} = 0$.

Далее, для любых $c \neq d$ из $\mathcal{X}'$ в событии $\{\xi = c, \eta = d\}$ при фиксации $z_0 = a$ определяются однозначно значения $z_1 = a + c$, $z_2 = a + d$. Поэтому $\mathsf{P}\{\xi = c, \eta = d\} = \dfrac{M}{(M+1)_3} = \dfrac{1}{(M)_2}$. Таким образом, $\omega = (\xi, \eta)$ имеет распределение выборки без возвращения из $\mathcal{X}'$, и с учётом (22) по п. 1 леммы 3 получим

$$\mathsf{E}v_{\xi(i,j)}v_{\xi(k,l)} = \frac{-\mathsf{E}v_\xi^2}{|\mathcal{X}'|} = -\frac{1}{M(M-1)},\ |\{i,j\}\cap\{k,l\}| = 1. \tag{26}$$

Это равенство поможет вычислить возникающую сумму $S_1 = \sum u_{i+j}u_{k+l}$ по индексам (24) при $m = 1$. Одной тройке $0 \leqslant a < b < c \leqslant M$ соответствует ровно шесть индексов со свойством $\{i, j\} \cup \{k, l\} = \{a, b, c\}$: тремя способами из $\{a, b, c\}$ выбираем повторяющийся элемент и двумя — суммируемый с ним индекс в u_{i+j}. Сумму соответствующих шести слагаемых можно записать в виде $\sum\limits_{(i,j,k)\in\mathcal{P}\{a,b,c\}} u_{i+j}u_{i+k}$ по всем перестановкам множества $\{a, b, c\}$. Интерпретируя (i, j, k) как реализацию случайного вектора (z_0, z_1, z_2), находим

$$S_1 = \sum_{(i,j,k)} u_{i+j}u_{i+k} = (M+1)_3\mathsf{E}u_{\xi(0,1)}u_{\xi(0,2)} = (M+1)_3\frac{-1}{M(M-1)} = -(M+1),$$

где в предпоследнем переходе использовано выражение (26). Итак,

$$\mathsf{E}\Sigma_1 = \sum u_{i+j}u_{k+l}\mathsf{E}v_{\xi(i,j)}v_{\xi(k,l)} = S_1\frac{-1}{M(M-1)} = \frac{M+1}{M(M-1)}. \tag{27}$$

3. Вычисление $\mathsf{E}\Sigma_0$ проведём аналогично п. 2. Исследуем совместное распределение случайных величин $\xi = z_0 + z_1$ и $\eta = z_2 + z_3$. Здесь для $c = d \in \mathcal{X}'$ в событии $\{\xi = \eta = c\}$ при фиксации $z_0 = a$, $z_2 = b$ определяются однозначно значения $z_1 = a + c$, $z_3 = b + c$, и в четвёрке (z_0, z_1, z_2, z_3) все элементы различны тогда и только тогда, когда $b \notin \{a, a + c\}$. Следовательно,

$$\mathsf{P}\{\xi = \eta = c\} = \frac{(M+1)(M-1)}{(M+1)_4} = \frac{M-1}{(M)_3}.$$

При $c \neq d$ из $\mathcal{X}'$ в событии $\{\xi = c, \eta = d\}$ при фиксации z_0, z_2 значениями $a, b \in \mathcal{X}$ четвёрка $(z_0, z_1, z_2, z_3) = (a, a+c, b, b+d)$ состоит из различных элементов тогда и только

тогда, когда $b \notin \{a, a+c, a+c+d, a+d\}$. Отсюда $\mathsf{P}\{\xi = c, \eta = d\} = \dfrac{(M+1)(M-3)}{(M+1)_4} = \dfrac{M-3}{(M)_3}$.

Заметим, что:

а) сумма вероятностей распределения (ξ, η) равна

$$M\frac{M-1}{(M)_3} + M(M-1)\frac{M-3}{(M)_3} = \frac{1}{M-2} + \frac{M-3}{M-2} = 1;$$

б) при $n = 2$ имеем $M = 3$, и распределение (ξ, η) сосредоточено на диагонали.

Таким образом, $\omega = (\xi, \eta)$ имеет распределение, как в п. 2 леммы 3, поэтому здесь с учётом (22)

$$\mathsf{E}v_{\xi(i,j)}v_{\xi(k,l)} = \frac{2/M}{(M-1)(M-2)} = \frac{2}{(M)_3}, \ \{i,j\} \cap \{k,l\} = \varnothing. \tag{28}$$

Далее находим

$$S_0 = \sum u_{i+j}u_{k+l} = \frac{1}{4}\sum_{i\neq j, k\neq l} u_{i+j}u_{k+l} = \frac{(M+1)_4}{4}\mathsf{E}u_{\xi(0,1)}u_{\xi(2,3)} = \frac{(M+1)_4}{4}\frac{2}{(M)_3} = \frac{M+1}{2},$$

где в предпоследнем переходе использовано (28). Итак,

$$\mathsf{E}\Sigma_0 = S_0\mathsf{E}v_{\xi(0,1)}v_{\xi(2,3)} = \frac{M+1}{2}\frac{2}{(M)_3} = \frac{M+1}{(M)_3}. \tag{29}$$

Окончательно из представления (23) с учётом (25), (27) и (29) находим

$$\mathsf{E}S^2 = \frac{M+1}{2M} + \frac{M+1}{M(M-1)} + \frac{M+1}{(M)_3} = \frac{M+1}{M}\left(\frac{1}{2} + \frac{1}{M-1} + \frac{1}{(M-1)(M-2)}\right) = \frac{M+1}{2(M-2)}.$$

Теорема 3 доказана. ■

Замечание 4. Величина $\mathsf{E}S^2 = \dfrac{1}{2}\left(1 + \dfrac{3}{M-2}\right)$ монотонно убывает от 1 к $1/2$ при $M = 5, 6, \ldots$

Замечание 5. Если бы слагаемые в определении S были независимы, то дисперсия с учётом п. 1 доказательства теоремы 3 была бы равна $\sum\limits_{i<j} u_{i+j}^2/M = (M+1)/(2M)$, что равно доле $(M-2)/M$ реального значения.

Замечание 6. При $n = 2, 3$ равенства (26), (28) и утверждение теоремы 3 проверены перебором на ЭВМ всех подстановок степеней 4 и 8 соответственно при некоторых u, v. Так, при $n = 2$ использовались $u = (1, -1, 0)/\sqrt{2}$, $v = (2, 1, -1)/\sqrt{6}$; если здесь целочисленные векторы не нормировать, то результаты будут больше соответственно в $|v|^2 = 6$ и в $|u|^2|v|^2 = 12$ раз.

Из теоремы 3 вытекает, что при гипотезе H_1

$$\mathsf{D}_1 S(\mathbf{k}) = M^2\frac{M+1}{2(M-2)} = N_0\frac{M}{M-2}, \quad M = 2^n - 1, \tag{30}$$

что немного (в $M/(M-2)$ раз) больше дисперсии суммы $\mathsf{D}_1 S(N_0) = N_0$ с таким же количеством независимых слагаемых, имеющих такое же распределение.

3.2. Критерий и оценка требуемого числа книг

С учётом формулы (30) для дисперсии книжного вклада при гипотезе H_1 рассматриваем критерий

$$d:\ S'(N_b)\ \mathrm{sign}(\mu_2(R)) > \varkappa_{1-\alpha}\sqrt{\frac{NM}{M-2}} \Longrightarrow H_2, \tag{31}$$

где $N = N_0 N_b$.

Теорема 4. Пусть при $\mathbf{K} \sim U(\mathscr{K}^R)$ шифр с алфавитом $(\mathcal{X}, +) = (\mathbb{Z}_2^n, \oplus)$ является марковским с МПВР P, удовлетворяющей условию (3) доминирования λ_2, и векторы статистики $S'(N_b)$ выбраны согласно (14). Тогда для любого α, $0 < \alpha < 1$, при $N_b \to \infty$ критерий (31) имеет асимптотический размер α и

$$\alpha_2(d) = \Phi\Big(-\frac{\sqrt{N}|\mu_2(R)| - \varkappa_{1-\alpha}\sqrt{M/(M-2)}}{\sqrt{\mathsf{D}_2 S(\mathbf{k})/N_0}}\Big) + O\left(\frac{1}{\sqrt{N_b}}\right).$$

Доказательство. Слагаемые в представлении (20) для $S'(N_b)$ при обеих гипотезах являются независимыми одинаково распределёнными случайными величинами и, согласно центральной предельной теореме, распределения нормированных сумм $\dfrac{S'(N_b) - \mathsf{E}_i S'(N_b)}{\sqrt{N_b \mathsf{D}_i S(\mathbf{k})}}$ сходятся к $\mathcal{N}(0,1)$, $i = 1, 2$. Далее асимптотические вероятности ошибок критерия рассчитываются стандартным способом с учётом равенств (21) и (30). В частности, при $\mu_2(R) > 0$ с учётом неравенства Берри — Эссеена

$$\alpha_2(d) = \mathsf{P}_2\left\{S' \leqslant \varkappa_{1-\alpha}\sqrt{\frac{NM}{M-2}}\right\} =$$

$$= \mathsf{P}_2\Big\{\frac{S' - N\mu_2(R)}{\sqrt{N_b \mathsf{D}_2 S(\mathbf{k})}} \leqslant \frac{\varkappa_{1-\alpha}\sqrt{NM/(M-2)} - N\mu_2(R)}{\sqrt{N_b \mathsf{D}_2 S(\mathbf{k})}}\Big\} =$$

$$= \Phi\Big(\frac{\varkappa_{1-\alpha}\sqrt{M/(M-2)} - \sqrt{N}\mu_2(R)}{\sqrt{\mathsf{D}_2 S(\mathbf{k})/N_0}}\Big) + O\left(\frac{1}{\sqrt{N_b}}\right).$$

Теорема 4 доказана. ■

Предполагая, что при больших R шифрование даёт подстановки, близкие к случайным равновероятным, и поэтому $\mathsf{D}_2 S(\mathbf{k}) \approx \mathsf{D}_1 S(\mathbf{k})$, приравниваем главный член в $\alpha_2(d)$ к β. Таким образом получаем оценку для числа кодовых книг при больших значениях N_b, $M = 2^n - 1$:

$$\sqrt{N}|\mu_2(R)| - \varkappa_{1-\alpha}\sqrt{\frac{M}{M-2}} \approx \varkappa_{1-\beta}\sqrt{\frac{M}{M-2}},$$

$$N_b \approx \frac{(\varkappa_{1-\alpha} + \varkappa_{1-\beta})^2 \dfrac{M}{M-2}}{\dbinom{M+1}{2}\mu_2^2(R)} \approx \frac{2(\varkappa_{1-\alpha} + \varkappa_{1-\beta})^2}{(2^n \mu_2(R))^2}.$$

Отметим, что в 50 %-й атаке для обоснования этой оценки не требуется предположения о дисперсии $\mathsf{D}_2 S(\mathbf{k})$ и $\varkappa_{1-\beta} = 0$. Тогда при одновекторном построении (17) статистики имеем $\mu_2(R) = \lambda_2^R$, $N_{b,50\,\%} \approx \dfrac{\varkappa_{1-\alpha}^2}{2^{2n-1}\lambda_2^{2R}}$.

ЛИТЕРАТУРА

1. *Lai X., Massey J., and Murphy S.* Markov ciphers and differential cryptanalysis // LNCS. 1991. V. 547. P. 17–38.
2. *Погорелов Б. А., Пудовкина М. А.* Разбиения на биграммах и марковость алгоритмов блочного шифрования // Матем. вопр. криптогр. 2017. Т. 8. Вып. 1. С. 107–142.
3. *Денисов О. В.* Критерии марковости алгоритмов блочного шифрования // Прикладная дискретная математика. 2018. № 41. С. 28–37.
4. *Денисов О. В.* Спектральный критерий для проверки гипотез о случайных подстановках // Матем. вопр. криптогр. 2016. Т. 7. Вып. 3. С. 19–28.
5. *Ланкастер П.* Теория матриц. М.: Наука, 1978. 280 с.
6. *Альпин Ю. А., Альпина В. С.* Теорема Перрона — Фробениуса: доказательство с помощью цепей Маркова // Зап. научн. сем. ПОМИ. 2008. Т. 359. С. 5–16.
7. *Денисов О. В.* Атаки различения на блочные шифрсистемы по разностям двублочных текстов // Прикладная дискретная математика. 2020. № 48. С. 43–62.
8. *Albrecht M. and Leander G.* An all-in-one approach to differential cryptanalysis for small block ciphers // LNCS. 2013. V. 7707. P. 1–15.
9. *Хорн Р., Джонсон Ч.* Матричный анализ. М.: Мир, 1989. 655 с.
10. *Боровков А. А.* Теория вероятностей. М.: Эдиториал УРСС, 1999. 472 с.
11. *Lai X.* On the Design and Security of Block Ciphers: Dissertation for the degree of Doctor of Technical Sciences. Swiss Federal Institute of Technology, Zurich, 1992. 118 p.
12. *Xue W., Lin T., Shun X., et al.* On the estimation of the second largest eigenvalue of Markov ciphers // Security Comm. Networks. 2016. V. 9. P. 2093–2099.
13. *Vaudenay S.* On the security of CS-cipher // LNCS. 1999. V. 1636. P. 260–274.

REFERENCES

1. *Lai X., Massey J., and Murphy S.* Markov ciphers and differential cryptanalysis. LNCS, 1991, vol. 547, pp. 17–38.
2. *Pogorelov B. A. and Pudovkina M. A.* Razbieniya na bigrammah i markovost' algoritmov blochnogo shifrovaniya [Partitions on bigrams and Markov property of block ciphers]. Matematicheskie Voprosy Kriptografii, 2017, vol. 8, iss. 1, pp. 107–142. (in Russian)
3. *Denisov O. V.* Kriterii markovosti algoritmov blochnogo shifrovaniya [Criteria for Markov block ciphers]. Prikladnaya Diskretnaya Matematika, 2018, no. 41, pp. 28–37. (in Russian)
4. *Denisov O. V.* Spektral'nyj kriterij dlya proverki gipotez o sluchajnyh podstanovkah [Spectral criterion for testing hypotheses about random substitutions]. Matematicheskie Voprosy Kriptografii, 2016, vol. 7, iss. 3, pp. 19–28. (in Russian)
5. *Lankaster P.* Theory of Matrices. N.Y.; London, Academic Press, 1969.
6. *Al'pin Yu. A. and Al'pina V. S.* Teorema Perrona — Frobeniusa: dokazatel'stvo s pomoshch'yu cepej Markova [Perron — Frobenius theorem: proof using Markov chains]. Zapiski Nauchnyh Seminarov POMI, 2008, vol. 359, pp. 5–16. (in Russian)
7. *Denisov O. V.* Ataki razlicheniya na blochnye shifrsistemy po raznostyam dvublochnyh tekstov [Distinguishing attacks on block ciphers by differentials of 2-blocks texts]. Prikladnaya Diskretnaya Matematika, 2020, no. 48, pp. 43–62. (in Russian)
8. *Albrecht M. and Leander G.* An all-in-one approach to differential cryptanalysis for small block ciphers. LNCS, 2013, vol. 7707, pp. 1–15.
9. *Horn R. and Johnson C.* Matrix Analysis. Cambridge University Press, 1986.
10. *Borovkov A. A.* Teoriya veroyatnostej [Probability Theory]. Moscow, URSS, 1999. 472 p. (in Russian)

11. *Lai X.* On the Design and Security of Block Ciphers: Dissertation for the degree of Doctor of Technical Sciences. Swiss Federal Institute of Technology, Zurich, 1992. 118 p.
12. *Xue W., Lin T., Shun X., et al.* On the estimation of the second largest eigenvalue of Markov ciphers // Security Comm. Networks, 2016, vol. 9, pp. 2093–2099.
13. *Vaudenay S.* On the security of CS-cipher. LNCS, 1999, vol. 1636, pp. 260–274.

МАТЕМАТИЧЕСКИЕ ОСНОВЫ КОМПЬЮТЕРНОЙ БЕЗОПАСНОСТИ

УДК 004.056.53

ОБНАРУЖЕНИЕ СЕРВЕРНЫХ ТОЧЕК ВЗАИМОДЕЙСТВИЯ В ВЕБ-ПРИЛОЖЕНИЯХ НА ОСНОВЕ АНАЛИЗА КЛИЕНТСКОГО JavaScript-КОДА[1]

Д. А. Сигалов, А. А. Хашаев, Д. Ю. Гамаюнов

МГУ имени М. В. Ломоносова, г. Москва, Россия

Рассматривается задача обнаружения серверных точек взаимодействия в динамических веб-приложениях в контексте анализа защищенности веб-приложений в модели «черного ящика». Предложен метод повышения полноты обнаружения серверных интерфейсов на основе статического анализа клиентского кода JavaScript для поиска в нем функций, которые порождают HTTP-запросы к серверной стороне приложения, и определения возможных значений параметров найденных функций. В контексте решаемой задачи статический анализ позволяет находить такие функции в том числе в недостижимом или мёртвом JavaScript-коде, что в ряде случаев позволяет обнаружить серверные интерфейсы, скрытые для динамического анализа. Проведено экспериментальное исследование полноты выявления серверных точек взаимодействия предложенным алгоритмом на синтетическом веб-приложении, уязвимом к SQL-инъекции, и сравнение с популярными сканерами защищенности веб-приложений. Показано, что использование статического анализа клиентского JavaScript-кода в дополнение к традиционному динамическому краулингу приложений может значительно повысить полноту выявления серверных точек взаимодействия в веб-приложениях.

Ключевые слова: *веб-приложения, статический анализ, JavaScript.*

DOI 10.17223/20710410/53/3

DETECTING SERVER-SIDE ENDPOINTS IN WEB APPLICATIONS BASED ON STATIC ANALYSIS OF CLIENT-SIDE JavaScript CODE

D. A. Sigalov, A. A. Khashaev, D. Yu. Gamayunov

Lomonosov Moscow State University, Moscow, Russia

E-mail: {asterite, arthur, gamajun}@seclab.cs.msu.ru

[1]В работе использованы результаты проекта «Система автоматического поиска уязвимостей в веб-приложениях на основе обработки больших данных», выполняемого в рамках реализации Программы Центра компетенций Национальной технологической инициативы «Центр хранения и анализа больших данных», поддерживаемого Министерством науки и высшего образования РФ по Договору МГУ имени М. В. Ломоносова с Фондом поддержки проектов Национальной технологической инициативы от 15.08.2019 № 7/1251/2019.

The problem of server-side endpoint detection in the context of blackbox security analysis of dynamic web applications is considered. We propose a method to increase coverage of server-side endpoint detection using static analysis of client-side JavaScript code to find functions which generate HTTP requests to the server-side of the application and reconstruct parameters for those functions. In the context of application security testing, static analysis allows to find such functions even in dead or unreachable JavaScript code, which cannot be achieved by dynamic crawling or dynamic code analysis. Evaluation of the proposed method and its implementation has been done using synthetic web application with endpoints vulnerable to SQL injections, and the same application was used to compare the proposed method with existing solutions. Evaluation results show that adding JavaScript static analysis to traditional dynamic crawling of web applications may significantly improve server-side endpoint coverage in blackbox application security analysis.

Keywords: *web applications, static analysis, JavaScript.*

Введение

Рассматривается задача обнаружения уязвимостей в веб-приложениях в модели «чёрного ящика», когда для анализа доступна только клиентская часть веб-приложения и интерфейсные точки взаимодействия на его серверной стороне. В этой модели процесс поиска уязвимостей обычно состоит из трёх этапов: поиска доступных функций веб-приложения (их также иногда называют серверными точками взаимодействия, серверными интерфейсами, точками входа в приложение или API-вызовами); вызова функций приложения с различными значениями параметров; анализа результатов выполнения действий [1]. Данная работа посвящена автоматизации поиска доступных функций веб-приложения.

Поиск доступных функций мы сводим к поиску так называемых *точек ввода данных* (DEP — Data Entry Points) [1]. Под DEP будем понимать *спецификацию* допустимого для некоторой функции веб-приложения множества HTTP-запросов, заданную как совокупность фиксированных элементов запроса и ограничения на вариативную часть запроса. Фиксированная часть запроса идентифицирует DEP (часто это URL и метод), вариативная является совокупностью значений его параметров (часто это query-параметры, тело запроса, значения заголовков). Объединение множеств порождаемых всеми DEP запросов веб-приложения даёт множество всех допустимых HTTP-запросов, которые принимает сервер.

Множество допустимых HTTP-запросов полностью определяется кодом серверной части приложения и, возможно, его динамическим состоянием (конфигурацией, историей функционирования). При анализе веб-приложения как чёрного ящика эта информация недоступна, поэтому множество допустимых DEP определить полно и точно не представляется возможным, но можно решать задачу построения аппроксимирующего множества DEP. DEP полностью определяют пространство атаки для исследуемого веб-приложения, поэтому при поиске DEP наиболее важным критерием качества является полнота.

Строить аппроксимацию множества DEP можно следующими способами:

— статическим анализом HTML в ответах сервера и выявлением статических элементов интерфейса, которые могут порождать запросы к серверу;
— динамическим обходом приложения с помощью управляемого браузера (Selenium Web Driver, Headless Chrome, Firefox, WebKit, htmlunit и др.);

— статическим анализом JavaScript для поиска вызовов `XMLHttpRequest()`, `fetch()` и библиотечных обёрток вокруг браузерного API;
— активным управляемым перебором запросов к серверной стороне приложения и анализом ответов для обнаружения DEP, недоступных из интерфейса; примером такого перебора является поиск URL-путей по словарю, так называемый дирбастинг, по результатам работы которого может также использоваться фаззинг.

Дирбастинг или фаззинг в общем случае не позволяют определить имена параметров HTTP-запроса и возможные диапазоны их значений и обычно направлены на поиск функций по словарю значений фиксированной части DEP [2]. Остальные методы так или иначе получают информацию из пользовательского интерфейса приложения, т. е. HTML-страниц и подключаемых ими ресурсов.

Сложность аппроксимации DEP по интерфейсу существенно различается для статических и динамических интерфейсов: в случае статического интерфейса набор видимых серверных точек взаимодействия однозначно определяется HTML, семантика элементов разметки и их реакция на действия пользователя всегда фиксированы.

В случае динамического интерфейса использование встроенных программ на языке JavaScript расширяет его функциональность, в частности программа может инициировать отправку запроса на сервер, а также динамически создавать, изменять и удалять элементы HTML-разметки страницы. В результате часть допустимых HTTP-запросов формируется в результате выполнения JavaScript-программы, и аппроксимация DEP лишь на основе HTML будет существенно неполной. При этом, по данным статистики, 97,1 % веб-сайтов используют на своих страницах JavaScript [3].

Предельным случаем динамического интерфейса в настоящее время являются так называемые *single-page applications* (SPA), в которых при первом обращении к веб-приложению один раз загружается HTML-страница с JavaScript-программой и в дальнейшем все запросы с клиента порождаются только в результате выполнения этой программы.

Функции веб-приложения можно классифицировать по их доступности из веб-интерфейса: публичные, пользовательские и скрытые функции [2]. Интерфейсная часть публичных функций доступна любому пользователю, возможно, после обращения к некоторым другим публичным функциям. Пользовательские функции, как правило, содержатся в защищённой части веб-приложения, которая доступна только аутентифицированному пользователю. Отметим следующую особенность динамических интерфейсов: если для статических интерфейсов, как правило, соблюдается изоляция функций, то в динамических интерфейсах JavaScript-программы часто нарушают такую изоляцию [2]. Наконец, скрытые функции полностью недоступны в интерфейсе.

Примером пользовательских функций является административная панель управления веб-приложением. Анонимный пользователь не увидит ссылку на административную панель или форму смены пароля администратора в своём интерфейсе (по крайней мере, пока не пройдёт аутентификацию). При этом, как упоминается в [2], возможно такое, что клиентский JavaScript-код, соответствующий административной панели, будет доступен анонимному пользователю. Примером скрытых функций может быть старая версия функции поиска по сайту, которая была удалена из пользовательского интерфейса, но при этом продолжает поддерживаться сервером — например, потому, что запланированное обновление кода сервера ещё не произошло, либо потому, что разработчики не сочли нужным удалять её из сервера вообще. Другим примером может служить отладочный интерфейс на сервере, который используется разработчиками

для получения диагностики о состоянии сервера и его отладки и не предназначен для использования рядовыми пользователями сайтов.

В современных сканерах безопасности чаще всего используется динамический обход в качестве основного способа поиска серверных точек взаимодействия, при этом у динамического обхода есть недостатки: он не позволяет обнаруживать скрытые функции, может не обнаружить публичные функции, требующие сложной навигации по интерфейсу, и плохо работает для автоматического обнаружения пользовательских функций, так как требует аутентификации в разных пользовательских ролях, что в модели «чёрного ящика» может быть сложно или невозможно. Мы протестировали ряд популярных сканеров безопасности веб-приложений и убедились, что во многих случаях они не находят пользовательские и скрытые функции, даже когда соответствующий им JavaScript-код доступен на странице. Сканеры не находили даже простые функции, где вызывающий их JavaScript содержал URL в явном виде и параметры запроса в виде констант, локализованных в одном линейном участке клиентского кода (полные результаты описаны в п. 4).

В данной работе предлагается подход к поиску DEP на основе статического анализа JavaScript-кода клиентской части приложения. Алгоритм ищет вызовы функций, порождающих запросы на сервер, и пытается определить возможные значения аргументов найденных функций с помощью статического анализа, работающего над абстрактным синтаксическим деревом JavaScript-кода.

1. Особенности реализации JavaScript DEP в реальных приложениях

Характерной особенностью работ по анализу JavaScript-кода является то, что авторы в явной или неявной форме делают ряд предположений об устройстве реальных JavaScript-программ и этим формируют ограниченное подмножество языка, подлежащее анализу. Эти предположения используются для обоснования практической применимости предлагаемых методов анализа, которые не являются формально полными или корректными — например, не моделируют некоторые особенности и возможности языка либо имеют принципальные технические ограничения по скорости работы и объёму используемой памяти. Иными словами, исследователи предполагают, что на практике в реальных приложениях не используется целый ряд возможностей языка JavaScript, а потому и методы анализа JavaScript могут игнорировать наличие таких возможностей без ущерба для точности и полноты анализа.

В [4] исследуется вопрос о применимости таких предположений к анализу кода общего вида. В качестве анализируемых веб-сайтов и приложений выбраны 100 сайтов из наиболее популярных по версии Alexa [5]. В настоящей работе рассматриваются частные вопросы, относящиеся к анализу точек ввода данных.

HTTP-запрос из JavaScript-кода в браузерном окружении можно сделать следующими способами:

- явным:
  - используя API `XMLHttpRequest`;
  - используя функцию `fetch`.
- неявным:
  - статически: отправив форму, заданную в статической HTML-разметке страницы;
  - динамически: например, создав и отправив форму, изменив свойство `window.location`, выполнив вызов `window.open` и так далее.

Отметим, что неявный статический способ не представляет сложности для анализа, а динамический тяжело поддаётся полному описанию, поэтому в рамках данной работы рассматриваются исключительно явные способы. Стоит также отметить, что разработчики приложений часто используют библиотеки, такие, как jQuery [6], которые предоставляют обёртки (например, функцию `$.ajax`) над браузерным API для совершения HTTP-запросов.

1.1. Мотивирующие примеры

Рассмотрим следующие примеры:

1) На веб-сайте присутствует HTML-форма, при отправке которой вызывается зарегистрированный JavaScript-обработчик, который выполняет HTTP-запрос на сервер. При этом используется решение reCAPTCHA [7] для защиты формы от отправки автоматизированными средствами (ботами). Однако валидация CAPTCHA производится исключительно на стороне клиента, в этом обработчике, что само по себе является уязвимостью. Такая конфигурация не анализируется динамическими средствами, основанными на управляемом браузере, поскольку требует автоматизированного решения CAPTCHA. Вместе с тем статический анализ мог бы проанализировать путь исполнения программы, в которой условие проверки CAPTCHA на клиентской стороне выполнено, и таким образом обнаружить целевую точку ввода данных.

2) На веб-сайте присутствует JavaScript-сценарий из листинга 1. Если предположить, что функция `window.__makePostCallback` не вызывается, то динамический анализ также не обнаружит скрытую функцию, относящуюся к данной точке ввода данных. Отметим также, что в строке 15 присутствует нетривиальная проверка содержимого Cookie, которую статический анализ, в отличие от динамического, может проигнорировать.

```
1  var baseURL = "/site/",
2      config = {
3          section: "forum",
4          apiVersion: "1.1",
5          pageType: "forumPage",
6          trackingSegment: 374000000,
7      },
8      pageInfo = "type=" + config.pageType + "&page=" + location;
9  (function() {
10     function makePost(user, title, postText) {
11         var actionEndpoint = "post.php";
12         var tracking = pageInfo + "&trid=" + trackingID +
13                        "&u=" + user;
14         var sender = function() {
15             if (!document.cookie.includes("uname=" + user))
16                 throw "ERROR";
17             var method = "POST",
18                 apiVer = config.apiVersion,
19                 urlPrefix = baseURL + config.section +
20                             "/api/" + apiVer;
21             $.ajax(urlPrefix + "/" + actionEndpoint +
22                    "?" + tracking, {
23                 method: method,
24                 data: postParams,
25             });
26         }
27         var postParams = {"title": title, "text": postText};
```

```
28          $("#confirm-btn").on("click", sender);
29      }
30      window.__makePostCallback = makePost;
31  })();
32  trackingID = config.trackingSegment + 22293;
```

Листинг 1. Пример кода, трудно поддающегося динамическому анализу

Из данных примеров видно, что динамический анализ обладает рядом ограничений в контексте задачи поиска функций веб-приложения, особенно скрытых.

1.2. Сложность задачи анализа JavaScript-кода

Для построения аппроксимации DEP для JavaScript-кода нужно, как минимум, уметь находить вызовы методов, отправляющих запросы на сервер, и определять возможные наборы значений аргументов этих вызовов. В литературе упоминается ряд трудностей, делающих сложным решение подобных задач для JavaScript-кода [8 – 10]. К ним относятся:

— динамическое выполнение кода с помощью функций `eval`, `Function` и некоторых других;
— динамическая типизация с неявным динамическим приведением типов;
— обращение к полям объектов по динамически вычисленному имени поля;
— динамическое добавление и удаление свойств объектов;
— изменение прототипов встроенных в язык классов, позволяющее изменить семантику встроенных в стандартную библиотеку языка методов;
— работа с функциями как со значениями, в т. ч. присвоение их переменным и полям объектов и передача в качестве аргументов функций;
— динамическое создание областей видимости с помощью выражения `with`.

Эти трудности могут делать анализ особенно сложным в комбинации друг с другом. Например, как отмечается в [11], если в анализируемом участке программы свойство глобального объекта берётся по вычисленному имени и есть вызов взятого значения как функции и если при этом алгоритму анализа не удаётся найти имя свойства и значения аргументов функции (т. е. они остаются неизвестными), то возможны ситуации, когда именем свойства будет строка «`eval`». Тогда последующий вызов приведёт к динамическому выполнению какого-то неизвестного кода. Схожий пример приводится в [10].

Для клиентского JavaScript-кода ещё одной проблемой является взаимодействие анализируемого кода с объектной моделью документа (DOM-моделью) [10] — кроме того, что её моделирование в анализаторе само по себе является трудоёмкой задачей, какие-то части DOM-модели определяются пользователем (например, значение заполненного пользователем поля формы), и в ходе статического анализа эти данные неизвестны, что является принципиальным ограничением для любого метода статического анализа.

Существующие исследования показывают, что осложняющие статический анализ конструкции языка JavaScript существуют не только в теории, но используются в реальных веб-приложениях [4].

1.3. Модельный пример

Рассмотрим следующий модельный пример веб-приложения, в клиентском JavaScript-коде которого присутствует фрагмент из листинга 2. Обработчик `remove(id)` использует библиотеку jQuery для выполнения HTTP-запроса и вызывается при клике

на кнопку удаления сообщения с идентификатором `id`. Обработчик POST-запроса на стороне сервера уязвим к атаке SQL-инъекцией в параметре `ident`.

```
1 var api = "/application/iuT6ei/";
2 function remove(id) {
3     if (prompt("Enter 'yes' to remove") !== "yes") return;
4     $.post(api + "interface/remove/handle", {ident: id});
5 }
```

Листинг 2. Модельный пример DEP

Данный DEP не анализируется динамическими методами, поскольку для выполнения запроса необходимо пройти проверку в условном операторе. Этот же DEP не анализируется статическими методами существующими сканерами безопасности, как показали наши эксперименты (см. п. 4).

2. Существующие работы

Рассмотрим две группы работ, наиболее близкие к решаемой задаче: исследования в области сканеров безопасности динамических веб-приложений и в области статического анализа программ на языке JavaScript, в том числе в контексте задач обнаружения программных уязвимостей.

В 2010 г. Адам Дюпе с коллегами из университета Калифорнии в Санта-Барбаре опубликовали исследование ограничений современных на тот момент сканеров безопасности при работе с динамическими приложениями [12]. Они продемонстрировали, что для обнаружения большинства типов уязвимостей от сканеров требуется качественная поддержка динамического обхода (краулинга) веб-приложений, поддержка исполнения JavaScript, при этом в исследованных сканерах были выявлены ошибки в реализации парсеров HTML, недостаточная поддержка JavaScript, в результате из 16 уязвимостей 8 не были обнаружены ни одним из сканеров. Среди исследованных сканеров были Acunetix, AppScan, Burp, w3af, WebInspect, которые существуют и активно развиваются и в настоящее время. В 2012 г. Адам Дюпе с коллегами опубликовали работу по теме автоматического обнаружения уязвимостей в веб-приложениях [13], а в 2014 г. он защитил диссертацию по той же теме [14]. В диссертации предложена архитектура сканера безопасности для динамических веб-приложений, в которой строится модель состояний анализируемого приложения в виде автомата Мили, где входные символы интерпретируются как запросы сканера, а состояния определяются по ответам. Основной акцент сделан на динамическом краулинге для достижения полноты обхода состояний и никак не затронут вопрос анализа клиентского кода JavaScript.

В эти же годы направление динамического краулинга веб-приложений активно изучается в других группах, в частности авторами IBM AppScan и Crawljax. В [15] исследованы различные стратегии динамического обхода веб-приложений, которые могут быть применимы для полноценного функционального тестирования приложений, т. е. критерием также является полнота покрытия кода серверной части приложения, но при этом обход и анализ выполняются методом «чёрного ящика». Следует отметить работу [16], в которой описан один из широко используемых динамических краулеров — Crawljax. Работа посвящена сложности задачи динамического обхода современных веб-приложений, которые активно используют язык JavaScript на стороне клиентской части приложения. Основная цель работы совпадает с предыдущей: обеспечить полноту функционального тестирования веб-приложений.

Эти работы всецело сфокусированы на задаче динамического краулинга веб-приложений, в том числе в контексте поиска уязвимостей, и не рассматривают статический

анализ клиентского кода JavaScript в качестве альтернативного способа извлечения знаний о структуре серверной стороны веб-приложения. Если поставить перед собой такую задачу, то логичный способ её решения — использование одного из существующих статических анализаторов кода для языка JavaScript, например SAFE, WALA или TAJS. Мы рассмотрели некоторые работы, в которых авторы статических анализаторов или сторонние исследователи пытались применить их для анализа реальных веб-приложений в Интернете.

При сравнительном анализе статических анализаторов для построения графа вызовов по JavaScript-программе отмечено, что далеко не все анализаторы дают адекватный API для доступа к объектам внутреннего представления, в частности SAFE и JSAI по этой причине оказались неприменимы для решаемой задачи [17].

В [18] сделана попытка исследования кода больших интернет-приложений с помощью существующих статических анализаторов JavaScript. Авторы утверждают, что полные (sound) методы анализа неприменимы на практике. Они опробовали схему анализа, которая использует комбинацию двух статических анализаторов — результаты работы быстрых, но неточных и неполных методов анализа из WALA используются далее в качестве начальных данных для более точного контекстно-чувствительного анализа средством SAFE. В экспериментах анализируется код пяти популярных библиотек (jQuery, MooTools, Prototype, YUI, Underscore) и код с заглавной страницы пяти реальных сайтов (live.com, wikipedia.org, facebook.com, youtube.com, baidu.com). Показано, что с анализом успешно справляется только комбинация анализаторов SAFE и WALA. Иными словами, возможность анализа кода реальных веб-приложений достигается только путём заведомого снижения полноты и точности анализа ради его масштабируемости.

В [10] исследована задача повышения контекстной чувствительности анализа циклов для анализатора SAFE на примере корпуса реальных приложений. Эксперименты проводились на библиотеках jQuery, Modernizr, BootStrap, Mootools и Prototype, а также на JavaScript-коде с главных страниц сайтов google.com, facebook.com, youtube.com, baidu.com и yahoo.com. Авторам удалось показать улучшение на части кода библиотек, но для реальных сайтов анализ оказался неприменим, время работы программы составило в среднем 3500–7000 с.

Отдельно следует упомянуть работу [19], в которой предложен способ проверки соответствия API-запросов в коде на JavaScript спецификации соответствующего серверного API. Для этого предлагается при помощи статического анализа JavaScript извлекать из точек обращения к API соответствующие URL, метод и параметры запросов и сопоставлять спецификации в формате OpenAPI/Swagger для проверки корректности использования API в клиентском коде. Авторы использовали построение упрощённого field-based CFG, идентификацию точек обращения к API лексическим поиском конструкций jQuery и определение параметров запроса с помощью построения обратных программых срезов с анализом потока данных средствами WALA. Тестирование показало применимость в том числе для приложений, которые используют jQuery. Однако следует понимать, что из-за специфики анализа (отсутствия чувствительности к путям и контексту) параметры запроса получались как комбинация всевозможных значений различных переменных вдоль разных путей в программе, формирующих данный запрос. На практике это ведёт к большому количеству возможных значений параметров для каждого из запросов, что не критично в контексте решаемой авторами задачи, но может быть сильным ограничивающим фактором для задачи поиска уязвимостей.

Использование полноценных статических анализаторов JavaScript-кода представляется весьма перспективным направлением развития сканеров безопасности веб-приложений, но, к сожалению, в настоящее время их использование для этой задачи представляется непрактичным, в первую очередь из-за большой вычислительной сложности и в некоторых случаях недостаточной точности анализа.

3. Метод анализа JavaScript-кода для поиска DEP

Как уже было сказано, существующие средства анализа JavaScript-кода, как встроенные в сканеры безопасности, так и самостоятельные, плохо справляются с задачей поиска DEP на страницах современных веб-приложений. Опишем новый метод анализа JavaScript-кода с целью поиска серверных точек взаимодействия в приложениях. В п. 4 приведены результаты его сравнения с существующими средствами. Метод обнаруживает только точки взаимодействия, порождаемые вызовами JavaScript-функций.

Метод работает над веб-страницей: он принимает на вход URL страницы, её HTML-разметку и набор подключаемых этой разметкой внешних ресурсов. Результатом является множество спецификаций отправляемых на сервер запросов, найденных в ходе анализа. Это множество считается аппроксимацией DEP, к которым обращается JavaScript-код страницы.

Метод анализа статический, нечувствительный к управляющим конструкциям (*path-insensitive*), однако чувствительный к порядку инструкций (*flow-sensitive*). Делается также попытка в простых случаях определить значения передаваемых в вызываемые функции аргументов с помощью анализа цепочки вызовов, что добавляет некоторую чувствительность к контексту вызова (*context-sensitivity*).

Концептуально метод работает следующим образом: производится поиск вызовов функций, отправляющих запросы на сервер (далее для краткости будем называть их *AJAX-функциями*), и определение аргументов этих вызовов. Затем для каждого найденного вызова с данным набором аргументов определяется, какой вид имеют запросы, сделанные таким вызовом, — формируются спецификации запросов.

Метод работает над абстрактным синтаксическим деревом кода (Abstract Syntax Tree, AST). Поиск вызовов делается с помощью рекурсивного обхода дерева в глубину; вызовы обнаруживаются простым сигнатурным методом — проверяется вхождение имени вызываемой функции (и, если это вызов метода объекта, имени объекта) в заранее заданный список имён (*набор сигнатур*). В этот список входят, например, имена `fetch` и `axios`, пары `($, ajax)` и `($http, post)`.

3.1. Моделирование переменных

Метод корректно моделирует области видимости переменных, используя *модель лексических областей видимости*. Это отображение, сопоставляющее каждому использованию идентификатора — имени переменной в программе — специальное значение, уникальное для переменной (*binding*). Оно одинаково для идентификаторов, относящихся к одной и той же переменной, и разное для идентификаторов, относящихся к разным переменным, в т. ч. в случае, когда имена совпадают. Модель учитывает границы областей видимости, перекрытия одноимённых переменных и вложенные области видимости. В условиях отсутствия конструкций `eval` и `with` (которые могут добавлять имена в области видимости динамически) задача построения подобной модели является достаточно простой и может быть решена статически. В предлагаемой реализации метода для её решения используется библиотка Babel [20]. В дальнейшем, когда мы будем говорить о *переменной*, будем иметь в виду уникально идентифицируемую с помощью модели лексических областей видимости переменную.

Для моделирования значений переменных используется *модель памяти* — отображение, ставящее в соответствие переменным значения. Каждой переменной может быть поставлено в соответствие только одно значение. Это ограничение приводит к тому, что некоторые элементы семантики языка JavaScript не могут быть промоделированы алгоритмом корректно, так как при выполнении JavaScript-программ одной и той же с точки лексики переменной могут соответствовать несколько значений. Примерами являются локальные переменные и аргументы рекурсивных функций, а также переменные, попавшие в замыкание вложенной функции. Для всех таких случаев модель памяти будет содержать для каждой переменной только какое-то одно значение, что является ограничением описываемого метода.

3.2. Вычисление выражений

Метод использует механизм вычисления JavaScript-выражений. Он поддерживает выражения, содержащие литералы, переменные, операцию +, доступ к полю объекта, вызовы некоторых встроенных в браузер функций и ещё некоторые операции. При использовании в выражениях переменных делается попытка взять их значения из модели памяти.

В случае, если конкретное значение выражения вычислить не удалось (например, используется переменная, для которой в модели памяти нет значения, или в выражении используется неподдерживаемая анализатором конструкция), вырабатывается *неизвестное значение* — `Unknown`. Особым случаем является использование формального аргумента функции, значение которого неизвестно — при этом вырабатывается значение `FromArg`, которое является подвидом неизвестного значения, но дополнительно сигнализирует о том, что данные пришли именно из формального аргумента функции. Это используется для того, чтобы принять решение о необходимости анализа с учётом цепочек вызова.

Вычисление выражений делается следующим образом. Для большинства операций, если все участвующие в выражении значения конкретны, то выполняется соответствующая операция в оригинальной семантике JavaScript.

Если в выражении участвуют значения `Unknown` или `FromArg`, то при простом присваивании переменной или полю объекта и при передаче в качестве аргумента функции они остаются без изменений, а в остальных случаях либо приводятся к конкретным значениям (так делается при операциях, в ходе которых значение приводится к строке), либо результат всего выражения становится `Unknown` (или `FromArg`, если это значение `FromArg` участвовало в вычислении). При приведении к конкретным значениям значение `Unknown` становится строкой «`UNKNOWN`», а значение `FromArg` — строкой «`FROM_ARG`». В конце работы алгоритма все вхождения `FromArg` в результирующих данных заменяются на `Unknown`, а все вхождения строки «`FROM_ARG`» — на «`UNKNOWN`».

Ключевое слово `this` текущей версией алгоритма не поддерживается — при его обработке будет выработано значение `Unknown`.

3.3. Алгоритм анализа

Алгоритм анализа (алгоритм 1) получает на вход URL-адрес страницы, AST-дерево содержащегося в ней JavaScript-кода и модель лексических областей видимости. AST-дерево можно получить конкатенацией текстов всех JavaScript-программ на странице и последующим синтаксическим разбором полученной программы с помощью библиотеки Babel [20]. Модель лексических областей видимости также может быть получена с помощью Babel на основе AST-дерева. Результатом анализа является набор пар, описывающих найденные вызовы и их аргументы. Каждая пара состоит из сигнатуры

функции, вызов которой был найден, и списка аргументов её вызова. Этот набор будет далее переработан в набор спецификаций HTTP-запросов, как описано в п. 3.4.

Алгоритм 1. Анализ

Вход: `PageURL`, `AST`, `ScopeModel`.
Выход: `CallDescrs`.

```
1: MemModel := SeedInitialMemModel(PageURL).
2: MemModel := GatherVarValues(AST, MemModel, ScopeModel).
3: Queue := [].
4: CallDescrs, Queue := ExtractDEPs(AST, MemModel, ε, Queue).
5: Пока |Queue| > 0:
6:    вытолкнуть первый элемент из Queue в (callerAST, Chain);
7:    newCallDescrs, Queue := ExtractDEPs(callerAST, MemModel, Chain, Queue);
8:    CallDescrs := CallDescrs ∪ newCallDescrs.
```

Основными шагами анализа являются поиск значений переменных, которому в псевдокоде соответствует функция GatherVarValues (алгоритм 2), и поиск вызовов AJAX-функций и их аргументов, которому соответствует код в строках 3–8 алгоритма 1. Его основная часть, включая непосредственно обход AST для поиска интересующих мест вызова и вычисление их аргументов, содержится в функции ExtractDEPs (её псевдокод приведён в алгоритме 3).

Алгоритм 2. Поиск значений переменных и присваивание значения

```
1: Функция Assign(v, MemModel)
2: Если v это VariableDeclarator и ∃ v.init и v.id это Identifier, то
3:    b := GetBinding(v.id, ScopeModel);
4:    value := EvalExpr(v.init, MemModel, ScopeModel);
5:    MemModel := MemModel ∪ (b ↦ value),
6: иначе если v это AssignmentExpression, то
7:    value := EvalExpr(v.right, MemModel, ScopeModel).
8:    Если v.left это Identifier, то
9:       b := GetBinding(v.left, ScopeModel);
10:      MemModel := MemModel ∪ [b ↦ value],
11:   иначе если v.left это MemberExpression, то
12:      AssignProp(v.left, value).
13: Вернуть MemModel.
14: Функция GatherVarValues(AST, MemModel, ScopeModel)
15: Для всех вершин v ∈ AST в порядке обхода в глубину:
16:   Если v это VariableDeclarator или v это AssignmentExpression, то
17:      MemModel := Assign(v, MemModel),
18:   иначе если v это FunctionDeclaration, то
19:      b := GetBinding(v.id, ScopeModel);
20:      MemModel := MemModel ∪ [b ↦ FuncVal(v)].
21: Вернуть MemModel.
```

Алгоритм 3. Поиск вызовов и их аргументов

```
1: Функция ExtractDEPs(AST, MemModel, Chain, Queue)
2: d := 0.
3: Для всех вершин v ∈ AST в порядке обхода в глубину:
4:   Если v это CallExpression, то
5:     Если ∃ sign ∈ SignatureSet: MatchesSignature(v, sign), то
6:       vals := [EvalExpr(arg, MemModel, ScopeModel) | arg ∈ v.arguments];
7:       vals, hadArgsDependency := CheckAndRemoveFromArg(vals);
8:       Results := Results ∪ {(sign, vals)}.
9:       Если hadArgsDependency, то
10:        Queue := BuildCallChains(v, AST, Queue, ScopeModel, MemModel).
11:    Если |Chain| > 0 и v.callee это Identifier, то
12:      (binding, ast, args), rest := Chain.
13:      Если binding = GetBinding(v.callee, ScopeModel), то
14:        act, MemModel := SetActualArgVals(func, args, MemModel);
15:        newResults, newQueue := ExtractDEPs(ast, MemModel, rest, Queue);
16:        Results := Results ∪ newResults;
17:        Queue := Queue + newQueue;
18:        MemModel := MemModel \ act,
19:  иначе если v это Function то
20:    d := d + 1;
21:    formal := {(GetBinding(a, ScopeModel) ↦ FromArg) | a ∈ v.arguments};
22:    MemModel := MemModel ∪ formal,
23:  иначе если d > 0 и v это {VariableDeclarator, AssignmentExpression}, то
24:    MemModel := Assign(v, MemModel).
25:  Если вышли из вершины v′ и v′ это Function, то
26:    d := d − 1;
27:    formal := {(GetBinding(a, ScopeModel) ↦ FromArg) | a ∈ v.arguments};
28:    MemModel := MemModel \ formal.
29: Вернуть Results, Queue.

30: Функция SetActualArgVals(func, args, MemModel)
31: vals := [EvalExpr(arg, MemModel, ScopeModel) | arg ∈ args];
32: act := {(GetBinding(a_i, ScopeModel) ↦ vals_i) | 0 ⩽ i < |func.arguments|};
33: MemModel := MemModel ∪ act.
34: Вернуть act, MemModel.

35: Функция BuildCallChains(v, AST, Queue, ScopeModel)
36: funcAST := GetCallerAST(v);
37: funcBindings := GetBindingsForAST(funcAST, ScopeModel, MemModel).
38: Для всех binding ∈ funcBindings:
39:   callSites := FindCallSites(AST, binding, ScopeModel).
40:   Для всех site ∈ callSites:
41:     chain := [(binding, funcAST, funcAST.arguments)] + Chain;
42:     callerAST := GetCallerAST(site);
43:     Queue := Queue + [(callerAST, chain)].
44: Вернуть Queue.
```

Можно заметить, что функция GatherVarValues учитывает только присваивания переменных, инициализации и объявления функций. Другие способы передачи значений, например задание явно аргументов самовызывающейся функции (IIFE), обработаны на этом этапе не будут.

Функция ExtractDEPs работает следующим образом: делается рекурсивный обход AST-дерева, при котором вычисляются выражения, при обнаружении присваиваний или инициализаций переменных на неглобальном уровне (т. е. внутри тела какой-то функции) они обрабатываются аналогично GatherVarValues, пополняя модель памяти. При обнаружении вызова функции, отвечающего одной из имеющихся сигнатур AJAX-функции, выражения, задающие аргументы вызова, вычисляются и набор аргументов вместе с сигнатурой запоминается и становится частью результата работы функции. При этом проверяется, встречаются ли в вычисленных аргументах значения `FromArg` или «`FROM_ARG`». Если это так, то регистрируется зависимость передаваемых в AJAX-вызов данных от формальных аргументов вызывающией функции. Обход AST-дерева делается рекурсивно в глубину. Все вершины посещаются при одном обходе только один раз, циклы и ветвления не влияют на него (т. е. тело цикла будет посещено единожды, обе ветви оператора `if` будут посещены безусловно).

Как видно в псевдокоде, функция ExtractDEPs вызывается в алгоритме несколько раз. Это делается для анализа с учётом цепочек вызова — в случае, если обнаружено, что параметры интересующего AJAX-вызова зависят от формальных аргументов функции, делается попытка найти места её вызова и выполнить анализ, начиная с каждого из этих мест (точнее, от начала тела функции, в которой место вызова найдено). В общем случае аргументы функции в месте вызова могут зависеть от формальных аргументов другой функции, тогда будет выполнен поиск, в свою очередь, места её вызова (и, таким образом, будут анализироваться уже цепочки из двух вызовов), и так далее. Самый первый вызов ExtractDEPs в строке 4 алгоритма 1 осуществляет анализ без учёта цепочек вызова, дальнейшие вызовы в строке 7 делаются по одному на каждую выявленную цепочку.

Функция GetBinding получает из модели лексических областей видимости уникальное значение, идентифицирующее переменную, по месту её использования (см. п. 3.1).

Функция EvalExpr вычисляет выражение в соответствии с механизмом вычисления, описанном в п. 3.2. Функция AssignProp принимает значение и вершину AST типа MemberExpression (такие вершины соответствуют обращению к полю объекта), вычисляет объект и имя свойства и добавляет в этот объект свойство с таким именем и данным значением (или заменяет существующее, если свойство с таким именем уже было). Функция MatchesSignature осуществляет сопоставление места вызова с сигнатурой, возвращает значение «истина», когда тип сигнатуры совпадает с типом вызова (сигнатура может быть именем свободностоящей функции или парой из имени метода и имени объекта) и имя в сигнатуре совпадает с именем в вызове (в случае с вызовом метода — имена объекта и метода), в остальных случаях функция возвращает значение «ложь».

Функция GetCallerAST принимает на вход вершину AST-дерева и, если эта вершина соответствует коду внутри функции, возвращает корень поддерева, соответствующего телу этой функции, иначе — корень всего AST-дерева программы.

Функция GetBindingsForAST принимает на вход AST-дерево, соответствующее телу функции, и возвращает набор всех переменных, которым в модели памяти соответствуют функциональные значения этой функции. Эта функция работает с текущим

состоянием модели памяти — на тот момент, когда она была вызвана. В это время она наполнена в результате работы GatherVarValues, а также делавшихся до этого вызовов ExtractDEPs. Делается обход всех пар «ключ–значение», содержащихся в модели памяти, и выбираются те ключи, которым соответствует функциональное значение для функции, чьё AST-дерево подано на вход GetBindingsForAST. Поскольку в модели памяти каждой переменной в один момент времени может соответствовать только одно значение, если в ходе работы программы одной переменной присваивались разные значения, включая интересующее, эта переменная может не найтись и не попасть в результат (так как на момент вызова GetBindingsForAST в модели памяти этой переменной соответствует другое значение).

Функция FindCallSites для данной переменной возвращает набор всех вершин — мест вызова, где в качестве вызываемой функции используется эта переменная. Это делается с помощью модели лексических областей видимости и обхода AST-дерева программы: делается обход AST и для каждого места вызова, где вызываемая функция задана идентификатором, из модели лексических областей видимости берётся переменная, соответствующая этому идентификатору. Если она совпадает с переменной, поданной на вход функции FindCallSites, то рассматриваемое в данный момент место вызова добавляется в список — результат работы функции. Эта функция работает только с местами вызова, где функция дана идентификатором (`f(x, y)`). Если вызываемая функция задана обращением к полю объекта (т. е. это вызов метода, например `o.func(x)`) или другим выражением, то такое место вызова будет проигнорировано. В том числе не будут найдены вызовы самовызывающихся функций.

Функция CheckAndRemoveFromArg выполняет описанное в п. 3.2 удаление значений `FromArg` — она возвращает версию набора аргументов с удалёнными вхождениями `FromArg` (и строки «`FROM_ARG`»), а также булево значение — признак того, были ли такие значения найдены (и удалены) среди аргументов.

3.4. Формирование спецификаций DEP на основе найденных вызовов

Какие HTTP-запросы будут отправлены тем или иным вызовом с некоторым набором аргументов, мы определяем с помощью набора эвристических моделей функций, отправляющих запросы на сервер. Они моделируют работу поддерживаемых библиотек и встроенных в браузер механизмов в части отправки запросов на сервер, т. е. для каждой функции, для которой реализация метода содержит сигнатуру вызова, реализация также содержит написанный вручную код, который на основе аргументов этого вызова и исходного URL-адреса страницы определяет, какой запрос будет отправлен в результате этого вызова.

В результате применения этих сигнатур к набору найденных вызовов и их аргументов получается множество спецификаций запросов. Каждая спецификация содержит следующие поля: имя метода, URL-адрес, список HTTP-заголовков, тело запроса. Это множество является конечным результатом работы метода.

Следует отметить, что полученные спецификации в общем случае не соответствуют серверным точкам взаимодействия веб-приложения один-в-один. Два разных вызова отправляющей запрос на сервер функции могут обращаться к одной и той же серверной точке взаимодействия; также один и тот же вызов может обращаться к нескольким серверным точкам взаимодействия в зависимости от значений параметров — например, значение параметра query string в URL HTTP-запроса может влиять на маршрутизацию по микросервисам. В результате запросы, соответствующие нескольким разным

спецификациям в выводе метода, могут относиться к одной серверной точке взаимодействия, и наоборот, одной выходной спецификации могут соответствовать запросы к разным точкам взаимодействия. Как было сказано, при поиске серверных точек взаимодействия наиболее важным критерием качества является полнота. С точки зрения этого критерия первый из описанных эффектов является несущественным — в результате него один и тот же DEP будет повторен несколько раз. Второй, однако, может привести к проблемам, так как часть запроса, относящаяся к фиксированной части DEP, может быть ошибочно отнесена к вариативной. Заметим, что полностью решить эти проблемы при анализе веб-приложения как чёрного ящика в общем случае невозможно — множество допустимых DEP определить полно и точно нельзя, так как недоступна полная информация о серверной части веб-приложения.

3.5. Иллюстрация работы метода на примерах

Покажем, как работает алгоритм на двух примерах JavaScript-кода.

Сначала рассмотрим код листинга 1. Он содержит вызов функции `$.ajax`, отправляющий запрос на сервер. Этот пример имеет вид, характерный для реального JavaScript-кода на веб-страницах. Запрос делается функцией библиотеки jQuery, использование которой, в том числе для отправки запросов на сервер, чрезвычайно распространено. Кроме того, здесь используются вспомогательные функции-обёртки для отправки запросов, а основная часть кода «завёрнута» в анонимную самовызывающуюся функцию — это также характерно для клиентского JavaScript-кода в Интернете. Ещё одной часто встречающейся в реальном коде особенностью является использование глобальных переменных для задания конфигурации. Вместе с тем этот код представляет сложность для динамического анализа: чтобы управление дошло до вызова `$.ajax`, небходимо, чтобы последовательно произошли вызовы функций `makePost` и `sender`, а также чтобы прошла проверка в строке 15.

Чтобы определить, какой вид будет иметь запрос, анализатору необходимо как можно точнее определить возможные аргументы вызова `$.ajax`, находящегося в строке 21. Эти аргументы задаются как выражения, зависящие от переменных, некоторые из которых заданы выражениями от других переменных. Переменная `urlPrefix` (от которой зависит первый аргумент вызова) задаётся выражением, которое зависит от локальной переменной `apiVer`, а также глобальных переменных `baseURL` и `config`. Выражение, задающее первый аргумент вызова, также зависит от переменных, находящихся в области видимости объемлющей функции `makePost`. В анализаторе есть механизм вычисления таких выражений (см. п. 3.2).

Можно заметить, что `trackingID` и `postParams` в коде задаются после использования. Тем не менее при реальном выполнении они будут использованы уже после того, как получат свои значения — функция `makePost`, в которой используется переменная `trackingID`, может быть вызвана в произвольный момент после того, как присваивание `trackingID` в строке 29 будет выполнено. Функция, в которой используется переменная `postParams`, также может быть вызвана в произвольный момент после того, как будет завершён вызов `makePost`, в котором она задаётся. Алгоритм анализа способен использовать значения этих переменных при вычислении зависящих от них выражений благодаря фазе поиска значений переменных (в псевдокоде ей соответствует функция GatherVarValues, см. алгоритм 2) — на этой стадии при обходе AST-дерева программы будут обработаны присваивания переменных, в том числе инициализация переменных `baseURL`, `config` и `pageInfo`, инициализация `postParams` и присваивание `trackingID`. При этом заметим, что даже объявление переменной `postParams` находит-

ся в коде ниже места её использования. Тем не менее, по правилам языка JavaScript, переменная, объявленная с ключевым словом `var`, находится в области видимости во всём теле функции. Для переменной `trackingID` места объявления в программе вообще нет — в результате при присваивании в строке 32 будет создана глобальная переменная с таким названием. Обе эти особенности учитываются алгоритмом благодаря корректному моделированию с использованием модели лексических областей видимости.

Метод поддерживает специальную переменную `location`, встроенную в браузер и предназначенную для работы с URL-адресом страницы. Напомним, URL-адрес страницы является частью входных данных алгоритма. Для данных примеров кода будем считать, что URL-адрес страницы — это `http://example.com/`. Таким образом, возможно вычисление значения `pageInfo`, это будет строка «`type=forumPage&page=http://example.com/`».

После шага поиска значений переменных будет выполнен шаг обхода AST-дерева с целью поиска вызовов AJAX-функций и их аргументов. В псевдокоде этот обход выполняется в функции ExtractDEPs (см. алгоритм 3). На этом шаге присваивания переменных, находящиеся внутри функций, также обрабатываются (строки 23–24 алгоритма 3). Таким образом, в ходе этого шага будут вычислены значения `actionEndpoint`, `tracking`, `method`, `apiVer`, `urlPrefix` и, наконец, аргументы вызова `$.ajax`. При этом вычисляемые значения зависят от переменных, значения которых неизвестны, — формальных аргументов функции `makePost`. В качестве их значений будут взяты `FromArg`. Они допускают конкатенацию с конкретными строками, причём как в качестве префикса, так и суффикса, при этом информация о конкретной строке не теряется, при конкатенации неизвестное значение приводится к строке. В данном примере это позволяет анализатору найти имена параметров запроса.

Для определения того, какой вид будет иметь HTTP-запрос, отправленный вызовом функции `$.ajax` с такими аргументами, будет использована модель библиотеки jQuery. В итоге определим, что будет отправлен POST-запрос на URL-адрес `http://example.com/site/forum/api/1.1/post.php?type=forumPage&page=http://example.com/&trid=374022293&u=UNKNOWN` с телом `title=UNKNOWN&text=UNKNOWN`. Заметим, что в финальном результате нет вхождений «`FROM_ARG`», они заменены на «`UNKNOWN`», как описано в п. 3.2.

Обнаружение аргументов вызова `$.ajax` в данном примере является нетривиальной задачей для динамического анализа — необходимо определить, что нужно вызвать сначала `makePost`, а затем `sender`. При выполнении `sender` нужно добиться, чтобы прошла проверка в строке 15, зависящая от аргумента `makePost`. Для статического анализа проблема прохождения проверки гораздо менее существенна, описываемый алгоритм анализа её просто проигнорирует.

Теперь рассмотрим, как будет обработан код на листинге 3.

```
1  var l = "ipsum";
2  function i(x, y, z) {
3      $.post("/action/endpoint.php?action=" + x + "&id=" + y, {
4          lorem: l,
5          dolor: "sit amet",
6          sit: z
7      });
8  }
9  var j = function(tok1, tok2) {
10     var xj = "abc",
```

```
11          zj = "[" + tok2 + "]";
12      i(xj, tok1, zj);
13  }
14  function k() {
15      j("123", "amet");
16  }
```

Листинг 3. Второй пример кода, иллюстрирующий работу алгоритма

Этот пример отличается от предыдущего тем, что он содержит вызовы функции, внутри которой находится AJAX-вызов. При этом также есть зависимость параметров запроса от аргументов этой функции. В результате, анализируя места её вызова и то, какие аргументы там передаются, можно более точно определить вид отправляемого на сервер HTTP-запроса.

Пример характерен для кода реальных приложений, при этом он уже представляет сложную задачу для статического анализа, так как для передачи данных через вызовы функций требуется межпроцедурный анализ.

В ходе анализа на этапе поиска вызовов AJAX-функций будет выявлено, что аргументы вызова `$.post` зависят от формальных аргументов функции `i` (соответствующая проверка делается в строках 7 и 9 алгоритма 3). Поэтому далее будет произведён поиск вызовов функции `i` (см. функцию BuildCallChains в алгоритме 3). Это именованная функция, объявленная в глобальной области видимости, в строке 12 она вызывается по своему имени `i` — это место вызова будет обнаружено. В результате будет проделан ещё один обход AST-дерева для поиска вызовов AJAX-функций, который будет начат с корня AST-дерева, соответствующего телу функции, находящейся в строках 9–13 (ему соответствует вызов ExtractDEPs в строке 7 алгоритма 1). При обработке вызова `i` обход перейдёт на дерево, соответствующее коду `i` (в псевдокоде это происходит в строке 15 алгоритма 3), в результате чего анализ её тела будет повторен, но на этот раз с уточнёнными значениями формальных аргументов. В результате будет использовано значение формального аргумента `x` функции `i` (это строка «abc»). Однако значения `y` и `z` будут оставаться неизвестными — точнее, значение `y` будет частично определённым («`UNKNOWN`»), значение `z` — полностью неопределённым. Они также будут зависеть от формальных аргументов функции — на этот раз от аргументов функции в строках 9–13. Хотя эта функция анонимная, она присваивается переменной `j`, что будет обнаружено на этапе поиска значений переменных, в результате её вызов по имени `j` можно будет найти в строке 15. Обход AST-дерева для поиска вызовов AJAX-функций будет проделан в третий раз, на этот раз начиная от тела функции `k`. В ней в вызов `j` передаются конкретные аргументы, что в конечном итоге позволит полностью вычислить аргументы функции `$.post`. Для первого аргумента это будет значение `"/action/endpoint.php?action=abc&id=123"`, для второго — значение `{"lorem": "ipsum"dolor": "sit amet"sit": "[amet]"}`.

4. Экспериментальное сравнение метода с другими

Для оценки эффективности предложенного метода мы провели сравнительное экспериментальное исследование его прототипной реализации и механизмов поиска серверных точек взаимодействия, которыми располагают существующие сканеры безопасности веб-приложений. В эксперименте участвовали следующие сканеры безопасности:

— PT BBS — Positive Technologies BlackBox Scanner [21];
— Acunetix [22];
— Detectify [23];

— Burp Scanner [24];
— HCL AppScan Cloud [25].

Для проведения эксперимента было подготовлено модельное веб-приложение.

4.1. Описание модельного веб-приложения

Будем говорить, что *веб-приложение содержит точку ввода данных* (*DEP*) или *в веб-приложении есть точка ввода данных* (*DEP*), если это приложение реализует некоторую функцию на стороне сервера, которая принимает все HTTP-запросы, соответствующие DEP.

Используемое в эксперименте модельное веб-приложение является синтетическим и содержит на серверной стороне несколько DEP, обращения к которым есть на клиентской стороне (т. е. на веб-страницах приложения).

С точки зрения серверного кода, все DEP устроены одинаково — они делают SQL-запрос в базу данных и выдают ответ на него, причём их реализация содержит тривиальную SQL-инъекцию: в SQL-запрос подставляется значение параметра HTTP-запроса от клиента, которое никак не фильтруется. Исключением являются два DEP, у которых серверный код, помимо описанного, содержит также проверку *контрольного* параметра запроса — при несовпадении его значения с фиксированной константой (*правильным* значением) обработка запроса прерывается с ошибкой, запрос не делается. При этом у каждого из этих DEP URL-адрес запроса и имена параметров (включая имя уязвимого параметра) являются случайными — URL содержит подстроки из не менее чем 12 случайных символов (из набора английских букв в верхнем и нижнем регистрах, а также цифр от 0 до 9), имена параметров также являются строками из 6 таких символов. Это делает подбор правильной комбинации URL и имён параметров невозможным на практике. Пример URL-адреса, соответствующего одному из DEP модельного веб-приложения: `http://test.stand/application/jie8Ye/interface/aesi9X/handle?Po3oom=1`. Пример JavaScript-кода со страницы модельного веб-приложения, отправляющего POST-запрос на сервер, приведён в листинге 4.

```
1 var api = "/application/iuT6ei/";
2
3 function request7() {
4     $.post(api + "interface/Eek0Mu/handle", {"eeNgi6": "1"});
5 }
```

Листинг 4. Код, отправляющий POST-запрос

Кроме URL-адреса и имён параметров, DEP различаются HTTP-методами и способом передачи параметров. Все обращения к DEP находятся либо на главной странице модельного веб-приложения, либо на страницах, на которые с главной страницы ведут прямые ссылки. Исходный код стенда доступен по ссылке `https://github.com/seclab-msu/js-dep-mining-test-app`.

То, какой вид имеют находящиеся на клиентской стороне обращения к DEP модельного веб-приложения, описано в следующем списке:

1) Запрос делается при загрузке страницы вызовом `$.ajax` в теге `<script>`. Все аргументы вызова фиксированы и заданы в месте вызова константными литералами. Вызов находится на глобальном уровне, не в функции или условном операторе, поэтому запрос отправится при загрузке страницы, при обработке тега `<script>`, в котором находится вызов.

2) То же, что 1, но запрос делается при нажатии на кнопку (обработчиком события `onclick` тега `button`).
3) То же, что 1, но запрос делается при нажатии на кнопку обработчиком, зарегистрованным с помощью `addEventListener`.
4) То же, что 1, но запрос делается из JavaScript-функции, которая нигде не вызывается (скрытая функция).
5) Запрос делается вызовом функции `$.post`, находящимся в теле другой функции, причём URL-адрес запроса (задаваемый первым аргументом) сформирован как конкатенация значений глобальной JavaScript-переменной и строкового литерала. Эта глобальная переменная объявлена и инициализирована в том же скрипте выше по коду, ей при инициализации присвоено константное значение (строковый литерал). Параметры запроса (определяемые вторым аргументом вызова) заданы полностью константным литералом.
6) Запрос делается вызовом функции `$.ajax`, находящимся в теле другой функции. В качестве аргументов вызова, опередеяющих URL-адрес и параметры запроса, переданы локальные переменные, которые объявлены и инициализированы в той же функции, при инициализации им присвоены константные значения (строковые литералы).
7) Запрос делается вызовом функции. В запросе передаются два параметра. Значение первого фиксировано прямо в месте вызова (задано строковым литералом), этот параметр подставляется на сервере в SQL-запрос (через него возможна эксплуатация уязвимости). Второй параметр является *контрольным*. Его значение задается глобальной переменной, которая объявляется в одном месте и присваивается в другом, принимая несколько возможных значений. Во всех этих присваиваниях значения, которые присваиваются, — константные (заданы строковыми литералами), одно из них является *правильным*.
8) Запрос делается вызовом функции, параметры которого зависят от переменных, имена которых дублируются в разных перекрывающихся областях видимости.
9) То же, что 7, но значение *контрольного* параметра и иных параметров запроса при этом передаётся через цепочку вложенных функциональных вызовов.
10) Запрос делается вызовом функции, URL-адрес запроса берётся с помощью операции доступа к полю объекта из объекта — глобальной переменной, которая задана на странице через объектный литерал.
11) То же, что 10, но глобальная переменная-объект создаётся не объектным литералом, а посредством `new Object`, после чего её свойства заполняются с помощью присваиваний полей.
12) Запрос делается вызовом функции `$.get`, вызов находится в теле функции. Объект, опередеяющий набор параметров запроса (передаваемый во втором аргументе вызова), создаётся с помощью `new Object`, после чего его поля заполняются с помощью присваиваний полей. Создание этого объекта и присваивание полей делается в той же функции, что и вызов `$.get`, непосредственно перед местом вызова.
13) Запрос делается JavaScript-кодом в теге `<script>`, которого изначально нет в HTML-разметке страницы, этот фрагмент разметки скачивается с сервера JavaScript-кодом и добавляется на страницу с помощью вызова jQuery `$(...).html(...)` во время загрузки страницы.
14) Для отправки запроса используется цепочка вызовов, одна функция передаёт аргументы в другую, а та делает отправляющий запрос вызов. Все объявления

функций завёрнуты в анонимную самовызывающуюся функцию. Значения параметров берутся из локальных переменных объемлющей функции. URL-адрес запроса приходит из глобальной переменной-конфига.

15) POST-запрос, отправляемый вызовом функции, для формирования параметров которого используется `FormData`, так что получается multipart-запрос.
16) Запрос делается вызовом функции, часть параметров которого (определяющих URL-адрес) получается парсингом `location`.
17) Запрос делается кодом, активно использующим современные возможности языка JavaScript: `let` и `const`, arrow-функции и конструкцию `try..catch` без переменной у `catch`.

4.2. Методика эксперимента

Для проведения эксперимента модельное веб-приложение было развернуто на сервере, после чего по очереди запускались все участвующие в сравнении сканеры. На стороне приложения записывались все поступающие от сканеров HTTP-запросы, включая заголовки и тела запросов, причём для каждого сканера собирался отдельный журнал.

Для проверки того, что DEP обнаружен сканером, использован следующий критерий:

— если в отчёте о сканировании была запись о наличии уязвимости с URL и именами параметров, соответствующими реально существующему DEP, то мы считали, что DEP обнаружен;
— иначе мы проверяли, есть ли в журнале запросов попытки эксплуатации с соответствующими DEP URL-адресом и набором параметров. Попыткой эксплуатации является любой запрос со значениями параметров, отличных от тех, которые заданы на странице статически;
— если ни одно из условий выше не выполнялось, считалось, что DEP не обнаружен.

Такой способ проверки выбран по той причине, что сканеры, как правило, не предоставляют в явном виде информацию об обнаруженных DEP, а также для некоторых из них не удалось получить список найденных уязвимостей (удалось лишь получить информацию об их количестве).

Для реализованного прототипа предложенного метода информация о найденных DEP может быть получена непосредственно, поэтому для него результаты получены иначе: произведён анализ каждой из страниц модельного веб-приложения, после чего взято объединение множеств описаний DEP, найденных на всех страницах, и для каждого из искомых DEP проверено, входит ли соответствующее ему описание в полученное множество.

4.3. Результаты эксперимента

В таблице приведены результаты экспериментального исследования — количество связанных с уязвимостями точек ввода данных, найденных сканерами безопасности и прототипной реализацией предложенного метода анализа, которая обозначена как *Прототип*.

Отметим, что только сканеры Acunetix и HCL AppScan Cloud смогли успешно обнаружить часть наиболее тривиальных динамически создаваемых DEP и корректно определить их параметры. Таким образом, все уязвимости, DEP и параметры которых определяются в рамках динамического исполнения JavaScript, не могут быть успешно обнаружены большинством существующих сканеров безопасности.

Прототипная реализация предложенного метода анализа успешно обнаружила все DEP, связанные с уязвимостями модельного приложения.

№ DEP	1	2	3	4–17
PT BBS				
Acunetix	✓	✓	✓	
Detectify				
Burp Scanner				
HCL AppScan	✓	✓	✓	
Прототип	✓	✓	✓	✓✓✓

Заключение

Поиск серверных точек ввода данных является важным этапом работы сканера безопасности веб-приложений в модели «чёрного ящика», так как его результаты определяют, какие функции приложения будут проанализированы сканером, а какие нет.

В данной работе предложен метод повышения полноты поиска серверных DEP за счёт статического анализа клиентского JavaScript-кода. Алгоритм выявляет в клиентском коде функции, которые порождают запросы на сервер, и использует статический анализ, работающий над AST-деревом клиентского кода для определения возможных значений параметров этих функций. Найденные функции в дальнейшем позволяют выполнять динамический анализ серверной части приложения, например с помощью фаззинг-тестирования.

Отметим, что задача поиска серверных точек ввода данных отличается от других задач анализа безопасности клиентской стороны веб-приложения. Например, для поиска DOM-based XSS клиентский JavaScript-код, который никогда не выполняется на странице (т. е. «мёртвый код»), не представляет интереса — даже если он содержит уязвимость, её невозможно эксплуатировать. В то же время для получения информации о сервере информация о запросах из мёртвого кода может быть полезна, но такой код можно проанализировать только статически.

Для проведения экспериментов мы реализовали тестовое приложение, уязвимое к SQL-инъекции в серверной части, и 17 различных DEP, большая часть из которых — скрытые функции, использующие различные способы передачи параметров и формирования HTTP-запросов, взятые из реально существующих веб-приложений. На этом приложении протестировано несколько популярных сканеров безопасности — Acunetix, Positive Technologies Black Box Scanner, Detectify, Burp Scanner, HCL AppScan. Эксперименты показали, что предложенный алгоритм существенно превосходит возможности популярных сканеров в обнаружении скрытых функций веб-приложений.

ЛИТЕРАТУРА

1. *Huang Y. W., Huang S. K., Lin T. P., and Tsai C. H.* Web application security assessment by fault injection and behavior monitoring // Proc. WWW2003. Budapest, Hungary, May 21–25, 2003. P. 148–159.
2. *Раздобаров А. В., Петухов А. А., Гамаюнов Д. Ю.* Проблемы обнаружения уязвимостей в современных веб-приложениях // Проблемы информационной безопасности. Компьютерные системы. 2015. № 4. С. 64–69.
3. `https://w3techs.com/technologies/overview/client_side_language` — Статистика использования языков программирования на клиентской стороне веб-приложений по данным сайта w3techs.com.
4. *Richards G., Lebresne S., Burg B., and Vitek J.* An analysis of the dynamic behavior of JavaScript programs // ACM SIGPLAN Notices. 2010. V. 45. No. 6. P. 1–12.
5. `https://www.alexa.com/topsites` — Alexa Top 500 Global Sites.

6. `https://jquery.com` — Библиотека jQuery.
7. `https://www.google.com/recaptcha/about/` — Система защиты веб-сайтов от интернет-ботов reCAPTCHA.
8. *Kwangwon S. and Sukyoung R.* Analysis of JavaScript programs: Challenges and research trends // ACM Comput. Surveys. 2017. V. 50. No. 4. Article 59.
9. *Andreasen E., Gong L., Møller A., et al.* A survey of dynamic analysis and test generation for JavaScript // ACM Comput. Surveys. 2017. V. 50. No. 5. Article 66.
10. *Ryu S., Park J., and Park J.* Toward analysis and bug finding in JavaScript web applications in the wild // IEEE Software. 2018. V. 36. No. 3. P. 74–82.
11. *Andreasen E. and Møller A.* Determinacy in static analysis of jQuery // ACM SIGPLAN Notices. 2014. V. 49. No. 10. P. 17–31.
12. *Doupé A., Cova M., and Vigna G.* Why Johnny can't pentest: An analysis of black-box web vulnerability scanners // Proc. DIMVA 2010. Berlin; Heidelberg: Springer, 2010. P. 111–131.
13. *Doué A., Cavedon L., Kruegel C., and Vigna G.* Enemy of the state: A state-aware black-box web vulnerability scanner // 21st USENIX Security Symp. 2012. P. 523–538.
14. *Doupé A.* Advanced Automated Web Application Vulnerability Analysis. Diss. UC Santa Barbara, 2014.
15. *Choudhary S., Dincturk M., Mirtaheri S., et al.* Crawling rich internet applications: the state of the art // Proc. Conf. of the Center for Advanced Studies on Collaborative Research, 2012. P. 146–160.
16. *Mesbah A., Deursen A., and Lenselink S.* Crawling Ajax-based web applications through dynamic analysis of user interface state changes // ACM Trans. Web. 2012. V. 6. P. 3:1–3:30.
17. *Antal G., Hegedüs P., Toth Z., et al.* Static javascript call graphs: A comparative study // IEEE 18th Intern. Conf. SCAM. 2018. P. 177–186.
18. *Ko Y., Lee H., Dolby J., and Ryu S.* Practically tunable static analysis framework for large-scale JavaScript applications (T) // 30th IEEE/ACM Intern. Conf. ASE. 2015. P. 541–551.
19. *Wittern E., Ying A. T. T., Zheng Y., et al.* Statically checking web API requests in JavaScript // Proc. 39th Intern. Conf. Software Eng. 2017. P. 244–254.
20. `https://babeljs.io/` — Описание и документация библиотеки Babel.
21. `https://www.slideshare.net/pdug_slides/pt-blackbox-scanner` — Презентация с описанием возможностей средства PT BBS.
22. `https://www.acunetix.com` — Официальный сайт инструментального средства Acunetix.
23. `https://detectify.com` — Официальный сайт средства Detectify.
24. `https://portswigger.net/burp/burp-scanner` — Страница инструментального средства Burp Scanner на официальном сайте платформы Burp Suite.
25. `https://www.hcltechsw.com/wps/portal/products/appscan/home` — Официальный сайт сканера HCL AppScan Cloud.

REFERENCES

1. *Huang Y. W., Huang S. K., Lin T. P., and Tsai C. H.* Web application security assessment by fault injection and behavior monitoring. Proc. WWW2003, Budapest, Hungary, May 21–25, 2003, pp. 148–159.
2. *Razdobarov A. V., Petukhov A. A., and Gamayunov D. Yu.* Problemy obnaruzheniya uyazvimostey v sovremennykh veb-prilozheniyakh [Problems overview for modern web applications vulnerabilities discovery]. Problemy Informatsionnoy Bezopasnosti. Komp'yuternye Sistemy, 2015, no. 4, pp. 64–69. (in Russian)
3. `https://w3techs.com/technologies/overview/client_side_language` — Usage statistics of client-side programming languages for websites.

4. *Richards G., Lebresne S., Burg B., and Vitek J.* An analysis of the dynamic behavior of JavaScript programs. ACM SIGPLAN Notices, 2010, vol. 45, no. 6, pp. 1–12.
5. `https://www.alexa.com/topsites` — Alexa Top 500 Global Sites.
6. `https://jquery.com` — jQuery.
7. `https://www.google.com/recaptcha/about/` — reCAPTCHA.
8. *Kwangwon S. and Sukyoung R.* Analysis of JavaScript programs: Challenges and research trends. ACM Comput. Surveys, 2017, vol. 50, no. 4, Article 59.
9. *Andreasen E., Gong L., Møller A., et al.* A survey of dynamic analysis and test generation for JavaScript. ACM Comput. Surveys, 2017, vol. 50, no. 5, Article 66.
10. *Ryu S., Park J., and Park J.* Toward analysis and bug finding in JavaScript web applications in the wild. IEEE Software, 2018, vol. 36, no. 3, pp. 74–82.
11. *Andreasen E. and Møller A.* Determinacy in static analysis of jQuery. ACM SIGPLAN Notices, 2014, vol. 49, no. 10, pp. 17–31.
12. *Doupé A., Cova M., and Vigna G.* Why Johnny can't pentest: An analysis of black-box web vulnerability scanners. Proc. DIMVA 2010, Berlin; Heidelberg, Springer, 2010, pp. 111–131.
13. *Doué A., Cavedon L., Kruegel C., and Vigna G.* Enemy of the state: A state-aware black-box web vulnerability scanner. 21st USENIX Security Symp., 2012, pp. 523–538.
14. *Doupé A.* Advanced Automated Web Application Vulnerability Analysis. Diss. UC Santa Barbara, 2014.
15. *Choudhary S., Dincturk M., Mirtaheri S., et al.* Crawling rich internet applications: the state of the art. Proc. Conf. of the Center for Advanced Studies on Collaborative Research, 2012, pp. 146–160.
16. *Mesbah A., Deursen A., and Lenselink S.* Crawling Ajax-based web applications through dynamic analysis of user interface state changes. ACM Trans. Web, 2012, vol. 6, pp. 3:1–3:30.
17. *Antal G., Hegedüs P., Toth Z., et al.* Static javascript call graphs: A comparative study. IEEE 18th Intern. Conf. SCAM, 2018, pp. 177–186.
18. *Ko Y., Lee H., Dolby J., and Ryu S.* Practically tunable static analysis framework for large-scale JavaScript applications (T). 30th IEEE/ACM Intern. Conf. ASE, 2015, pp. 541–551.
19. *Wittern E., Ying A. T. T., Zheng Y., et al.* Statically checking web API requests in JavaScript. Proc. 39th Intern. Conf. Software Eng., 2017, pp. 244–254.
20. `https://babeljs.io/` — Babel.
21. `https://www.slideshare.net/pdug_slides/pt-blackbox-scanner` — PT BBS Slides.
22. `https://www.acunetix.com` — Acunetix.
23. `https://detectify.com` — Detectify.
24. `https://portswigger.net/burp/burp-scanner` — Burp Scanner.
25. `https://www.hcltechsw.com/wps/portal/products/appscan/home` — HCL AppScan Cloud.

ПРИКЛАДНАЯ ТЕОРИЯ КОДИРОВАНИЯ

УДК 519.7

О ГРАНИЦАХ МОЩНОСТИ ЗЛОУМЫШЛЕННИКОВ ДЛЯ ИДЕНТИФИЦИРУЮЩИХ АЛГЕБРОГЕОМЕТРИЧЕСКИХ КОДОВ НА СПЕЦИАЛЬНЫХ КРИВЫХ[1]

В. М. Деундяк[*,**], Д. В. Загуменнов[**]

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

Под схемами широковещательного шифрования понимают такие протоколы распространения легально тиражируемой цифровой продукции, которые способны предотвратить несанкционированный доступ к распространяемым данным. Эти схемы широко используются как для распределённого хранения данных, так и для защиты данных при передаче по каналам связи, и исследование таких схем представляется актуальной задачей. Для предотвращения коалиционных атак в схемах широковещательного шифрования используются классы помехоустойчивых кодов со специальными свойствами, в частности c-FP- и c-TA-свойствами. Рассматривается задача оценки нижней и верхней границ мощности коалиции злоумышленников, в пределах которых алгеброгеометрические коды обладают этими свойствами. Ранее были получены границы для одноточечных алгеброгеометрических кодов на кривых общего вида. В работе эти границы уточняются для одноточечных кодов на кривых специального вида; в частности, для кодов на кривых, на которых имеется достаточно много классов эквивалентности после факторизации множества точек кривой по отношению равенства соответствующих координат.

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

DOI 10.17223/20710410/53/4

THE ATTACKERS POWER BOUNDARIES FOR TRACEABILITY OF ALGEBRAIC GEOMETRIC CODES ON SPECIAL CURVES

V. M. Deundyak[*,**], D. V. Zagumennov[**]

[*] *FGANU NII "Specvuzavtomatika", Rostov-on-Don, Russia*
[**] *Southern Federal University, Rostov-on-Don, Russia*

E-mail: zagumionnov.denis@yandex.ru, vl.deundyak@gmail.com

Broadcast encryption is a data distribution protocol which can prevent malefactor parties from unauthorized accessing or copying the distributed data. It is widely used in distributed storage and network data protection schemes. To block the so-called coalition attacks on the protocol, classes of error-correcting codes with special

[1]Исследование выполнено при финансовой поддержке РФФИ в рамках научного проекта № 20-31-90098.

properties are used, namely c-FP and c-TA properties. We study the problem of evaluating the lower and the upper boundaries on coalition power, within which the algebraic geometry codes possess these properties. Earlier, these boundaries were calculated for single-point algebraic-geometric codes on curves of the general form. Now, we clarified these boundaries for single-point codes on curves of a special form; in particular, for codes on curves on which there are many equivalence classes after factorization by equality of the corresponding points coordinates relation.

Keywords: *traceability codes, frameproof codes, algebraic geometry codes, broadcast encryption.*

Введение

В [1, 2] представлена криптографическая схема широковещательного шифрования, способная обеспечивать защиту легально тиражируемой цифровой продукции от несанкционированного доступа. В этой схеме дистрибьютор распространяет данные свободно в зашифрованном виде, а для обеспечения доступа к этим данным со стороны легальных пользователей он должен выдать каждому из них уникальный набор ключей. Далее пользователи применяют свои ключи для расшифрования и получения доступа к информации. В случае обнаружения нелегального использования ключей их владелец может быть идентифицирован контролёром. Тем не менее допускаются так называемые коалиционные атаки, при которых легальные пользователи объединяются в коалиции злоумышленников и, комбинируя свои ключи, генерируют «пиратские» наборы ключей, которые могут быть использованы с целью выполнения несанкционированного доступа к данным. Если в качестве «пиратского» набора коалиции удается реплицировать набор ключей законопослушного пользователя схемы, не входящего в коалицию, то говорят, что этот набор скомпрометирован.

Для борьбы с коалиционными атаками в [3] предложено усиление этих схем, найденное в использовании специальных классов помехоустойчивых кодов, называемых идентифицирующими кодами (в частности, c-FP- и c-TA-кодов [4, 5]), а также быстрых алгоритмов списочного декодирования для них. Векторы идентифицирующих кодов используются в алгоритме раздачи ключей легальным пользователям, а списочный декодер может использоваться в алгоритме определения злоумышленников из коалиции в случае, если это возможно [1–3; 6, разд. 1]. Так, использование c-TA-кодов гарантирует успешность определения злоумышленника из коалиции мощности максимум $c \in \mathbb{N}$ [7, п. 1.3]. Использование списочных декодеров позволяет делать это достаточно быстро [3, п. III]. Использование c-FP-кодов не гарантирует успешность нахождения злоумышленников, но c-FP-коды, наряду с c-TA-кодами, гарантируют невозможность компрометации наборов ключей законопослушных пользователей со стороны коалиций мощности максимум $c \in \mathbb{N}$ [7, п. 1.3], что также может являться важным преимуществом схемы.

Таким образом, для построения корректной и быстрой схемы широковещательного шифрования достаточно найти и рассмотреть классы линейных кодов, такие, чтобы, во-первых, они обладали идентифицирующими свойствами, и, во-вторых, для них существовали полиномиальные списочные декодеры. В [7, 8] рассмотрены коды Рида — Соломона и списочный декодер Судана — Гурусвами [9]; в [10] вместе с кодами Рида — Соломона исследуются циклические коды; в [11] исследуются коды Рида — Маллера, для которых также разработаны методы списочного декодирования, например [12]; в [6, 13] рассматриваются алгеброгеометрические коды, для списочного декодирова-

ния которых применим списочный декодер из [9]. В [14] построены коды, обладающие идентифицирующим c-IPP-свойством, для которых существует алгоритм определения злоумышленника из коалиции мощности c, основанный на списочном декодере из [9].

Говоря об иных исследованиях идентифицирующих свойств линейных кодов и их приложениях в широковещательном шифровании, отметим следующие работы. В [15] вместе с постановкой задачи, представленной выше, приведён краткий обзор применений широковещательного шифрования в конкретных телекоммуникационных сетях и распределённых хранилищах. Описание и анализ эффективности схем широковещательного шифрования, основанных на применении идентифицирующих кодов, даны в [16]. В [3, п. IV, Question 11, 12] поставлены вопросы об эквивалентности идентифицирующих c-TA- и c-IPP-свойств, обсуждение которых ведётся в [17, 18]. Отметим, что зачастую в литературе рассматривается задача, отличная от предложенной авторами: сначала фиксируется мощность коалиции злоумышленников, а оценки для длины, размерности или мощности кода получают через значение мощности коалиции, не рассматривая возможности списочного декодирования [19 – 22]. Наконец, схемы широковещательного шифрования могут рассматриваться и без явного использования линейных кодов [23].

В настоящей работе уточняются условия наличия идентифицирующих свойств c-FP и c-TA для алгеброгеометрических кодов на кривых специального вида; в частности, для алгеброгеометрических кодов на кривых, на которых найдётся достаточно много классов эквивалентности после факторизации множества точек кривой по отношению равенства соответствующих координат. Заметим, что доказательство теоремы о границах основано на ключевой конструктивной технической лемме, которая фактически анализирует действия злоумышленников.

Структура статьи организована следующим образом: в п. 1 приводятся необходимые сведения об идентифицирующих кодах, а в п. 2 — об алгеброгеометрических кодах. В п. 3 формулируются основные результаты о границах и приводятся иллюстративные примеры, а в п. 4 доказываются основные результаты.

1. Идентифицирующие коды

Пусть $q = p^r$, где p — простое число; $r \in \mathbb{N}$. Рассмотрим линейное подпространство C размерности k в линейном пространстве $\mathbb{F}_q^n$, которое будем называть линейным кодом, k — размерностью кода, n — длиной кода.

Рассмотрим отображение $\rho : \mathbb{F}_q^n \times \mathbb{F}_q^n \to \mathbb{N} \cup \{0\}$, определяемое следующим образом:

$$\rho(x, y) = |\{i \in \mathbb{N} : 1 \leqslant i \leqslant n,\ x_i \neq y_i\}|.$$

Через d обозначим величину

$$d = \min\{\rho(x, y) : x, y \in C,\ x \neq y\};$$

будем называть её минимальным кодовым расстоянием кода C. Код C над полем $\mathbb{F}_q$ длины n, размерности k и с минимальным кодовым расстоянием d будем называть $[n, k, d]_q$-кодом, или просто $[n, k, d]$-кодом.

Пусть C — линейный $[n, k, d]_q$ код, $x, y \in C$,

$$I(x, y) = \{i \in \mathbb{N} : i \in \{1, \ldots, n\},\ x_i = y_i\};$$

ясно, что $|I(x, y)| = n - \rho(x, y)$.

Пусть $c \in \mathbb{N} \setminus \{1\}$. *Коалицией* кода C называется множество

$$C_0 = \{(u_1, u_2, \dots, u_c) : u_i \in C,\ u_i = (u_{i,1}, u_{i,2}, \dots, u_{i,n})\}.$$

Число c называется мощностью коалиции, а множество коалиций мощности не более c обозначается как $\mathrm{coal}_c(C)$. Множеством потомков коалиции C_0 называется множество

$$\mathrm{desc}(C_0) = \{(y_1, y_2, \dots, y_n) \in \mathbb{F}_q^n : \exists i \in \{1, \dots, c\}\, \exists j \in \{1, \dots, n\}\, (y_j = u_{i,j})\}.$$

Линейный код C будем называть c-FP-кодом [5, определение 1.1.1], если выполняется следующее условие:

$$\forall\, C_0 \in \mathrm{coal}_c(C)\ \forall z \in C \setminus C_0\ (z \notin \mathrm{desc}(C_0) \setminus C_0).$$

Множеством FP-компрометации для кода C [11, с. 101] называется множество

$$\Omega_{\mathrm{FP}}(C) = \{c \in \mathbb{N} \setminus \{1\} : \exists C_0 \in \mathrm{coal}_c(C)\ \exists z \in C \setminus C_0\ \big(z \in \mathrm{desc}(C_0) \setminus C_0)\}.$$

Линейный код C называется c-TA-кодом [5, определение 1.1.4], если

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

Множеством TA-компрометации для кода C [11, с. 101] называется множество

$$\Omega_{\mathrm{TA}}(C) = \{c \in \mathbb{N} \setminus \{1\} : \exists v \in C\ \exists C_0 \in \mathrm{coal}_c(C \setminus \{v\})\ \exists \omega \in \mathrm{desc}(C_0) \setminus C_0\ \forall u \in C_0\ \big(\rho(v, \omega) \leqslant \rho(u, \omega)\big)\}.$$

Множества $\Omega_{\mathrm{TA}}(C)$ и $\Omega_{\mathrm{FP}}(C)$ есть целочисленные лучи:

$$\Omega_{\mathrm{TA}}(C) = \{R_{\mathrm{TA}}(C), R_{\mathrm{TA}}(C) + 1, \dots\}, \quad \Omega_{\mathrm{FP}}(C) = \{R_{\mathrm{FP}}(C), R_{\mathrm{FP}}(C) + 1, \dots\}.$$

Величины $R_{\mathrm{TA}}(C)$ и $R_{\mathrm{FP}}(C)$ называются рубежами множеств компрометации. Из определений вытекает, что

$$\Omega_{\mathrm{FP}}(C) \subseteq \Omega_{\mathrm{TA}}(C), \quad R_{\mathrm{TA}}(C) \leqslant R_{\mathrm{FP}}(C).$$

2. Алгеброгеометрические коды

Рассмотрим конечное поле $\mathbb{F}_q$. Пусть $\mathbb{F}_q[x_1, x_2]$ и $\mathbb{F}_q[X_1, X_2, X_3]$ — кольца многочленов от двух и трёх переменных над $\mathbb{F}_q$ соответственно; $\mathbb{F}_q^{\mathrm{hom}}[X_1, X_2, X_3]$ — множество однородных многочленов из $\mathbb{F}_q[X_1, X_2, X_3]$.

Отметим, что между многочленами f из $\mathbb{F}_q[x_1, x_2]$ и однородными многочленами F из $\mathbb{F}_q^{\mathrm{hom}}[X_1, X_2, X_3]$ существует взаимно-однозначное соответствие [24, с. 106–107], определяемое по следующему правилу: если d — максимальная степень одночлена в многочлене $f \in \mathbb{F}_q[x_1, x_2]$, то F получается из f заменой каждого одночлена вида $x_1^i x_2^j$ на одночлен вида $X_1^i X_2^j X_3^{d-i-j}$. Это соответствие называется проективизацией.

Далее будем рассматривать аффинное $\mathbb{A}^3$ и проективное $\mathbb{P}^2$ пространства, а также естественное вложение аффинного пространства в проективное [24, с. 106]. В частности, если точка P имеет аффинные координаты (a_1, a_2), то проективные координаты соответствующей точки из $\mathbb{P}^2$ будем записывать $(a_1 : a_2 : 1)$; в случае так называемых точек на бесконечности третья проективная координата равна нулю [25, с. 7–8].

Пусть $F \in \mathbb{F}_q^{\hom}[X_1, X_2, X_3]$ — неприводимый однородный многочлен, а $\mathcal{X} = \mathcal{X}(F, \mathbb{F}_q)$ — плоская гладкая проективная кривая над $\mathbb{F}_q$, заданная корнями $F \in \mathbb{F}_q^{\hom}[X_1, X_2, X_3]$ [24, п. 2.1.2]. Каждая такая кривая имеет неотрицательный целочисленный параметр g, который называется родом. В случае плоской гладкой кривой он может быть легко вычислен [24, следствие 2.2.8].

Рассмотрим поле рациональных функций над $\mathcal{X}$; обозначим его $\mathbb{F}_q(\mathcal{X})$. Это поле содержит рациональные функции, где числитель и знаменатель — однородные многочлены из $\mathbb{F}_q^{\hom}[X_1, X_2, X_3]$ одной степени. Две такие функции определяют один и тот же элемент поля $\mathbb{F}_q(\mathcal{X})$, если они аддитивно отличаются на рациональную функцию, делящуюся на F [24, п. 2.5.4].

Согласно [24, п. 2.5.2], для любой функции $H \in \mathbb{F}_q(\mathcal{X})$ и любой точки $M \in \mathcal{X}$ можно задать целочисленный параметр, называемый порядком и обозначаемый $\mathrm{ord}_M(H)$.

Дивизором D на кривой $\mathcal{X}$ называют формальную взвешенную сумму точек:

$$D = \sum_{M \in \mathcal{X}} a_M M, \quad a_M \in \mathbb{Z}.$$

Множество $\mathrm{supp}(D) = \{M \in \mathcal{X} : a_M \neq 0\}$ называется носителем D; $\deg(D) = \sum a_M$ — степенью D. Иногда, если $\deg(D) = \alpha$, вместо D будем писать D_α. Говорят, что D эффективен, если все $a_M \geqslant 0$. Если D эффективен, то пишут $D \geqslant 0$.

Для $H \in \mathbb{F}_q(\mathcal{X})$ определим дивизор функции (H) следующим образом:

$$(H) = \sum_{M \in \mathcal{X}} \mathrm{ord}_M(H) M.$$

Для любой точки $M \in \mathcal{X}$ рассмотрим произвольный линейный однородный многочлен $L \in \mathbb{F}_q^{\hom}[X_1, X_2, X_3]$, такой, что $L(M) \neq 0$, и многочлен $G \in \mathbb{F}_q^{\hom}[X_1, X_2, X_3]$, у которого $\deg(G) = r$. Пусть $\mathcal{I}(M; \mathcal{X}; G) = \mathrm{ord}_M(G/L^r)$ [25, определение 2.22]. Дивизором пересечения G и $\mathcal{X}$ называется дивизор вида

$$\mathcal{X} \cdot G = \sum_{M \in \mathcal{X}} \mathcal{I}(M; \mathcal{X}; G) M.$$

Пусть D — дивизор на плоской гладкой проективной кривой $\mathcal{X}(F, \mathbb{F}_q)$. Тогда множество

$$L(D) = \{H \in \mathbb{F}_q(\mathcal{X}) : (H) + D \geqslant 0\}$$

называется пространством Римана — Роха, ассоциированным с D. Известно, что $L(D)$ является конечномерным векторным пространством [25, теорема 2.37].

Пусть $P = \{P_1, \ldots, P_n\} \subset \mathcal{X}$, $\deg(D) = \alpha$ и $\mathrm{supp}(D) \cap P = \varnothing$. Определим алгебро-геометрический код L-конструкции как образ отображения

$$Ev_P : L(D) \to \mathbb{F}_q^n, \quad Ev_P(H) = (H(P_1), H(P_2), \ldots, H(P_n)),$$

и обозначим его как $C(\mathcal{X}, P, D)$. Дивизор $D = D_\alpha$ называется дивизором кода C.

Утверждение 1 [24, теорема 4.1.1]. Пусть $\mathcal{X}(F, \mathbb{F}_q)$ — плоская гладкая проективная кривая рода g, $0 < \alpha < n$. Тогда $C(\mathcal{X}, P, D_\alpha)$ является $[n, k, d]_q$-кодом, где

$$k \geqslant \alpha - g + 1, \ d \geqslant n - \alpha.$$

Если $\alpha > 2g - 2$, то $k = \alpha - g + 1$.

Отметим, что если $g = 0$, $\deg(F) = 1$, P — множество всех конечных точек кривой, то $C(\mathcal{X}, P, D_\alpha)$ является $[q, \alpha + 1, q - \alpha]_q$-кодом Рида — Соломона [24, с. 60, с. 263–264].

3. Формулировка основных результатов об улучшении границ для кодов на алгебраических кривых со специальными свойствами

3.1. Алгебраические кривые со специальными свойствами

Пусть $\mathcal{X} = \mathcal{X}(F, \mathbb{F}_q)$ — плоская гладкая проективная кривая; $P = \{P_1, \ldots, P_n\}$ — множество всех точек вида $P_i = (P_{i,1} : P_{i,2} : 1)$ на кривой. Назовём это множество множеством конечных точек кривой.

Введём на нём два отношения эквивалентности:

$$P_i \sim_1 P_j \Leftrightarrow P_{i,1} = P_{j,1}, \quad P_i \sim_2 P_j \Leftrightarrow P_{i,2} = P_{j,2}.$$

Отношение $\sim_1$ разбивает P на классы эквивалентности

$$P/_{\sim_1} = \{R^1, \ldots, R^{k_1}\}, \quad R^i = \{R^i_j = (R^i_{j,1} : R^i_{j,2} : 1) \in P,\ j = 1, \ldots, l_i\}, \tag{1}$$

отношение $\sim_2$ разбивает P на классы эквивалентности

$$P/_{\sim_2} = \{S^1, \ldots, S^{k_2}\}, \quad S^i = \{S^i_j = (S^i_{j,1} : S^i_{j,2} : 1) \in P,\ i = 1, \ldots, m_i\}, \tag{2}$$

где l_i, m_i — мощности смежных классов R^i и S^i соответственно.

Значение k_1 назовём индексом множества P по первой координате; k_2 — индексом множества P по второй координате. Очевидно, что $k_1 \leqslant n$, $k_2 \leqslant n$. Легко проверить, что если оба индекса равны 1, то множество P состоит из одной точки. Таким образом, если $|P| > 1$, то один из индексов k_1, k_2 также больше 1.

Рассмотрим некоторые примеры.

Пример 1. Рассмотрим в качестве $\mathcal{X}$ проективную прямую, имеющую род 0 и заданную многочленом $F = X_2 - X_3$ над произвольным полем $\mathbb{F}_q$. Множество P (см. (1), (2)) для такой кривой состоит из точек вида $(\alpha_j : 1 : 1)$, $\alpha_j \in \mathbb{F}_q$. Таким образом, на этой кривой есть q классов R^i мощности $\deg(F) = 1$.

Пример 2. Рассмотрим в качестве $\mathcal{X}$ проективную прямую, имеющую род 1 и заданную многочленом

$$F(X_1, X_2, X_3) = X_2^2 X_3 + X_1 X_2 X_3 + X_2 X_3^2 - X_1^3 - X_3^3$$

над полем $\mathbb{F}_8 = \mathbb{F}_2[\xi]/(\xi^3 + \xi + 1)$. Выпишем все точки кривой:

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

Построим классы S^i:

$$S^0 = \{(1 : 0 : 1)\},$$
$$S^1 = \{(\xi^4 : 1 : 1), (\xi^2 : 1 : 1), (\xi : 1 : 1)\},$$
$$S^2 = \{(\xi : \xi : 1), (\xi^5 : \xi : 1), (\xi^6 : \xi : 1)\},$$
$$S^3 = \{(\xi^2 : \xi^2 : 1), (\xi^5 : \xi^2 : 1), (\xi^3 : \xi^2 : 1)\},$$
$$S^4 = \{(\xi^3 : \xi^4 : 1), (\xi^4 : \xi^4 : 1), (\xi^6 : \xi^4 : 1)\}.$$

Пример 3. Рассмотрим кривую, имеющую род 6 и заданную многочленом

$$X^5 + Y^4 Z + Y Z^4 = 0$$

над полем $\mathbb{F}_{16}$. Можно проверить, что это максимальная кривая, содержащая 65 точек и 12 классов S^i мощности $\deg(F) = 5$.

3.2. Улучшение границ

Теорема 1. Пусть $\mathcal{X}(F, \mathbb{F}_q)$ — плоская гладкая проективная кривая, Q — её единственная точка на бесконечности. Рассмотрим код $C = C(\mathcal{X}(F, \mathbb{F}_q), P, D_\alpha)$, где $D = \alpha Q$. Пусть $|P| = n > 1$, $\varkappa$ — меньший из индексов k_1, k_2, $\varkappa > 1$, $\delta = \lfloor \alpha / \deg(F) \rfloor$. Предположим, что на кривой $\mathcal{X}$ существует $\delta \cdot \max\{2, \lfloor \varkappa/\delta \rfloor\}$ классов эквивалентности мощности $\deg(F)$, соответствующих индексу $\varkappa$ (если $\varkappa = k_1$, то классов R^i, иначе — S^i). Тогда

$$\begin{aligned}\sqrt{n/\alpha} \leqslant R_{\mathrm{TA}}(C) \leqslant \tilde{B}_{\mathrm{TA}}(C) = \min\left\{\left\lceil \frac{n+\alpha}{2\delta \deg(F)} \right\rceil, \left\lceil \frac{\varkappa}{\delta} \right\rceil\right\} \leqslant \left\lceil \frac{n}{\alpha} \right\rceil \leqslant R_{\mathrm{FP}}(C) \leqslant \\ \leqslant \tilde{B}_{\mathrm{FP}}(C) = \min\left\{\left\lceil \frac{n}{\lfloor \alpha/\deg(F) \rfloor \deg(F)} \right\rceil, \left\lceil \frac{\varkappa}{\delta} \right\rceil\right\}.\end{aligned} \tag{3}$$

Замечание 1. Ранее в [13, теоремы 3 и 4] получены оценки для рубежей R_{TA} и R_{FP} для более широкого класса кривых. Для кривых с рассматриваемыми специальными свойствами эти оценки усилены, при этом новая верхняя оценка для R_{TA} не превосходит новой нижней оценки для R_{FP}.

Замечание 2. Рассмотрим код $C = (\mathcal{X}(F, \mathbb{F}_q), P, D_\alpha)$. Если род g кривой $\mathcal{X}$ равен нулю, $\deg(F) = 1$ и $D = \alpha Q$, где Q — единственная точка на бесконечности, то код C является кодом Рида — Соломона, а его размерность k в силу утверждения 1 равна $\alpha + 1$. В таком случае неравенства (3) имеют вид

$$\sqrt{\frac{n}{k-1}} \leqslant R_{\mathrm{TA}} \leqslant \tilde{B}_{\mathrm{TA}}(C) = \left\lceil \frac{n+k-1}{2(k-1)} \right\rceil \leqslant R_{\mathrm{FP}} = \tilde{B}_{\mathrm{FP}}(C) = \left\lceil \frac{n}{k-1} \right\rceil$$

и совпадают по существу с оценками, полученными для кодов Рида — Соломона в [7].

3.3. Примеры вычисления параметров кодов

Пример 4. На каждой из рассмотренных в примерах 1–3 кривых находится одна и только одна точка на бесконечности. На первой кривой имеется q классов эквивалентности мощности, равной степени задающего её многочлена, на второй — 4 таких класса, а на третьей — 12. Поэтому эти кривые удовлетворяют условиям теоремы 1 при $\delta \leqslant \lfloor q/2 \rfloor$, $\delta \leqslant 2$ и $\delta \leqslant 6$ соответственно.

Пример 5. Рассмотрим кривую из примера 1 над полем $\mathbb{F}_{16}$ и дивизор $D = 2 \cdot (0 : 1 : 0)$ на ней. Выше отмечено, что эта кривая удовлетворяет условиям теоремы 1 при $\delta \leqslant 8$. В качестве множества P возьмём всё множество конечных точек кривой. Согласно утверждению 1, на рассмотренной кривой можно построить алгеброгеометрический $[16, 3, d]_{16}$-код L-конструкции C, где $d \geqslant 14$; в силу неравенства Синглтона $d = 14$. Известно, что этот код является кодом Рида — Соломона. В этом случае $\varkappa = n = 16$, $\alpha = 2$, $\delta = \lfloor 2/1 \rfloor = 2$. Для кода C выполняются все условия теоремы 1, следовательно

$$3 \leqslant R_{\mathrm{TA}}(C) \leqslant 5 \leqslant R_{\mathrm{FP}}(C) = 8$$

(см. также замечание 2).

Пример 6. Рассмотрим кривую над полем $\mathbb{F}_8$ из примера 2 и дивизор $D = 3Q$ на ней. В примере 4 отмечено, что эта кривая удовлетворяет условиям теоремы 1 при $\delta \leqslant 2$. В качестве множества P возьмём всё множество конечных точек кривой. Согласно утверждению 1, на этой кривой можно построить алгеброгеометрический $[13, 3, d]_8$-код L-конструкции C, где $d \geqslant 10$. В этом случае $n = 13$, $\varkappa = 5$, $\alpha = 3$, $\delta = \lfloor 3/3 \rfloor = 1$. Для кода C выполняются все условия теоремы 1, следовательно

$$R_{\mathrm{TA}}(C) = 3, \quad R_{\mathrm{FP}}(C) = 5.$$

Пример 7. Рассмотрим кривую над полем $\mathbb{F}_{16}$ из примера 3 и дивизор $D = 9 \cdot (0 : 1 : 0)$ на ней. В примере 4 показано, что эта кривая удовлетворяет условиям теоремы 1 при $\delta \leqslant 6$. В качестве множества P возьмём всё множество конечных точек кривой, общее количество которых, как показано в примере 3, равно 64. Тогда, согласно утверждению 1, на этой кривой можно построить алгеброгеометрический $[64, k, d]_{16}$-код L-конструкции C, где $k \geqslant 4$, $d \geqslant 55$ и, как следует из неравенства Синглтона, $k + d \leqslant 65$. В этом случае $n = 64$, $\varkappa = 12$, $\alpha = 9$, $\delta = \lfloor 9/5 \rfloor = 1$. Таким образом, для кода C выполняются все условия теоремы 1, значит,

$$3 \leqslant R_{\mathrm{TA}}(C) \leqslant 8 \leqslant R_{\mathrm{FP}}(C) \leqslant 12.$$

3.4. Применимость к схемам широковещательного шифрования

В работе рассмотрены идентифицирующие свойства алгеброгеометрических кодов в следующей постановке. Пусть выбран алгеброгеометрический код, для которого выполнены условия теоремы 1, и зафиксированы его основные параметры. Возникает вопрос, какое значение может принимать c, чтобы код сохранял c-FP- или c-TA-свойства? Ответ на этот вопрос дан в теореме 1. Таким образом, можно выбрать алгеброгеометрический код C со специальными свойствами (теорема 1), списочный декодер Судана — Гурусвами [9] и, проверив условия применимости этого списочного декодера к C [3, теорема 5; 6, следствие 1], построить:

1) устойчивую схему широковещательного шифрования, способную гарантировать невозможность прямой компрометации ключей законопослушных пользователей со стороны коалиций мощности вплоть до $c_1 = \lceil n/\alpha \rceil - 1 < \lceil n/\alpha \rceil \leqslant R_{\mathrm{FP}}(C)$;
2) отслеживающую схему широковещательного шифрования, способную находить участников коалиций мощности вплоть до $c_2 = \lceil \sqrt{n/\alpha} - 1 \rceil < \sqrt{n/\alpha} \leqslant R_{\mathrm{TA}}(C)$;
3) устойчивую и отслеживающую схему широковещательного шифрования, способную гарантировать как невозможность прямой компрометации ключей законопослушных пользователей со стороны коалиций мощности вплоть до $c_1 = \lceil n/\alpha \rceil - 1 < \lceil n/\alpha \rceil \leqslant R_{\mathrm{FP}}(C)$, так и находить при этом злоумышленников из коалиций мощности вплоть до $c_2 = \lceil \sqrt{n/\alpha} - 1 \rceil < \sqrt{n/\alpha} \leqslant R_{\mathrm{TA}}(C)$.

Возможность построения схем с такими свойствами обоснована в [7, п. 1.3]. Максимальное число пользователей такой схемы ограничено сверху мощностью кода [6, утверждение 2]. Сформулируем и докажем следующее утверждение:

Теорема 2. Пусть выполняются условия теоремы 1, $c \in \mathbb{N} \setminus \{1\}$. Тогда

1) если $c < \sqrt{n/\alpha}$, то
$$\max_{\alpha} |C| = q^{\lceil n/c^2 - 1 \rceil - g + 1};$$
2) если $c < \lceil n/\alpha \rceil$, то
$$\max_{\alpha} |C| = q^{\lceil n/(c-\varepsilon) - 1 \rceil - g + 1},$$
где $\varepsilon = \lceil n/\alpha \rceil - n/\alpha$.

Доказательство. Если C — линейный $[n, k, d]_q$-код, то $|C| = q^k$.

1) Если $c < \sqrt{n/\alpha}$, то $\alpha < n/c^2$, тогда
$$\max_{\alpha} \{\alpha : \alpha < n/c^2\} = \lceil n/c^2 - 1 \rceil.$$

Из утверждения 1 получим, что $\max\limits_{\alpha} k = \lceil n/c^2 - 1 \rceil - g + 1$. Следовательно, $\max\limits_{\alpha} |C| = q^{\lceil n/c^2 - 1 \rceil - g + 1}$.

2) Аналогично, так как $c < \lceil n/\alpha \rceil = n/\alpha + \varepsilon$, то $\alpha < n/(c-\varepsilon)$. Тогда

$$\max_{\alpha}\{\alpha : \alpha < n/(c-\varepsilon)\} = \lceil n/(c-\varepsilon) - 1 \rceil,$$

значит, $\max\limits_{\alpha} k = \lceil n/(c-\varepsilon) - 1 \rceil - g + 1$ и $\max\limits_{\alpha} |C| = q^{\lceil n/(c-\varepsilon) - 1 \rceil - g + 1}$. ■

Справедливость утверждения означает, что максимальное число пользователей устойчивой схемы равно $q^{\lceil n/(c_1-\varepsilon) - 1 \rceil - g + 1}$, так как $c_1 < \lceil n/\alpha \rceil$, а для отслеживающей $q^{\lceil n/c_2^2 - 1 \rceil - g + 1}$, так как $c_2 < \sqrt{n/\alpha}$. Для усточивой и одновременно отслеживающей схемы число пользователей равно $\min\{q^{\lceil n/c_1^2 - 1 \rceil - g + 1}, q^{\lceil n/(c_2-\varepsilon) - 1 \rceil - g + 1}\}$, так как в такой схеме $c_1 < \lceil n/\alpha \rceil$ и $c_2 < \sqrt{n/\alpha}$ одновременно.

Пример 8. Используя код из примера 5, можно построить устойчивую схему для $c_1 = 4$, отслеживающую схему для $c_2 = 2$ или устойчивую и отслеживающую для $c_1 = 4$, $c_2 = 2$. Для устойчивой схемы максимальное число пользователей равно $16^{\lceil 16/(4-0) - 1 \rceil + 1} = 16^4 = 2^{16}$, так как $\varepsilon = \lceil 16/4 \rceil - 16/4 = 0$; для отслеживающей схемы максимальное число пользователей также равно $16^{\lceil 16/4 - 1 \rceil + 1} = 16^4 = 2^{16}$; для устойчивой и отслеживающей схемы максимальное число пользователей равно $\min\{2^{16}, 2^{16}\} = 2^{16}$.

Пример 9. Используя код из примера 6, можно построить устойчивую схему для $c_1 = 4$, отслеживающую схему для $c_2 = 2$, устойчивую и одновременно отслеживающую схему для $c_1 = 4$, $c_2 = 2$. Для устойчивой схемы максимальное число пользователей равно $8^{\lceil 13/(4-2/3) - 1 \rceil - 1 + 1} = 8^3 = 2^9$, так как $\varepsilon = \lceil 13/3 \rceil - 13/3 = 2/3$; для отслеживающей схемы также равно $8^{\lceil 13/4 - 1 \rceil - 1 + 1} = 8^3 = 2^9$; для усточивой и отслеживающей схемы равно $\min\{2^9, 2^9\} = 2^9$.

Пример 10. Используя код из примера 7, можно построить устойчивую схему для $c_1 = 7$, отслеживающую схему для $c_2 = 2$, устойчивую и одновременно отслеживающую для $c_1 = 7$, $c_2 = 2$. Максимальное число пользователей для устойчивой схемы равно $16^{\lceil 64/(7-8/9) - 1 \rceil - 6 + 1} = 16^5 = 2^{20}$, так как $\varepsilon = 8 - 7\frac{1}{9} = \frac{8}{9}$; для отслеживающей схемы максимальное число пользователей равно $16^{\lceil 64/4 - 1 \rceil - 6 + 1} = 16^{10} = 2^{40}$; для устойчивой и отслеживающей — $\min\{2^{40}, 2^{20}\} = 2^{20}$.

Рассмотрим иную постановку задачи. Предположим, что известна максимально возможная мощность коалиции злоумышленников c. Возникает вопрос, какой код (в частности, какой алгеброгеометрический код) выбрать, чтобы построить схему широковещательного шифрования при заданном c? Возможно ли использование списочного декодирования в этом случае? Эти задачи выходят за рамки данной работы и представляются темой отдельного исследования. При их решении может возникать необходимость перебора кодов; формализация такого перебора для одноточечных алгеброгеометрических кодов общего вида рассмотрена в [6, алгоритм 1].

4. Доказательство основного результата

4.1. Вспомогательные леммы

Следующая конструктивная техническая лемма является ключевой для доказательства теоремы 1. В её доказательстве, фактически имитирующем действия злоумышленников, явно показан способ построения коалиции и её потомка, на которых достигаются представленные в утверждении леммы оценки и равенства.

Лемма 1. Пусть $\mathcal{X}(F,\mathbb{F}_q)$ — плоская гладкая проективная кривая, Q — её единственная точка на бесконечности. Рассмотрим код $C=(\mathcal{X}(F,\mathbb{F}_q),P,D_\alpha)$, где $D=\alpha Q$. Пусть $|P|>1$, $\varkappa$ — меньший из индексов k_1,k_2, $\varkappa>1$, $\delta=\lfloor\alpha/\deg(F)\rfloor$. Предположим, что на кривой $\mathcal{X}$ существует $\delta\cdot\max\{2,\lfloor\varkappa/\delta\rfloor\}$ классов эквивалентности мощности $\deg(F)$, соответствующих индексу $\varkappa$ (если $\varkappa=k_1$, то классов R^i, иначе — S^i). Тогда

— для всех c, таких, что $2\leqslant c\leqslant\varkappa/\delta$

$$\forall v\in C\ \exists C_0\in\operatorname{coal}_c(C\setminus\{v\})\ \exists\omega\in\operatorname{desc}(C_0)\setminus C_0\ \big(|I(\omega,v)|\geqslant\min\{c\delta\deg(F),n\}\big),\quad(4)$$

— для всех c, таких, что $c>\varkappa/\delta$

$$\forall v\in C\ \exists C_0\in\operatorname{coal}_c(C\setminus\{v\})\ \exists\omega\in\operatorname{desc}(C_0)\setminus C_0\ \big(|I(\omega,v)|=n\big).\quad(5)$$

Доказательство. Предположим, что лемма доказана для $v=0$, т. е. можно построить коалицию $\hat{C}_0=\{\hat{u}_1,\dots,\hat{u}_c\}$, такую, что при $v=0$ выполняются (4) и (5). Рассмотрим произвольный вектор $v\in C$, коалицию $C_0=\{\hat{u}_1+v,\dots,\hat{u}_c+v\}$ и вектор $\omega=\hat{\omega}+v$. Так как C — линейный код, $C_0\in\operatorname{coal}_c(C\setminus\{v\})$, $\omega\in\operatorname{desc}(C_0)\setminus C_0$, и (4), (5) выполняются для произвольного v. Таким образом, если лемма справедлива для $v=0$, то она справедлива и для любых других $v\in C$.

Докажем теперь лемму в предположении, что $v=0$.

Рассмотрим сначала случай, когда $c\delta<n$, и разобьём доказательство на несколько шагов.

I. Рассмотрим множество $P\subset\mathcal{X}$. Так как по построению АГ-кода $\operatorname{supp}(D)\cap P=\varnothing$, $\operatorname{supp}(D)=\{Q\}$ и Q — единственная бесконечная точка на кривой $\mathcal{X}$, то в P нет бесконечных точек, т. е. точек вида $(X_1:X_2:0)$. Значит, множество P является подмножеством множества конечных точек кривой, причём, так как $|P|>1$, один из индексов P больше единицы (см. (1), (2)). Не нарушая общности, будем считать, что $\varkappa=k_2>1$, и рассмотрим классы эквивалентности S^i. Перенумеруем множество P так, чтобы для первых $\varkappa$ точек из P выполнялось условие $P_i\in S^i$, $i\in\{1,\dots,\varkappa\}$.

II. Так как коалиция является набором кодовых векторов, каждый из которых является образом некоторой рациональной функции из пространства Римана — Роха $L(D)$ при кодирующем отображении, для построения искомой коалиции необходимо сначала предъявить соответствующий набор рациональных функций.

Для искомых функций построим вспомогательные многочлены. Рассмотрим несколько случаев:

а) Пусть $c\delta\leqslant\varkappa$. Предположим, что F является проективизацией некоторого многочлена $f\in\mathbb{F}_q[x_1,x_2]$. По условию существует $\max\{2\delta,\lfloor\varkappa/\delta\rfloor\delta\}$ классов S^i, мощность которых равна $\deg(F)$. Так как $2\leqslant c\leqslant\varkappa/\delta$, это означает, что существует как минимум $c\delta$ классов S^i, каждый из которых имеет мощность $\deg(F)$. Следовательно, существуют $\beta_1,\dots,\beta_{c\delta}$, $\beta_i\in\mathbb{F}_q$ (β_i соответствует классу S^i), такие, что $f(x_1,\beta_i)$ имеет $\deg(F)$ различных корней. Обозначим корни $f(x_1,\beta_i)$ как α_{lj}, $l=1,\dots,\deg(F)$, $j=1,\dots,c\delta$. Каждому такому корню соответствует точка $(\alpha_{li}:\beta_i:1)\in S^i$.

Тогда многочлены r_i будем строить следующим образом:

$$r_i(x_1,x_2)=(x_2-\beta_{(i-1)\delta+1})\dots(x_2-\beta_{i\delta}),\ i=1,\dots,c.\quad(6)$$

Каждый одночлен в $r_i(x_1,x_2)$ имеет $\deg(F)$ корней вида

$$(\alpha_{lj},\beta_j),\ l=1,\dots,\deg(F),\ j\in\{(i-1)\delta+1,\dots,i\delta\},$$

значит, у $r_i(x_1, x_2)$ всего $\delta \deg(F)$ корней вида

$$(\alpha_{lj}, \beta_j),\ l = 1, \ldots, \deg(F),\ j = (i-1)\delta + 1, \ldots, i\delta.$$

Для каждого r_i и точки $P_l = R_1^{i\delta+1} = (P_{l,1} : P_{l,2} : 1)$ выполняется: $r_i(P_{l,1}, P_{l,2}) \neq 0$. Степень каждого r_i равна δ.

б) Пусть теперь $c\delta > \varkappa$, причём $\varkappa \leqslant \delta$ и $\varkappa < c$. Тогда в кольце $\mathbb{F}_q[x_1, x_2]$ рассмотрим следующие многочлены:

$$r_i(x_1, x_2) = (x_2 - S_{1,1}^i),\ i \in \{1, \ldots, \varkappa\}. \tag{7}$$

Для каждого r_i и точки $P_l = S_1^{i+1} = (P_{l,1} : P_{l,2} : 1)$ выполняется: $r_i(P_{l,1}, P_{l,2}) \neq 0$. Степень каждого r_i не превышает δ. Рассмотрим множество ненулевых многочленов степени не выше δ, не совпадающих с r_i, $i \in \{1, \ldots, \varkappa\}$. Таких многочленов $q^{\delta+1} - \varkappa - 1$ штук. Тогда в качестве r_j, $j \in \{\varkappa + 1, \ldots, c\}$, возьмём многочлены из этого множества, такие, что $r_j(P_{l,1}, P_{l,2}) \neq 0$ для некоторой $P_l = (P_{l,1} : P_{l,2} : 1) \in P$.

в) Пусть теперь $c\delta > \varkappa$, причём $\varkappa \leqslant \delta$, но $\varkappa \geqslant c$. Тогда в кольце $\mathbb{F}_q[x_1, x_2]$ можно рассмотреть следующие многочлены:

$$r_i(x_1, x_2) = (x_2 - S_{1,1}^i),\ i \in \{1, \ldots, c-1\},\quad r_c(x_1, x_2) = (x_2 - S_{1,1}^c) \ldots (x_1 - S_{1,1}^{\varkappa}). \tag{8}$$

Для каждого такого r_i найдётся точка $P_l = (P_{l,1} : P_{l,2} : 1)$, для которой $r_i(P_{l,1}, P_{l,2}) \neq 0$. Для $i < c$ такой точкой, например, является S_1^{i+1}, а для $i = c$ — точка S_1^{c-1}. Так как $\varkappa - c + 1 \leqslant \delta - c + 1 \leqslant \delta$, степень каждого r_i не превышает δ.

г) Пусть $c\delta > \varkappa$ и $\varkappa > \delta$. Тогда в кольце $\mathbb{F}_q[x_1, x_2]$ можно рассмотреть следующие многочлены:

$$\begin{gathered} r_i = (x_1 - S_{1,1}^{(i-1)\delta+1}) \ldots (x_1 - S_{1,1}^{i\delta}),\quad i \in \{1, \ldots, \lceil \varkappa/\delta \rceil - 1\}, \\ r_{\lceil \varkappa/\delta \rceil} = (x_2 - S_{1,1}^{(\lceil \varkappa/\delta \rceil - 1)\delta+1}) \ldots (x_2 - S_{1,1}^{\varkappa}). \end{gathered} \tag{9}$$

Для каждого такого r_i найдётся точка $P_l = (P_{l1} : P_{l2} : 1)$, такая, что $r_i(P_{l1}, P_{l2}) \neq 0$. Для $i \leqslant \lceil \varkappa/\delta \rceil - 1$ такой точкой, например, является $S_1^{i\delta+1}$, а для $i = \lceil \varkappa/\delta \rceil$ — точка $S_1^{(\lceil \varkappa/\delta \rceil - 1)\delta}$. Так как

$$\varkappa - ((\lceil \varkappa/\delta \rceil - 1)\delta + 1) + 1 < c\delta - (\lceil c\delta/\delta \rceil - 1)\delta \leqslant c\delta - (c-1)\delta \leqslant \delta,$$

то степень каждого r_i не превышает δ. В качестве r_j, $j \in \{\lceil \varkappa/\delta \rceil + 1, \ldots, c\}$, возьмём произвольные ненулевые многочлены степени не выше δ, не совпадающие с r_i, $i \in \{1, \ldots, \lceil \varkappa/\delta \rceil\}$, такие, что для них существует точка $(P_{l,1} : P_{l,2} : 1) \in P$, такая, что $r_i(P_{l,1}, P_{l,2}) \neq 0$.

Во всех случаях *а–г* степень каждого из многочленов r_i не превышает δ, все многочлены различны и для каждого r_i найдётся такая точка $P_l = (P_{l,1} : P_{l,2} : 1) \in P$, что $r_i(P_{l,1}, P_{l,2}) \neq 0$.

Рассмотрим проективизацию многочленов $r_i(x_1, x_2)$ и получим однородные многочлены $R_i(X_1, X_2, X_3)$ степени не выше δ.

В случае II, *а* однородные многочлены $R_i(X_1, X_2, X_3)$, являющиеся проективизацией $r_i(x_1, x_2)$, обращаются в нуль на точках проективного пространства вида

$$P_{j,\beta_i} = (\alpha_{lj} : \beta_l : 1),\ j = 1, \ldots, \deg(F),\ l = (i-1)\delta + 1, \ldots, i\delta.$$

Легко проверить, что эти точки лежат на кривой $\mathcal{X}$, так как $f(\alpha_{lj}, \beta_l) = 0$, а значит, $F(\alpha_{lj}, \beta_l, 1) = 0$. Таким образом, в случае II, *а* $R_i(X_1, X_2, X_3)$ обращаются в нуль на $\delta \deg(F)$ точках кривой $\mathcal{X}$, причём для разных R_i наборы таких точек не пересекаются.

Построим рациональные функции:

$$H_i = \frac{R_i(X_1, X_2, X_3)}{X_3^{\deg(R_i)}}, \tag{10}$$

являющиеся элементами поля $\mathbb{F}_q(\mathcal{X})$. Покажем, что H_i принадлежит пространству Римана — Роха $L(D)$, ассоциированному с дивизором D. Действительно, согласно [6, замечание 2], дивизор построенной функции H_i имеет вид

$$(H_i) = \mathcal{X} \cdot R_i(X_1, X_2, X_3) - \mathcal{X} \cdot X_3^{\deg(R_i)}.$$

В силу замечания к теореме 2.23 [25],

$$\sum_{M \in \mathcal{X}(F, \mathbb{F}_q)} \mathcal{I}(M; \mathcal{X}(F, \mathbb{F}_q); G) \leqslant \deg(G) \deg(F),$$

поэтому

$$\mathcal{X} \cdot X_3^{\deg(R_i)} = \sum_{M \in \mathcal{X}(F, \mathbb{F}_q)} \mathcal{I}\left(M; \mathcal{X}(F, \mathbb{F}_q); X_3^{\deg(R_i)}\right) M = \mathcal{I}(Q; \mathcal{X}(F, \mathbb{F}_q); X_3^{\deg(R_i)}) Q \leqslant$$
$$\leqslant \deg\left(X_3^{\deg(R_i)}\right) \deg(F) Q = \deg(R_i) \deg(F) Q.$$

Тогда

$$(H_i) = \mathcal{X} \cdot R_i(X_1, X_2, X_3) - \mathcal{X} \cdot X_3^{\deg(R_i)} \geqslant \sum_{P_j \in P} \mathcal{I}(P_j, X, R_i) P_j - \deg(R_i) \deg(F) Q,$$
$$(H_i) + D \geqslant \sum_{P_j \in P} \mathcal{I}(P_j, \mathcal{X}, R_i) P_j - \deg(R_i) \deg(F) Q + \alpha Q =$$
$$= \sum_{P_j \in P}^{\delta i} \mathcal{I}(P_j, \mathcal{X}, \mathcal{P}) P_j + (\alpha - \deg(R_i) \deg(F)) Q.$$

Так как $\deg(R_i) \leqslant \delta = \lfloor \alpha / \deg(F) \rfloor$, то

$$(H_i) + D \geqslant \sum_{P_j \in P} \mathcal{I}(P_j, \mathcal{X}, \mathcal{P}) P_j + (\alpha - \lfloor \alpha / \deg(F) \rfloor \deg(F)) Q \geqslant 0,$$

значит, $H_i \in L(D)$.

III. Построим теперь искомую коалицию $C_0 = \{u_1; \ldots; u_c\}$ следующим образом:

$$u_i = Ev_P(H_i) = (H_i(P_1), \ldots, H_i(P_n)). \tag{11}$$

Все многочлены r_i различны, поэтому и все R_i, а также H_i тоже различны. Для каждого r_i существует точка $P_l = (P_{l,1} : P_{l,2} : 1) \in P$, такая, что $r_i(P_{l,1}, P_{l,2}) \neq 0$, следовательно, и $H_i(P_{l,1} : P_{l,2} : 1) \neq 0$. Таким образом, в коалиции ровно c различных ненулевых векторов.

IV. Теперь для каждого из рассмотренных на шаге II случаев *а–г* построим искомого потомка ω.

В случае *а* коалиция C_0 (см. (11)) в силу (6) и (10) выглядит следующим образом:

$$\begin{cases} u_1 = (0, \dots, 0, H(P_{\delta \deg(F)+1}), \dots, H(P_n)), \\ \dots \\ u_i = (H(P_1), \dots, H(P_{\delta \deg(F)(i-1)}), 0, \dots, 0, H(P_{\delta \deg(F)i+1}), \dots, H(P_n)), \\ \dots \\ u_c = (H(P_1), \dots, H(P_{\delta \deg(F)(c-1)}), 0, \dots, 0, H(P_{\delta \deg(F)c+1}), \dots, H(P_n)). \end{cases}$$

Рассмотрим потомка коалиции C_0

$$\omega = (0, \dots, 0, \omega_{\delta \deg(F)c+1}, \dots, \omega_n),$$

где для каждого $j \in \{\delta c + 1, \dots, n\}$ значение ω_j задаётся как произвольный элемент из $\{u_{1,j}, \dots, u_{c,j}\}$. Ясно, что $\omega \in \operatorname{desc}(C_0) \setminus C_0$. По построению

$$|I(\omega, 0)| \geqslant c\delta \deg(F) = c \lfloor \alpha / \deg(F) \rfloor \deg(F).$$

В случае *б* коалиция в силу (7) и (10) выглядит следующим образом:

$$\begin{cases} u_1 = (0, \dots, H(P_i), \dots, H(P_n)), \\ \dots \\ u_i = (H(P_1), \dots, H(P_{i-1}), 0, H(P_{i+1}), \dots, H(P_n)), \\ \dots \\ u_\varkappa = (H(P_1), \dots, H(P_{\varkappa-1}), 0, H(P_{\varkappa+1}), \dots, H(P_n)), \\ \dots \\ u_j = (H(P_1), \dots, H(P_{j-1}), H(P_j), H(P_{j+1}), \dots, H(P_n)), \\ \dots \\ u_c = (H(P_1), \dots, H(P_{c-1}), H(P_{c+1}), \dots, H(P_n)). \end{cases}$$

Построим потомка ω следующим образом. В качестве ω_i, где $1 \leqslant i \leqslant \varkappa$, из вектора u_i возьмём нуль, стоящий там на i-й позиции. Заметим, что для любой позиции j, такой, что $j > \varkappa$, точка P_j лежит в каком-либо классе эквивалентности S^m. Тогда $u_{m,j} = 0$, так как по построению значение H_m равно нулю на любой точке из S^m, в том числе на P_j. Значит, для любой такой позиции j мы можем выбрать $\omega_j = u_{m,j} = 0$. Таким образом, комбинированием только первых $\varkappa$ векторов можно выбрать потомка ω, совпадающего с нулевым вектором. Тогда

$$|I(\omega, 0)| = n.$$

В случае *в* коалиция в силу (8) и (10) выглядит следующим образом:

$$\begin{cases} u_1 = (0, \dots, H(P_i), \dots, H(P_n)), \\ \dots \\ u_i = (H(P_1), \dots, H(P_{i-1}), 0, H(P_{i+1}), \dots, H(P_n)), \\ \dots \\ u_c = (H(P_1), \dots, H(P_{c-1}), 0, \dots, 0, H(P_{\varkappa+1}), \dots, H(P_n)). \end{cases}$$

Построим потомка ω. В качестве ω_i, где $1 \leqslant i \leqslant c$, из вектора u_i возьмём нуль, стоящий там на i-й позиции. Если $c \leqslant i \leqslant \varkappa$, то в качестве элемента на позиции i возьмём нуль из вектора u_c, также стоящий там на i-й позиции. Аналогично предыдущему случаю, для любой позиции j, такой, что $j > \varkappa$, точка P_j лежит в одном из классов эквивалентности S^m. Тогда $u_{m,j} = 0$, так как значение H_m равно нулю на любой точке из S^m. Значит, для любой такой позиции j мы можем выбрать $\omega_j = u_{m,j} = 0$. Комбинированием всех c векторов можно выбрать потомка ω, совпадающего с нулевым вектором:

$$|I(\omega, 0)| = n.$$

В случае *г* коалиция в силу (9) и (10) выглядит следующим образом:

$$\begin{cases} u_1 = (0, \ldots, 0, H(P_{\delta+1}), \ldots, H(P_n)), \\ \ldots \\ u_i = (H(P_1), \ldots, H(P_{\delta(i-1)}), 0, \ldots, 0, H(P_{\delta i+1}), \ldots, H(P_n)), \\ \ldots \\ u_{\lceil k/\delta \rceil} = (H(P_1), \ldots, H(P_{(\lceil \varkappa/\delta \rceil - 1)\delta,1}), 0, \ldots, 0, H(P_\varkappa), \ldots, H(P_n)), \\ \ldots \\ u_c = (H(P_1), \ldots, H(P_{c-1}), H(P_{c+1}), \ldots, H(P_n)). \end{cases}$$

Построим потомка ω в этом случае. В качестве ω_i, где $1 \leqslant i \leqslant \lceil \varkappa/\delta \rceil$, из вектора u_i возьмём нуль, стоящий там на i-й позиции. Если $\lceil \varkappa/\delta \rceil \leqslant i \leqslant \varkappa$, то в качестве элемента на позиции i возьмём нуль из вектора $u_{\lceil \varkappa/\delta \rceil}$, также стоящий там на i-й позиции. Аналогично предыдущему случаю, для любой позиции j, такой, что $j > \varkappa$, существует номер m, такой, что $u_{m,j} = 0$. Значит, для любой такой позиции j мы можем выбрать $\omega_j = u_{m,j} = 0$. Тогда комбинированием первых $\lceil \varkappa/\delta \rceil$ векторов можно выбрать потомка ω, совпадающего с нулевым вектором:

$$|I(\omega, 0)| = n.$$

Итак, при $c\delta < n$, если $c\delta \leqslant \varkappa$, найдётся такой потомок ω коалиции C_0, что $|I(\omega, 0)| \geqslant c\delta \deg(F)$, а если $c\delta > \varkappa$, то найдется потомок ω, такой, что $|I(\omega, 0)| = n$.

Таким образом, лемма в случае $c\delta < n$ доказана. Теперь рассмотрим случай, когда $c\delta \geqslant n$. Для доказательства необходимо показать, что при условиях леммы можно построить коалицию, способную породить нулевого потомка.

Построим коалицию C_0 в этом случае. Пусть $\hat{c} = \lfloor n/\delta \rfloor$, тогда $\hat{c}\delta < n$. Набор многочленов r_i, $i \in \{1, \ldots, \hat{c}\}$, и первые $\hat{c}$ элементов коалиции построим так же, как для случая $c\delta < n$ на шагах II и III. Теперь нужно достроить коалицию до необходимой мощности c.

Если на шаге II для $\hat{c}$ реализовались случаи *б*, *в* или *г*, то, как показано на шаге IV, в качестве потомка построенной коалиции мощности $\hat{c}$ уже может быть выбран нулевой вектор. Поэтому в качестве остальных $c - \hat{c}$ членов коалиции можно взять любые из оставшихся ненулевых кодовых векторов.

Предположим, что на шаге II для $\hat{c}$ реализовался случай *а*. Тогда мы можем построить потомка ω с $\hat{c}\delta \deg(F)$ нулями. Если $\hat{c}\delta \deg(F) \geqslant n$, то дополнительных построений проводить не нужно, так как в качестве потомка коалиции мощности $\hat{c}$ уже может быть выбран нулевой вектор, а в качестве остальных членов коалиции можно взять любые из оставшихся ненулевых кодовых векторов. Иначе, если $\hat{c}\delta \deg(F) < n$, то существуют $\varkappa - \hat{c}\delta > 0$ классов S^j, таких, что ни один из многочленов R_i не имеет

корней ни на одной точке — представителе каждого из этих классов. Тогда построим элемент коалиции с номером $\lceil n/\delta \rceil$ следующим образом. Возьмём проективизацию $R_{\lceil n/\delta \rceil}$ многочлена $r_{\lceil n/\delta \rceil} = (x_2 - S_{1,2}^{(\lceil n/\delta \rceil - 1)\delta + 1}) \dots (x_2 - S_{1,2}^{\varkappa})$ и поделим на X_3^{δ}, получив рациональную функцию $H_{\lceil n/\delta \rceil}$. Степень $R_{\lceil n/\delta \rceil}$ не превышает δ, так как

$$\varkappa - (\lceil n/\delta \rceil - 1)\delta - 1 + 1 \leqslant n - (\lceil n/\delta \rceil - 1)\delta \leqslant n - \lceil n/\delta \rceil \delta + \delta \leqslant \delta.$$

Очевидно, что значение $H_{\lceil n/\delta \rceil}$ на всех точках из классов $S^{(\lceil n/\delta \rceil - 1)\delta + 1}, \dots, S^k$ равно нулю. Аналогично показанному на шаге II в случае $c\delta < n$ проверяется, что $H_{\lceil n/\delta \rceil} \in L(D)$. Тогда можно построить очередной член коалиции $u_{\lceil n/\delta \rceil}$, являющийся образом функции $H_{\lceil n/\delta \rceil}$. На позициях $(\lceil n/\delta \rceil - 1)\delta + 1, \dots, n$ в $u_{\lceil n/\delta \rceil}$ находятся нули. В этом случае оставшиеся $c - \lceil n/\delta \rceil$ членов коалиции выберем как произвольные кодовые ненулевые слова, не совпадающие с построенными ранее членами коалиции. В качестве потомка построенной коалиции может быть выбран нулевой вектор. Действительно, первые $\hat{c}\delta \deg(F)$ позиций могут быть заполнены нулями согласно построению из пункта II, a, а остальные $n - \hat{c}\delta \deg(F)$ позиций можно заполнить нулями, стоящими на соответствующих позициях в векторе $u_{\lceil n/\delta \rceil}$.

Таким образом, построена коалиция $C_0 \in \mathrm{coal}_c(C \setminus \{0\})$. В качестве $\omega \in \mathrm{desc}(C_0) \setminus C_0$ можно выбрать нулевой вектор. ■

Лемма 2.

$$\left\lceil \frac{n + \alpha}{2\lfloor \alpha / \deg(F) \rfloor \deg}(F) \right\rceil \leqslant \left\lceil \frac{n}{\alpha} \right\rceil.$$

Доказательство. Действительно,

$$\frac{n + \alpha}{2\,\lfloor \alpha / \deg(F) \rfloor \deg(F)} < \frac{2n}{2\,\lfloor \alpha / \deg(F) \rfloor \deg(F)} = \frac{n}{\lfloor \alpha / \deg(F) \rfloor \deg(F)},$$

так как $\alpha < n$, и, далее, так как $\lfloor \alpha / \deg(F) \rfloor \deg(F) \leqslant \alpha$, то

$$\frac{n}{\lfloor \alpha / \deg(F) \rfloor \deg(F)} \leqslant \frac{n}{\alpha},$$

значит, и

$$\left\lceil \frac{n + \alpha}{2\,\lfloor \alpha / \deg(F) \rfloor \deg(F)} \right\rceil \leqslant \left\lceil \frac{n}{\alpha} \right\rceil.$$

Лемма доказана. ■

4.2. Доказательство теоремы 1

1. Докажем сначала справедливость неравенства

$$R_{\mathrm{FP}}(C) \leqslant \tilde{B}_{\mathrm{FP}}(C) = \min \left\{ \left\lceil \frac{n}{\lfloor \alpha / \deg(F) \rfloor \deg(F)} \right\rceil, \left\lceil \frac{\varkappa}{\delta} \right\rceil \right\}. \tag{12}$$

Пусть $\hat{c}$ — произвольное целое, такое, что $\hat{c} \geqslant \tilde{B}_{FP}$. Чтобы доказать (12), достаточно показать, что при числе злоумышленников, равном как минимум $\hat{c}$, FP-свойство не выполнено.

Сначала пусть

$$\tilde{B}_{\mathrm{FP}} = \min \left\{ \left\lceil \frac{n}{\delta \deg(F)} \right\rceil, \left\lceil \frac{\varkappa}{\delta} \right\rceil \right\} = \left\lceil \frac{n}{\delta \deg(F)} \right\rceil.$$

Тогда в силу (4) из леммы 1

$$\forall v \in C\ \exists C_0 \in \mathrm{coal}_{\hat{c}}(C \setminus \{v\})\ \exists \omega \in \mathrm{desc}(C_0) \setminus C_0 \left(|I(\omega, v)| \geqslant \min\{\hat{c}\delta \deg(F), n\}\right).$$

По предположению $\hat{c} \geqslant B_{\mathrm{FP}}$, поэтому

$$|I(\omega, v)| \geqslant \min\{\hat{c}\delta \deg(F), n\} = n,$$

т. е. $|I(\omega, v)| = n$. Значит, $\omega = v \in \mathrm{desc}(C_0) \setminus C_0$. Следовательно, FP-свойство не выполняется, и $R_{\mathrm{FP}}(C) \leqslant B_{\mathrm{FP}}(C)$.

Пусть теперь

$$\tilde{B}_{\mathrm{FP}} = \min\left\{\left\lceil \frac{n}{\delta \deg(F)} \right\rceil, \left\lceil \frac{\varkappa}{\delta} \right\rceil\right\} = \left\lceil \frac{\varkappa}{\delta} \right\rceil.$$

Тогда в силу (5) из леммы 1

$$\forall v \in C\ \exists C_0 \in \mathrm{coal}_{\hat{c}}(C \setminus \{v\})\ \exists \omega \in \mathrm{desc}(C_0) \setminus C_0 \left(|I(\omega, v)| = n\right),$$

что также означает, что FP-свойство не выполняется.

Итак, неравенство (12) доказано.

2. Докажем неравенство

$$R_{\mathrm{TA}}(C) \leqslant \tilde{B}_{\mathrm{TA}}(C) = \min\left\{\left\lceil \frac{n+\alpha}{2\delta \deg(F)} \right\rceil, \left\lceil \frac{\varkappa}{\delta} \right\rceil\right\}. \tag{13}$$

Для этого достаточно показать, что если $\hat{c}$ — произвольное целое число, такое, что $\hat{c} \geqslant \tilde{B}_{\mathrm{TA}}(C)$, то C не является c-TA-кодом.

Пусть сначала

$$\tilde{B}_{\mathrm{TA}} = \min\left\{\left\lceil \frac{n+\alpha}{2\delta \deg(F)} \right\rceil, \left\lceil \frac{\varkappa}{\delta} \right\rceil\right\} = \left\lceil \frac{n+\alpha}{2\delta \deg(F)} \right\rceil.$$

Тогда, так как $\hat{c} \geqslant B_{\mathrm{TA}}(C)$, получаем

$$c\delta \deg(F) \geqslant (n+\alpha)/2. \tag{14}$$

Пусть $v \in C$ — произвольное кодовое слово. Согласно определению свойства c-TA, для того чтобы показать, что оно не выполнено при $c = \hat{c}$, достаточно доказать следующее:

$$\exists C_0 \in \mathrm{coal}_{\hat{c}}(C)\ \exists \omega \in \mathrm{desc}(C_0)\ \forall i \in \{1, \ldots, \hat{c}\} \left(|I(\omega, u_i)| \leqslant |I(\omega, v)|\right).$$

В качестве C_0 и ω рассмотрим построенные в лемме 1 для случая II, *а* коалицию C_0 и её потомка ω и докажем, что выполняются искомые условия.

Если $\hat{c}\delta \geqslant \varkappa$, то из (4) в лемме 1 легко показать, что C не является $\hat{c}$-FP, а значит, и $\hat{c}$-TA кодом, и (13) доказано.

Пусть $\hat{c}\delta \leqslant \varkappa$. Тогда в силу (4) из леммы 1

$$|I(\omega, v)| \geqslant \min\{\hat{c}\delta \deg(F), n\}. \tag{15}$$

Если $|I(\omega, v)| \geqslant \min\{\hat{c}\delta \deg(F), n\} = n$, то легко показать, что C не является $\hat{c}$-FP-, а значит, и $\hat{c}$-TA-кодом, и (13) доказано. Поэтому далее $|I(\omega, v)| \geqslant \hat{c}\delta \deg(F)$.

Для произвольного $S \subset \{1, \ldots, n\}$ определим $I_S(u, v) = \{i \in S : u_i = v_i\}$. Пусть $A = \{1, \ldots, \hat{c}\delta \deg(F)\}$. Докажем, что

$$\forall i \in \{1, \ldots, \hat{c}\} \left(|I_A(\omega, u_i)| \leqslant \alpha\right). \tag{16}$$

Предположим противное: найдётся такой номер i_0, что $|I_A(\omega, u_{i_0})| > \alpha$. Тогда существует $r > \alpha$ позиций из A, таких, что в каждой позиции k выполняется $u_{i_0,k} = \omega_k$. Согласно построению коалиции из леммы 1, $\omega_j = v_j$ для всех $j \in A$, поэтому $u_{i_0,k} = v_k$. Значит, $|I(v, u_{i_0})| = r > \alpha$. Тогда

$$\rho(v, u_{i_0}) = n - |I(v, u_{i_0})| < n - \alpha,$$

чего в силу утверждения 1 быть не может, значит, (16) выполнено.

Докажем неравенство

$$|I(\omega, u_i)| \leqslant n - \hat{c}\delta \deg(F) + \alpha.$$

Ввиду того, что

$$I(\omega, u_i) = I_A(\omega, u_i) \cup I_{\{1,\ldots,n\}\setminus A}(\omega, u_i),$$

и неравенства (16) получаем

$$|I(\omega, u_i)| = |I_A(\omega, u_i)| + |I_{\{1,\ldots,n\}\setminus A}(\omega, u_i)| \leqslant \alpha + n - \hat{c}\delta \deg(F). \tag{17}$$

Из (14) следует, что

$$n - \hat{c}\delta \deg(F) + \alpha \leqslant \hat{c}\delta \deg(F).$$

Тогда, учитывая (15) и (17), получаем

$$|I(\omega, u_i)| \leqslant n - \hat{c}\delta + \alpha \leqslant \hat{c}\delta \leqslant |I(\omega, v)|.$$

Это означает, что C не является $\hat{c}$-ТА-кодом.

Пусть теперь

$$\tilde{B}_{\mathrm{TA}} = \min\left\{\left\lceil \frac{n+\alpha}{2\delta \deg(F)} \right\rceil, \left\lceil \frac{\varkappa}{\delta} \right\rceil\right\} = \left\lceil \frac{\varkappa}{\delta} \right\rceil.$$

В этом случае, если $\hat{c} \geqslant \tilde{B}_{\mathrm{TA}}$, то с помощью (4) из леммы 1 легко показать, что C не является $\hat{c}$-FP-, а значит, и $\hat{c}$-ТА-кодом.

Неравенство (13) доказано.

3. Теперь, согласно лемме 2, получаем, что

$$\tilde{B}_{\mathrm{TA}}(C) = \left\lceil \frac{n+\alpha}{2\lfloor \alpha/\deg(F) \rfloor \deg(F)} \right\rceil \leqslant \left\lceil \frac{n}{\alpha} \right\rceil.$$

Справедливость неравенства

$$R_{\mathrm{TA}}(C) \geqslant \sqrt{n/\alpha}$$

доказана в [6, теорема 1], а справедливость

$$R_{\mathrm{FP}}(C) \geqslant \left\lceil \frac{n}{\alpha} \right\rceil$$

показана в [13, теорема 3].

Таким образом, доказана вся цепочка неравенств (3), а вместе с тем и теорема 1.

Заключение

В работе представлен специальный класс алгебраических кривых, на которых возможно уточнение рубежей множеств компрометации для идентифицирующих c-TA- и c-FP-свойств применительно к одноточечным алгеброгеометрическим кодам. Полученные для этих рубежей оценки даны в основной теореме; приведены примеры таких кривых и вычисления рубежей. Результаты работы представляют самостоятельный теоретичекий интерес, а также могут быть использованы в практических реализациях протоколов широковещательного шифрования.

ЛИТЕРАТУРА

1. *Fiat A. and Naor M.* Broadcast encryption // LNCS. 1994. V. 773. P. 480–491.
2. *Chor B., Fiat A., and Naor M.* Tracing traitors // LNCS. 1994. V. 839. P. 257–270.
3. *Silverberg A., Staddon J., and Walker J.* Applications of list decoding to tracing traitors // IEEE Trans. Inform. Theory. 2003. V. 49. No. 5. P. 1312–1318.
4. *Stinson D. R. and Wei R.* Combinatorial properties and constructions of traceability schemes and frameproof codes // SIAM J. Discr. Math. 1998. V. 11. Iss. 1. P. 41–53.
5. *Staddon J. N., Stinson D. R., and Wei R.* Combinatorial properties of frameproof and traceability codes // IEEE Trans. Inform. Theory. 2001. V. 47. No. 3. P. 1042–1049.
6. *Загуменнов Д. В., Мкртичян В. В.* О применимости алгеброгеометрических кодов L-конструкции как кодов защиты от копирования // Прикладная дискретная математика. 2019. № 44. С. 67–93.
7. *Деундяк В. М., Мкртичян В. В.* Исследование границ применения схемы защиты информации, основанной на РС-кодах // Дискретный анализ и исследование операций. 2011. Т. 18. № 3. С. 21–38.
8. *Ma Y. and Ding Y.* Reed — Solomon codes as traceability codes with an efficient tracing algorithm // 8th Intern. Conf. Signal Processing. Guilin, China, 2006. V. 4. `https://ieeexplore.ieee.org/document/4129649`.
9. *Guruswami V. and Sudan M.* Improved decoding of Reed-Solomon and algebraic-geometric codes // Foundations of Computer Science. Palo Alto: IEEE, 1998. P. 28–37.
10. *Fernandez M., Cotrina J., Soriano M., and Domingo N.* A note about the traceability properties of linear codes // LNCS. 2007. V. 4817. P. 251–258.
11. *Деундяк В. М., Евпак С. А., Мкртичян В. В.* Исследование свойств q-ичных помехоустойчивых кодов Рида — Маллера как кодов для защиты от копирования // Проблемы передачи информации. 2015. Т. 51. № 4. С. 99–111.
12. *Pellikaan R. and Wu X-W.* List decoding of q-ary Reed — Muller codes // IEEE Trans. Inform. Theory. 2004. V. 50. No. 4. P. 679–682.
13. *Деундяк В. М., Загуменнов Д. В.* Исследование свойств АГ-кодов как кодов для защиты от копирования // Моделирование и анализ информационных систем. 2020. Т. 27. № 1. С. 22–38.
14. *Barg A. and Kabatiansky G.* A class of I.P.P. codes with efficient identification // J. Complexity. 2004. V. 20. Iss. 2–3. P. 137–147.
15. *Zagumennov D., Deundyak V., Gufan A., and Mkrtichan V.* Algebraic geometry codes for special broadcast encryption schemes in telecommunication nets // Proc. MWENT. Moscow, Russia, 2020. P. 1–6.
16. *Кабатянский Г. А.* Идентифицирующие коды и их обобщения // Проблемы передачи информации. 2019. Т. 55. № 3. С. 93–105.
17. *Moreira J., Fernandez M., and Soriano M.* On the relationship between the traceability properties of Reed — Solomon codes // Adv. Math. Commun. 2012. V. 6. No. 4. P. 467–478.

18. *Fernandez M.* Codes with traceability properties and the Silverberg — Staddon — Walker conjecture // XVI Intern. Symp. "Problems of Redundancy in Information and Control Systems". Moscow, Russia, 2019. Plenary lecture.
19. *Safavi-Naini R. and Wang Y.* New results on frame-proof codes and traceability schemes // IEEE Trans. Inform. Theory. 2001. V. 47. No. 7. P. 3029–3033.
20. *Ge G. and Shangguan Ch.* Good traceability codes do exist. `https://arxiv.org/abs/1601.04810v1`.
21. *Ge G., Ma J., and Shangguan Ch.* New results for traitor tracing schemes. `https://arxiv.org/abs/1610.07719`.
22. *Chee Y. M. and Zhang X.* Improved constructions of frameproof codes // IEEE Trans. Inform. Theory. 2012. V. 58. No. 8. P. 5449–5453.
23. *Noskov I. and Bezzateev S.* Traceability schemes usings finite geometry // 10th Intern. Congress ICUMT. Moscow, Russia, 2018. P. 1–5.
24. *Цфасман М. А, Влэдуц С. Г., Ногин Д. Ю.* Алгеброгеометрические коды. Основные понятия. М.: МЦНМО, 2003.
25. *Hoholdt T., van Lint J. H., and Pellikaan R.* Algebraic geometry codes // Handbook of Coding Theory / V. S. Pless, W. C. Huffman, and R. A. Brualdi (eds.). V. 1. Amsterdam: Elsevier, 1998. P. 871–961.

REFERENCES

1. *Fiat A. and Naor M.* Broadcast encryption. LNCS, 1994, vol. 773, pp. 480–491.
2. *Chor B., Fiat A., and Naor M.* Tracing traitors. LNCS, 1994, vol. 839, pp. 257–270.
3. *Silverberg A., Staddon J., and Walker J.* Applications of list decoding to tracing traitors. IEEE Trans. Inform. Theory, 2003, vol. 49, no. 5, pp. 1312–1318.
4. *Stinson D. R. and Wei R.* Combinatorial properties and constructions of traceability schemes and frameproof codes. SIAM J. Discr. Math., 1998, vol. 11, iss. 1, pp. 41–53
5. *Staddon J. N., Stinson D. R., and Wei R.* Combinatorial properties of frameproof and traceability codes. IEEE Trans. Inform. Theory, 2001, vol. 47, no. 3, pp. 1042–1049
6. *Zagumennov D. V. and Mkrtichyan V. V.* O primenimosti algebrogeometricheskikh kodov *L*-konstruktsii kak kodov zashchity ot kopirovaniya [On application of algebraic geometry codes of *L*-construction in copy protection]. Prikladnaya Diskretnaya Matematika, 2019, no. 44, pp. 67–93. (in Russian)
7. *Deundyak V. M. and Mkrtichyan V. V.* Issledovaniye granits primeneniya skhemy zashchity informatsii, osnovannoy na RS-kodakh [Research of applying bounds of the information protection scheme based on PS-codes]. Diskretn. Anal. Issled. Oper., 2011, vol. 18, iss. 3, pp. 21–38 (in Russian)
8. *Ma Y. and Ding Y.* Reed — Solomon codes as traceability codes with an efficient tracing algorithm. 8th Intern. Conf. Signal Processing, Guilin, China, 2006, vol. 4. `https://ieeexplore.ieee.org/document/4129649`.
9. *Guruswami V. and Sudan M.* Improved decoding of Reed — Solomon and algebraic-geometric codes. Foundations of Computer Science. Palo Alto, IEEE, 1998, pp. 28–37.
10. *Fernandez M., Cotrina J., Soriano M., and Domingo N.* A note about the traceability properties of linear codes. LNCS, 2007, vol. 4817, pp. 251–258.
11. *Deundyak V. M., Evpak S. A., and Mkrtichyan V. V.* Analysis of properties of q-ary Reed — Muller error-correcting codes viewed as codes for copyright protection. Problems Inform. Transmission, 2015, vol. 51, no. 4, pp. 398–408.
12. *Pellikaan R. and Wu X-W.* List decoding of q-ary Reed — Muller codes. IEEE Trans. Inform. Theory, 2004, vol. 50, no. 4, pp. 679–682.

13. *Deundyak V. M. and Zagumennov D. V.* Issledovanie svoystv AG-kodov kak kodov dlya zashchity ot kopirovaniya [On the properties of algebraic geometric codes as copy protection codes]. Modelirovanie i Analiz Informatsionnykh Sistem, 2020, vol. 27, no. 1, pp. 22–38. (in Russian)
14. *Barg A. and Kabatiansky G.* A class of I.P.P. codes with efficient identification. J. Complexity, 2004, vol. 20, iss. 2–3, pp. 137–147.
15. *Zagumennov D., Deundyak V., Gufan A., and Mkrtichan V.* Algebraic geometry codes for special broadcast encryption schemes in telecommunication nets. Proc. MWENT, Moscow, Russia, 2020, pp. 1–6.
16. *Kabatyansky G. A.* Traceability codes and their generalizations. Problems Inform. Transmission, 2019, vol. 55, no. 3, pp. 283–294.
17. *Moreira J., Fernandez M., and Soriano M.* On the relationship between the traceability properties of Reed — Solomon codes. Adv. Math. Commun., 2012, vol. 6, no. 4, pp. 467–478.
18. *Fernandez M.* Codes with traceability properties and the Silverberg — Staddon — Walker conjecture. XVI Intern. Symp. "Problems of Redundancy in Information and Control Systems", Moscow, Russia, 2019, Plenary lecture.
19. *Safavi-Naini R. and Wang Y.* New results on frame-proof codes and traceability schemes. IEEE Trans. Inform. Theory, 2001, vol. 47, no. 7, pp. 3029–3033.
20. *Ge G. and Shangguan Ch.* Good traceability codes do exist. `https://arxiv.org/abs/1601.04810v1`.
21. *Ge G., Ma J., and Shangguan Ch.* New results for traitor tracing schemes. `https://arxiv.org/abs/1610.07719`.
22. *Chee Y. M. and Zhang X.* Improved constructions of frameproof codes. IEEE Trans. Inform. Theory, 2012, vol. 58, no. 8, pp. 5449–5453.
23. *Noskov I. and Bezzateev S.* Traceability schemes usings finite geometry. 10th Intern. Congress ICUMT, Moscow, Russia, 2018, pp. 1–5.
24. *Tsfasman M. A, Vleduts S. G., and Nogin D. Yu.* Algebrogeometricheskie kody. Osnovnye ponyatiya [Algebraic Geometric Codes. Basic Notions]. Moscow, MCCME Publ., 2003. (in Russian)
25. *Hoholdt T., van Lint J. H., and Pellikaan R.* Algebraic geometry codes. Handbook of Coding Theory, V. S. Pless, W. C. Huffman, and R. A. Brualdi (eds.), vol. 1, Amsterdam, Elsevier, 1998, pp. 871–961.

UDC 003.26, 519.725, 519.176 DOI 10.17223/20710410/53/5

CHARACTERISTICS OF HADAMARD SQUARE OF SPECIAL REED — MULLER SUBCODES

V. Vysotskaya

Moscow State University, Moscow, Russia
JSC "NPK Kryptonite", Moscow, Russia

E-mail: vysotskaya.victory@gmail.com

The existence of some structure in a code can lead to the decrease of security of the whole system built on it. Often subcodes are used to "disguise" the code as a "general-looking" one. However, the security of subcodes, whose Hadamard square is equal to the square of the original code, can be reduced to the security of this code. The paper finds the limiting conditions on the number of vectors of degree r whose removing retains this weakness for Reed — Muller subcodes and, accordingly, conditions for it to vanish. For $r = 2$ the exact structure of all resistant subcodes has been found. For an arbitrary code $\mathrm{RM}(r, m)$, the desired number of vectors to remove for providing the security has been estimated from both sides. Finally, the ratio of subcodes with Hadamard square unequal to the square of the original code has been proved to tend to zero if additional conditions on the codimension of the subcode and the parameter r are imposed and $m \to \infty$. Thus, the implementation of checks proposed in the paper helps to immediately filter out some insecure subcodes.

Keywords: *post-quantum cryptography, code-based cryptography, Reed — Muller codes, Reed — Muller subcodes, Hadamard product, McEliece cryptosystem.*

1. Introduction

The security of most standardized cryptographic algorithms used all around the world is based on the complexity of several number-theoretical problems. They usually are the discrete logarithm or factorization problem. However, in 1994 P. Shor showed [1] that quantum computers could break all schemes constructed in this way. And in 2001 the Shor's algorithm was implemented on a 7-qubit quantum computer. Since then various companies have been actively developing more powerful quantum computers. Progress in this area poses a real threat to modern public-key cryptography.

There are several approaches to build post-quantum cryptographic schemes. One approach is to use error-correcting codes. No successful quantum-computer attacks on "hard" problems from this area are known. Classical examples of code-based schemes are the McEliece cryptosystem [2] and the Niederreiter cryptosystem [3], which are equivalent in terms of security.

The interest in code-based schemes as post-quantum can be noticed while analyzing the works submitted to the contest for prospective public-key post-quantum algorithms which was announced in 2016 by the US National Institute of Standards and Technology (NIST) [4]. The algorithms that win this contest will be accepted as US national standards. 21 of 69 applications filed (that is, almost a third of all works) were based on coding theory. Despite the fact that some of them were attacked, it seems that this approach looks quite promising and deserves further study and development. This interest is also traced in Russian cryptography. Code-based schemes were chosen by the Technical Committee for

Standardization "Cryptographic and Security Mechanisms" (TC 26) as one of directions in developing draft Russian national standards of post-quantum cryptographic algorithms.

When one is facing the challenge to synthesize a new code-based scheme, the first thing to think about is the choice of basic code. Some schemes do not specify the code, thus leaving it to the discretion of the user. Such schemes are usually more reliable since their security is often directly reduced to NP-complete problems. Most often, these problems are decoding and syndrome decoding. However, choosing a special code also has some advantages. For example, such codes provide asymmetric complexity in solving the decoding problem for the legal user and adversary. In addition, due to the structure of the code, the sizes of the public keys can be significantly reduced.

However, the structure can also cause a significant decrease in security of the code, therefore one of the most important tasks is to "disguise" the code as a "general-looking" one. One solution is to use subcodes. This approach allows to "destroy" the structure of the code, retaining the ability to work with the result in mostly the same way as with the original one. Nevertheless, it is worth considering that many of proposed systems based on subcodes turned out to be vulnerable. So in [5, 6] C. Wieschebrink built efficient attacks on some special cases of the Berger — Loidreau cryptosystem [7], which is based on subcodes of the Reed — Solomon code. The McEliece cryptosystem based on subcodes of algebraic geometry codes was attacked in [8]. First version of digital signature pqsigRM [9] based on modified Reed — Muller codes, which was submitted at the NIST contest, was also attacked during the peer review.

One of the mechanisms for analyzing codes with a hidden structure is the use of the technique of Hadamard product of two codes. This method was used by M. Borodin and I. Chizhov [10] to improve Minder — Shokrollahi attack [11] on the McEliece cryptosystem based on Reed — Muller codes. In [12] this technique allowed Chizhov and Borodin to reduce the security of the cryptosystem on subcodes of Reed — Muller codes of codimension one to the security of the scheme on full codes. Recall that codimention means the number of vectors missing in the code basis. In [13] the distinguisher between random codes and Reed — Solomon codes using Hadamard product is described.

In this paper, the mentioned technique is used to analyze Reed — Muller subcodes in standard basis without restriction on codimension. The main question is: which Reed — Muller subcodes do not allow Chizhov — Borodin's approach. Since the reduction can be performed to a subcode whose Hadamard square coincides with the square of the original code, we look for conditions under which this equality ceases to hold. Codes obtaining these conditions will be called *unstable codes*, the others — *stable codes*. In addition, we compute the probability that a randomly chosen Reed — Muller subcode is unstable.

In Section 2, the exact structure of all stable subcodes of $\mathrm{RM}(2, m)$ is found. Thus, to provide the security, it is necessary to choose at least another subcode. To be sure that a subcode of $\mathrm{RM}(2, m)$ is unstable, it is sufficient to exclude $m+1$ monomials of degree 2 from it's standard basis. For an arbitrary Reed — Muller code $\mathrm{RM}(r, m)$, in Section 3 we estimate (both from the above and below) the number of vectors of degree r that must be excluded from the basis of the code in order to distort its square. Finally, in Section 4 we show that the ratio of unstable subcodes tends to zero (as $m \to \infty$) given some additional conditions on the codimension of the subcode and the parameter r. Thus, it is not enough to choose an arbitrary Reed — Muller subcode when synthesizing a real scheme. It is necessary to check the property formulated below as Proposition 4. At the same time subcodes satisfying this property require additional consideration since they may have some special structure.

2. The structure of stable $\mathbf{RM}(2, m)$ subcodes

Recall that *Reed — Muller code* $\mathrm{RM}(r, m)$ is the set of all Boolean functions f in m variables such that $\deg(f) \leqslant r$. Consider the code $\mathrm{RM}(1, m)$. We look for the minimum number of monomials $f_1, \ldots, f_{w(m,2)}$ of degree 2 such that the code

$$\operatorname{span}\big(\mathrm{RM}(1, m) \cup \{f_1, \ldots, f_{w(m,2)}\}\big) \tag{1}$$

is *stable*, i.e.,

$$\big(\operatorname{span}(\mathrm{RM}(1, m) \cup \{f_1, \ldots, f_{w(m,2)}\})\big)^2 = \mathrm{RM}(4, m). \tag{2}$$

Here, the squaring operation refers to the squaring of Hadamard. *Hadamard product* of two vectors is a vector obtained as a result of component-wise product of coordinates:

$$(a_1, \ldots, a_n) \circ (b_1, \ldots, b_n) = (a_1 b_1, \ldots, a_n b_n),$$

and Hadamard product of two codes A and B is the span of all pairwise products of form $a \circ b$, where $a \in A$, $b \in B$. Codes that do not satisfy condition (2) we will denote *unstable*.

We consider Reed — Muller codes spanned by their standard basis. *The standard basis of the Reed — Muller code* $\mathrm{RM}(r, m)$ includes all monomials of m variables of degree from 0 to r inclusively, i.e.,

$$1,\ x_1,\ x_2,\ \ldots,\ x_m,\ x_1x_2,\ \ldots,\ x_{m-1}x_m,\ \ldots,\ x_1 \ldots x_r,\ \ldots,\ x_{m-r-1} \ldots x_m.$$

Obviously, after finding the minimal number $w(m, 2)$ of monomials f_i, one can also answer another question: what is the maximum number $q(m, 2)$ of monomials of degree 2 that can be removed from the basis of the code $\mathrm{RM}(2, m)$ so that the code

$$\operatorname{span}\big(\{1, x_1, x_2, \ldots, x_m, x_1x_2, \ldots, x_{m-1}x_m\} \setminus \{g_1, \ldots, g_{q(m,2)}\}\big)$$

is still stable. The relation between these values is given by the following equality:

$$q(m, 2) = \binom{m}{2} - w(m, 2). \tag{3}$$

And so, after removing $q(m, 2) + 1 = \binom{m}{2} - w(m, 2) + 1$ basis vectors, one gets an unstable code. Therefore, we will not dwell on this issue separately.

Now let us proceed to the graph interpretation of this problem. We match a subcode $\mathcal{A} \subset \mathrm{RM}(2, m)$ with a graph $G = (V, E)$ with vertex set $V = \{x_1, \ldots, x_m\}$ and edge set E; $\{x_i, x_j\} \in E \Leftrightarrow x_i x_j \in \mathcal{A}$.

Let us denote by $\deg(v)$ the degree of vertex v in the graph. As vertices are monomials, it can be a little embarrassing, but we never use $\deg(\cdot)$ to refer to the degree of a polynomial.

We will say that a graph with m vertices *has the property* P if

1) the degree $\deg(v)$ of any vertex v is not less than $m - 3$;
2) if $\deg(v) = m - 3$ and $\{v, u_1\} \notin E$, $\{v, u_2\} \notin E$, then $\{u_1, u_2\} \in E$.

The case $\deg(x_1) = m - 3$ is shown in Fig. 1, where lines denote graph edges.

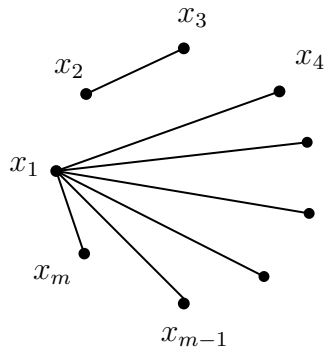

Fig. 1. Case $\deg(x_1) = m - 3$

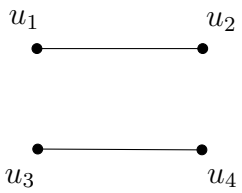

Fig. 2. Graph H

Theorem 1. For any $m \geqslant 4$ a subcode of $\mathrm{RM}(2, m)$ of the form (1) is stable if and only if the property P holds for the corresponding graph.

Proof. Denote $G = (V, E)$ the graph corresponding to the subcode of form (1). Note that the condition (2) is equivalent to the condition that any induced subgraph of G with 4 vertices has a subgraph isomorphic to the graph H shown in Fig. 2. The edges $\{u_1, u_2\}$ and $\{u_3, u_4\}$ correspond to degree-2 monomial used to produce the monomial $u_1u_2u_3u_4$. Also, note that to show that the subcode (1) is stable it is enough to prove that any monomial of degree 4 can be represented as a product of two monomials from this code. Then the same is automatically true for all monomials of degree 3. Indeed, for any monomial $u_1u_2u_3$ at least one of monomials u_1u_2, u_1u_3 or u_2u_3 lie in the code. Otherwise, no monomial of form $u_1u_2u_3v$ could be obtained after squaring. Degree-1 monomials are in the code by the definition.

To prove the necessity, we fix any vertex v. If any three incident edges $\{v, u_j\}$ for $j = 1, 2, 3$ are missing, then the induced subgraph on vertices v, u_1, u_2, u_3 would not have the required subgraph H. The contradiction proves that $\deg(v) \geqslant m - 3$. If, however, $\deg(v) = m - 3$ and $\{v, u_1\} \notin E$, $\{v, u_2\} \notin E$, then $\{u_1, u_2\} \in E$, as otherwise none of the induced 4-vertex subgraphs containing vertices v, u_1 and u_2 will have the required subgraph. Thus, the property P holds.

The sufficiency: fix any induced subgraph with 4 vertices (let us denote them v, u_1, u_2 and u_3). Note that it has the property P for $m = 4$. If vertex v has degree 1, i.e., $\{v, u_1\} \in E$, but $\{v, u_2\} \notin E$, $\{v, u_3\} \notin E$, then by the property P it follows that $\{u_2, u_3\} \in E$. Thus, we have edges $\{v, u_1\}$ and $\{u_2, u_3\}$ necessary for the H-isomorphic subgraph.

If all 4 vertices in the subgraph have degree at least 2, then there is a simple cycle of length 3 or 4. If it has length 4, the presence of H-isomorphic subgraph is obvious. Otherwise, we have a triangle $\{u_1, u_2, u_3\}$ and, moreover, the fourth vertex v has degree at least 2. Assume (without loss of generality) that $\{v, u_1\} \in E$, then for H-isomorphic subgraph we can take the edges $\{v, u_1\}$ and $\{u_2, u_3\}$. ■

From Theorem 1, the minimum number of edges is obtained if a graph has the property P and the degree of each vertex is $m - 3$. It remains to describe such graphs.

Proposition 1. Assume $m \geqslant 4$. If the property P holds for some graph G with m vertices such that the degree of each vertex is $m - 3$, then the complementary graph $\overline{G}$ is a union of cycles of length at least 4.

Proof. Since the degree of each vertex of graph G is $m - 3$, the degree of each vertex of $\overline{G}$ is 2. Moreover, from the second item of the property P follows that if $\overline{G}$ contains edges

$\{v, u_1\}$ and $\{v, u_2\}$, then it does not contain the edge $\{u_1, u_2\}$. So graph $\overline{G}$ is triangle-free. Choose an arbitrary vertex u_1. It is not isolated, therefore, one can select a vertex adjacent to it. Let us call it u_2. As $\deg(u_2) = 2$, there exists some adjacent vertex $u_3 \neq u_1$. Continue in this way until u_j coincides with one of $u_1, \ldots, u_{j-1}$. Note that u_j cannot coincide with u_i for $i > 1$ as it would mean that $\deg(u_i) \geqslant 3$. Thus, $u_1, \ldots, u_{j-1}$ form a simple cycle. Its length is at least 4, as $\overline{G}$ is triangle-free. ■

Thus, we have described the structure of the graph corresponding to the minimal stable subcode of form (1). Now let us describe the complete structure of such codes. Let us denote a *bamboo graph* a tree without branching (having no vertices of degree greater than 2).

Proposition 2. Assume $m \geqslant 4$. If the property P holds for some graph G with m vertices, then the complementary graph $\overline{G}$ is a union of cycles of length at least 4 and bamboo graphs.

Proof. We proceed as in Proposition 1 and try to find a cycle in $\overline{G}$. But we can stop in a vertex of degree 1, thus obtaining a bamboo graph. Isolated vertices are bamboo graphs by definition. ■

Corollary 1. For any $m \geqslant 4$, it holds that

$$w(m, 2) = m(m-3)/2.$$

Proof. As it was already mentioned after Theorem 1, we need to consider the subcodes corresponding to graphs with property P where degree of each vertex is $m - 3$. From Proposition 1 it follows that $\overline{G}$ has exactly m edges. Thus, G has at least $\binom{m}{2} - m = m(m-3)/2$ edges. Moreover, it means that after removing any m edges from a complete graph (corresponding to the full Reed — Muller code) we still obtain a stable code. ■

Note that, according to (3), removing $m + 1$ or more monomials of degree 2 from the basis of the code $\mathrm{RM}(2, m)$ leads to an unstable code.

3. Lower and upper bounds for minimal stable RM(r, m) subcode sizes

In this Section we carry out argument for $r > 2$. That is, we look for the minimum number $w(m, r)$, such that the code

$$\mathrm{span}\big(\mathrm{RM}(r-1, m) \cup \{f_1, \ldots, f_{w(m,r)}\}\big) \tag{4}$$

is stable. Here, f_i is a monomial of degree r. We match a subcode $\mathcal{A} \subset \mathrm{RM}(r, m)$ with a hypergraph $G = (V, E)$ with vertex set $V = \{x_1, \ldots, x_m\}$; an r-edge $\{x_{i_1}, \ldots, x_{i_r}\}$ is in E if and only if $x_{i_1} \ldots x_{i_r} \in \mathcal{A}$. In the general case, the condition similar to having an H-isomorphic subgraph in each 4-vertex induced subgraph is equivalent to condition of the code (4) being stable. Namely, each set of $2r$ vertices must be covered by two disjoint r-edges. Let us denote a graph satisfying this condition by *stable graph*. Note about covering monomials of lower degrees is the same as in the case of $r = 2$.

We can also extend relation (3) from Section 2 as:

$$q(m, r) = \binom{m}{r} - w(m, r).$$

And again we will not dwell on this issue separately.

We will use terms "graph" and "hypergraph" interchangeably. Denote $w(r, m)$ the minimal number of degree-r monomials needed to make subcode (4) stable, or, alternatively, minimal number of edges in a stable r-hypergraph with m vertices.

Proposition 3. For any natural r and $m \geqslant 2r$, it holds that

$$w(m,r) \geqslant \binom{m}{2r} \Big/ \binom{m-r}{r}.$$

Proof. Note that any set of $2r$ vertices in a stable graph contains at least one edge. Moreover, any edge is contained in exactly $\binom{m-r}{r}$ such sets. Thus, total number of edges multiplied by $\binom{m-r}{r}$ is at least number of all sets of $2r$ vertices, which is $\binom{m}{2r}$. This gives the necessary bound. ■

Corollary 2. Any stable graph contains at least $1/\binom{2r}{r}$ edges of a complete graph.

Proof. The total possible number of r-edges in a graph with m vertices is C_m^r. Then

$$\frac{\binom{m}{2r}}{\binom{m-r}{r}\binom{m}{r}} = \frac{(r!)^2}{(2r)!} = 1/\binom{2r}{r}.$$

Corollary 2 is proven. ■

This lower bound can be improved by the following theorem.

Theorem 2. For any natural r and $m \geqslant 2r$, it holds that

$$w(m,r) \geqslant \frac{1}{2}\left(\sqrt{(\gamma+1)^2 + 8\binom{m}{2r}} + \gamma + 1\right), \text{ where } \gamma = \sqrt{\sum_{u=\max\{1,3r-m\}}^{r-1} \binom{r}{u}}.$$

Proof. Fix smallest set of edges E such that every $2r$ vertices are covered by two disjoint edges from E. By definition, $|E| = w(m,r)$.

Fix any edge $e \in E$. Denote E_e the set of edges from E that intersect e and P_e — the set of unordered pairs $\{e', e''\}$, $e', e'' \in E_e$. Each pair $\{e', e''\}$ corresponds to the subset $B \subset e$, $B = (e' \cup e'') \cap e$. In the similar manner, each edge in E_e corresponds to the subset $B = e' \cap e$. On the other hand, let us fix any subset $B \subset e$ of size

$$\max\{1, 3r-m\} \leqslant |B| \leqslant r-1. \tag{5}$$

As $|B| \geqslant 3r-m$, we have $|V \setminus e| + |B| \geqslant 2r$, and thus there exists a set S such that $|S| = 2r$ and $S \cap e = B$. By the assumption on the edge set E, it contains a pair of edges covering S. Let us denote these edges e' and e''. There are two possible cases: either both e' and e'' intersect e or only one of them does. Thus, we can match the subset B with an element of $E_e \cup P_e$. Note that despite that the subset B can match several elements of $E_e \cup P_e$, the inverse mapping is single-valued, as we explained earlier. Thus, we can write

$$|P_e| + |E_e| \geqslant \sum_{u=\max\{1,3r-m\}}^{r-1} \binom{r}{u} = \gamma^2,$$

where the right-hand side is the number of all subsets $B \subset e$ satisfying (5).

Obviously, $|P_e| = \binom{|E_e|}{2}$, and thus

$$\binom{|E_e|}{2} + |E_e| \geqslant \gamma \;\Leftrightarrow\; |E_e|^2 + |E_e| \geqslant 2\gamma^2.$$

From this inequality it follows that $|E_e| \geqslant \gamma$ (technically, we use the fact that γ can not lie in the interval $(0, 1)$ following from its definition as a square root of 0 or a natural number).

Now we can estimate the cardinality of the set P of all unordered pairs $\{e', e''\}$ of edges from E. Denote $\widehat{P}$ the set of all *disjoint* unordered pairs of edges from E. It is clear that

$$P = \widehat{P} \cup \bigcup_{e \in E} \big\{\{e', e\} : e' \in E_e\big\}$$

and, moreover,

$$|P| = |\widehat{P}| + \frac{1}{2}\sum_{e \in E} |E_e|,$$

as $\widehat{P}$ is disjoint with the other set and in the union over all $e \in E$ we count each intersecting unordered pair exactly twice.

From the property that edges from E cover each set of size $2r$ we conclude that $|\widehat{P}| \geqslant \binom{m}{2r}$. Thus,

$$|P| - \frac{1}{2}\sum_{e \in E} |E_e| \geqslant \binom{m}{2r}.$$

As $|P| = \binom{|E|}{2} = \binom{w(m,r)}{2}$, we can write

$$\binom{w(m,r)}{2} - \frac{w(m,r)}{2}\gamma \geqslant \binom{m}{2r}.$$

Solving the square inequality

$$w(m,r)^2 - w(m,r)(\gamma + 1) - 2\binom{m}{2r} \geqslant 0,$$

we obtain the state of the theorem. ■

Now let us proceed to the proof of the upper bound. Let us fix the set of maximal size $\mathcal{S}$ consisting of sets $S_i \subset V$ of size $2r$ such that

$$\max_{i,j} |S_i \cap S_j| \leqslant h.$$

Lemma 1. If $h < r/3$, then for any set $Q \notin \mathcal{S}$, $|Q| = 2r$, there are at most two sets from $\mathcal{S}$ such that their intersection with Q have size at least r.

Proof. Assume that Q intersects with at least 3 sets such that intersection size is at least r. Without loss of generality we assume that the sets are S_1, S_2 and S_3. Let us denote $Q \cap S_1 = A_1$, $Q \cap S_2 = A_2$, $Q \cap S_3 = A_3$. Since $|Q| = 2r$, then it is obvious that $|A_1 \cup A_2 \cup A_2| \leqslant 2r$. On the other hand, according to the inclusion-exclusion formula,

$$|A_1 \cup A_2 \cup A_2| \geqslant |A_1| + |A_2| + |A_3| - |A_1 \cap A_2| - |A_1 \cap A_3| - |A_2 \cap A_3|.$$

Then

$$\sum_{i=1}^{3} |A_i| \leqslant 2r + 3h.$$

By condition $|A_i| \geqslant r$ for any $i \in \{1, 2, 3\}$, therefore

$$\sum_{i=1}^{3} |A_i| \geqslant 3r.$$

Whence $3r \leqslant 2r + 3h$ and $h \geqslant r/3$, which contradicts the condition of the lemma. ■

Let us find the maximum possible number of edges that can be removed from the complete graph using the above arguments such that the graph remains stable.

Theorem 3. For any natural $r \geqslant 2$, $m \geqslant 2r$, and $h < r/3$,

$$w(m, r) \leqslant \binom{m}{r} - T(r, m, h)\left(\binom{2r}{r} - 2\right),$$

where

$$T(r, m, h) = \max\big\{t : \exists S_1, \ldots, S_t \big(S_i \subset \{1, \ldots, m\} \;\&$$
$$\& |S_i| = 2r \;\&\; (i \neq j \Rightarrow |S_i \cap S_j| \leqslant h),\; i, j \in \{1, \ldots, t\}\big)\big\}.$$

Proof. Note that two disjoint r-edges are sufficient to cover a set of $2r$ vertices. Thus, it is possible to remove $\delta = \left(\binom{2r}{r} - 2\right)$ r-edges from the complete graph on the $2r$ vertices and preserve the stability of it. Obviously, no more edges can be removed.

Suppose that δ edges are removed from each set from $\mathcal{S}$ so that all of them are covered by at least two r-edges. It remains to verify that there exists a similar cover for *any* set of $2r$ vertices. Since by construction we can certainly cover any set S_i, we will prove that we can also cover any set $Q \notin S$, $|Q| = 2r$.

Note that if the cardinality of the intersection with some S_i does not exceed $(r - 1)$, then removing edges in it does not affect the number of edges in Q. At the same time, according to Lemma 1, for $h < r/3$ any set of size $2r$ can have intersection of size at least r with no more than two sets from $\mathcal{S}$. If there is only one such set, say S_1, then we have two cases:

1) $|Q \cap S_1| = 2r - 1$. In this case there exists some edge $e_1 \in Q \cap S_1$ not containing vertex v, $\{v\} = S_1 \setminus Q$ (as S_1 must be covered by two disjoint edges). Thus, we can take $e_2 = Q \setminus e_1$ (note that $e_2 \in E$ as we have removed only edges contained inside sets S_i), and $\{e_1, e_2\}$ form the disjoint cover of Q.
2) $|Q \cup S_1| < 2r - 1$. In this case there are at least two vertices v_1 and v_2 inside $Q \setminus S_1$ and the cover can be formed using any two disjoint edges $e_1, e_2 \subset Q$ such that $v_1 \in e_1$, $v_2 \in e_2$.

Now consider the case when there are exactly two sets S_1 and S_2 intersecting with Q at no less than r vertices. Assume that $|A_1| > r + h$. Then, according to the inclusion-exclusion formula, $|A_1 \cap A_2| = |A_1| + |A_2| - |A_1 \cup A_2| > r + h + r - 2r = h$, that contradicts with $|S_1 \cap S_2| \leqslant h$. Thus, $r \leqslant |A_i| \leqslant r + h$ for $i \in \{1, 2\}$. So there are at most $2\binom{r+h}{r}$ edges removed from Q. Note that

$$\binom{2r}{r} \Big/ \left(2\binom{r+h}{r}\right) = \frac{(2r)!\, r!\, h!}{2r!\, r!\, (r+h)!} = \frac{1}{2} \cdot \frac{2r}{r+h} \cdot \frac{2r-1}{r+h-1} \cdots \frac{r+1}{h+1}.$$

The last multiplier is greater than 2 for $r > 3$. For others holds

$$\frac{2r-i}{r+h-i} > \frac{2r}{r+h} > \frac{6}{4}.$$

Thus, for $r > 3$,

$$\binom{2r}{r} \Big/ \left(2\binom{r+h}{r}\right) > \frac{1}{2}\left(\frac{3}{2}\right)^{r-1} \cdot 2 > 2.$$

For $r = 2$ and 3 the inequality can be verified directly.

There are $\binom{2r}{r}\Big/2$ pairs of disjoint edges inside Q, so there remains at least one such pair after removal of $2\binom{r+h}{r} < \binom{2r}{r}\Big/2$ edges from Q.

So we have obtained a stable graph removing δ edges from a complete graph for each set from $\mathcal{S}$. It remains to remember that $|\mathcal{S}|$ is the number of sets of size $2r$ whose intersections are not larger than h and thus $|\mathcal{S}| = T(r, m, h)$. ■

Remark 1. In [14], P. Erdös and J. Spencer introduce the value $\mathbf{m}(n, k, t)$ (typeset in bold to avoid confusion with m). It determines the size of the largest set of k-element subsets of $\{1, \ldots, n\}$ such that any two members of this set intersect in less than t elements. Later V. Rödl [15] proves that

$$\lim_{n\to\infty} \mathbf{m}(n, k, t) = \binom{n}{t} \Big/ \binom{k}{t}.$$

That is, in our case, $\lim\limits_{m\to\infty} T(r, m, h) = \lim\limits_{m\to\infty} \mathbf{m}(m, 2r, \lfloor r/3 \rfloor) = \binom{m}{\lfloor r/3 \rfloor} \Big/ \binom{2r}{\lfloor r/3 \rfloor}$.

The upper bound can be also improved, but only empirically. We introduce an algorithm that on input set of vertices V returns a set of edges $E \subset V \times V$ such that in resulting graph each set of $2r$ vertices is covered by two disjoint r-edges. Its simplified form is presented in Algorithm 1. You can find the full version at `https://github.com/VysotskayaVictory/StableGraphGreedy/`.

Algorithm 1. Greedy r-covering

Input: set of vertices V, edge cardinality r.
Output: set E of r-edges that covers V.
Function ChooseEdge:
  $e := \varnothing$.
  For $i = 1, \ldots, r$:
    $V' := \{v \in V : v \text{ can be added to } e\}$;
    $v := \arg\min\limits_{v \in V'} \deg v$; $e := e \cup \{v\}$.
  Return e.
Function Main:
  $E := \varnothing$.
  While E does not cover all vertices of V:
    $e :=$ `ChooseEdge()`; $E := E \cup \{e\}$.
  Return E.

To finalize Section 3, we can compare all obtained bounds. On Fig. 3 one can see two lower and two upper bounds that are obtained in Proposition 3, Theorem 2, Theorem 3 and Algorithm 1. The Figure shows that improved bounds are rather tight.

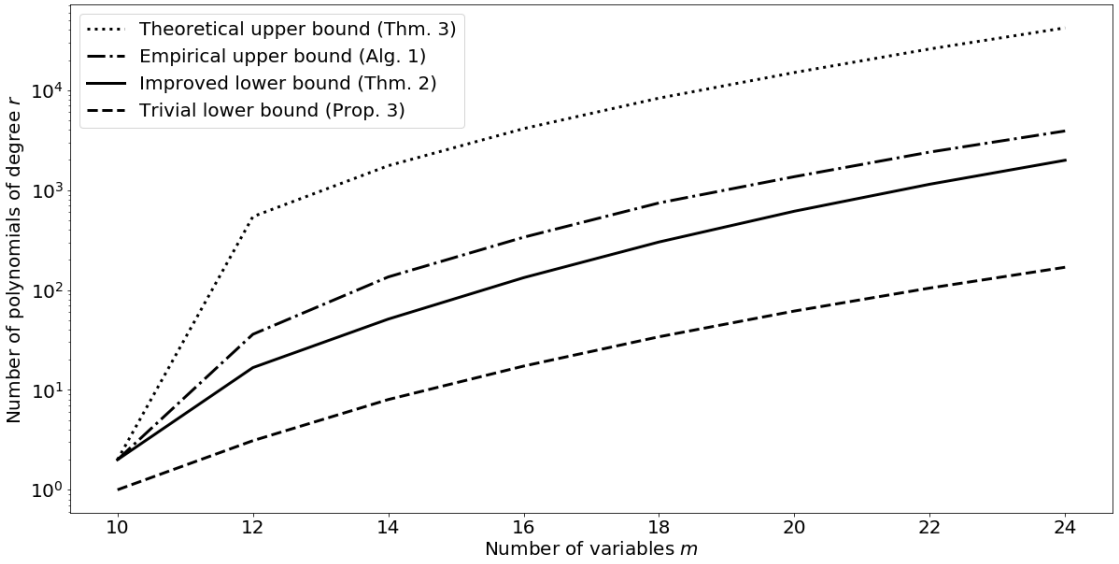

Fig. 3. Comparison of bounds

4. The ratio of unstable RM(r, m) subcodes

We consider subcodes of the standard basis of the Reed — Muller code in which ℓ vectors are missing.

For the given parameter s and the set $I = \{i_j : j = 1, \ldots, s\}$ we call unordered pairs $\{A, B\}$ *critical partition* if:

$$A \cap B = \varnothing,$$
$$A \cup B = I,$$
$$1 \leqslant |A|, |B| \leqslant r.$$

Then it is impossible to obtain the monomial $x_{i_1} \ldots x_{i_s}$ after squaring a subcode if and only if at least one element of each critical partition is removed. This follows from the fact that if this monomial is present in the square of the code, it should be formed of a pair $\{A, B\}$ from the appropriate critical partition. But by the hypothesis either A or B is absent. This argument proves the following proposition.

Proposition 4. A code is an unstable RM(r, m) subcode if and only if at least one element from each critical partition for some monomial $x_{i_1} \ldots x_{i_s}$ is removed.

Proposition 5. For the given parameter s and any set I of size s the number of critical partitions of I is

$$v(s) = \frac{1}{2} \sum_{p=\max\{s-r,1\}}^{\min\{r,s-1\}} \binom{s}{p}.$$

Proof. On the one hand, the sizes of the subsets must not exceed r. On the other hand, the partition must be non-trivial, that is, partitioning into an empty set and a set, coinciding with I, is unacceptable. Finally, when considering all partitions, each pair is counted twice. ■

Let us order in some way (say, lexicographically) the elements of each critical partition and then the critical partitions themselves. Now we consider any set M consisting of elements of critical partitions and having the property that for every critical partition M contains at least one element of this partition. We can encode M with a string $\alpha \in \{1,2,3\}^{v(s)}$, where

$$\alpha_j = \begin{cases} 1 & \Leftrightarrow \text{ the 1st element of the } j\text{-th pair lies in } M, \\ 2 & \Leftrightarrow \text{ the 2nd element of the } j\text{-th pair lies in } M, \\ 3 & \Leftrightarrow \text{ both elements of the } j\text{-th pair lie in } M. \end{cases}$$

We will also write $M(\alpha)$ to denote the set corresponding to a given $\alpha \in \{1,2,3\}^{v(s)}$. It can be seen that

$$|M(\alpha)| = \#_1(\alpha) + \#_2(\alpha) + 2 \cdot \#_3(\alpha),$$

where $\#_c(\alpha)$ is the number of symbols c in the string α.

Let us denote $k = \sum\limits_{p=0}^{r} \binom{m}{p}$ the dimension of the original code (or the number of vectors in its standard basis). There are exactly two kinds of unstable subcodes: those containing monomial 1 and those not containing it. There are $\binom{k-1}{\ell-1}$ subcodes of the second kind.

Now we fix s, an index set I of size s, and a string $\alpha \in \{1,2,3\}^{v(s)}$. Among the subcodes of the first kind there are

$$\binom{k-1-2v(s)}{\ell-|M(\alpha)|}$$

ones that has the property: among the monomials comprising critical partitions for I exactly monomials from $M(\alpha)$ are absent. The reason is that we need to choose $\ell - |M(\alpha)|$ monomials from all monomials of degree more than 0 that do not comprise any critical partition (there are $k-1-2v(s)$ of them).

For a given s there are $\binom{m}{s}$ variants of choosing index set I. But some codes may be counted several times. So we can consider the following theorem proved.

Theorem 4. The number of unstable $RM(r,m)$ subcodes is

$$\theta \leqslant \sum_{s=2}^{2r} \binom{m}{s} \cdot \sum_{\alpha \in \{1,2,3\}^{v(s)}} \binom{k-1-2v(s)}{\ell-|M(\alpha)|} + \binom{k-1}{\ell-1}.$$

Theorem 5. If $\ell = \text{const}$ and $r \geqslant 2\ell+1$, then the ratio of unstable $\mathrm{RM}(r,m)$ subcodes tends to zero as $m \to \infty$.

Proof. Our goal is the asymptotic estimate of the probability of the event that after removing ℓ vectors from the standard basis of the code $\mathrm{RM}(r,m)$, the square of the resulting code will differ from $\mathrm{RM}(2r,m)$. The upper bound for it is $\theta \Big/ \binom{k}{\ell}$. We divide this bound into two parts and show the tendency to zero for each of them independently. For one of them it follows immediately from the fact that

$$\binom{k-1}{\ell-1} \Big/ \binom{k}{\ell} = \frac{\ell}{k} \xrightarrow[m\to\infty]{} 0,$$

since $k \to \infty$ as $m \to \infty$.

Now we consider the first part and denote it's numerator by γ. Notice that

$$\#_\alpha(1) + \#_\alpha(2) + 2 \cdot \#_\alpha(3) = |M(\alpha)| \geqslant v(s) = \#_\alpha(1) + \#_\alpha(2) + \#_\alpha(3).$$

Then the number of removed vectors that are elements of critical partitions for s is $|M(\alpha)| \geqslant v(s)$ and the total number of removed vectors is ℓ. That is, $v(s) \leqslant \ell$ and we can consider only parameters s satisfying this condition. Then

$$2v(s) = \sum_{p=\max\{s-r,1\}}^{\min\{r,s-1\}} \binom{s}{p} \leqslant 2\ell. \tag{6}$$

We consider separately two cases. If $s \geqslant r+1$, we have $\min\{r, s-1\} = r$ and in the sum (6) there is the element $\binom{s}{r}$. Thus,

$$2\ell \geqslant 2v(s) \geqslant \binom{s}{r} \geqslant s.$$

The last inequality follows from the fact that

$$\binom{s}{r} = \frac{(r+1)}{2} \cdot \frac{(r+2)}{3} \cdot \ldots \frac{(s-1)}{r} \cdot \frac{s}{1}.$$

If, on the other hand, $s < r+1$, we have $\max\{s-r, 1\} = 1$ and there is the element $\binom{s}{1}$ in the sum (6). Hence

$$2\ell \geqslant 2v(s) \geqslant \binom{s}{1} = s.$$

So either way the inequality $s \leqslant 2\ell$ is satisfied.

We simplify the upper bound for γ using this inequality and the monotonicity of the binomial coefficient $\binom{n}{k}$ with respect to the parameter k, which guarantees the increase of the value $\binom{n}{k}$ with the increase of k:

$$\sum_{s=2}^{2r} \binom{m}{s} \sum_{\alpha\in\{1,2,3\}^{v(s)}} \binom{k-1-2v(s)}{\ell-|M(\alpha)|} \leqslant \sum_{s=2}^{2\ell} \binom{m}{2\ell} \sum_{\alpha\in\{1,2,3\}^{v(s)}} \binom{k-1-2v(s)}{\ell-|M(\alpha)|} \leqslant$$
$$\leqslant 2\ell \binom{m}{2\ell} \max_{s\in[2,2\ell]} \left\{ \binom{k-1-2v(s)}{\ell-z} \cdot 3^{v(s)} \right\},$$

where $z = \min\limits_{\alpha\in\{1,2,3\}^{v(s)}} \{|M(\alpha)|\}$.

Note that $\ell = \text{const}$ and $3^{w(s)} \leqslant \text{const}$, since $s \leqslant 2\ell$, and $v(s) < 2^s$. The last is true by virtue of

$$2^s = (1+1)^s = \sum_{p=0}^{s} \binom{s}{k} > \frac{1}{2} \sum_{p=\max\{s-r,1\}}^{\min\{r,s-1\}} \binom{s}{p} = v(s).$$

These considerations, as well as the monotonicity of the binomial coefficient $\binom{n}{k}$ with respect to n and the inequality $|M(\alpha)| \geqslant v(s)$, allow us to obtain the upper bound

$$\text{const} \cdot \binom{m}{2\ell} \binom{k}{\ell - v(s)} \leqslant \text{const} \cdot \binom{m}{2\ell} \binom{k}{\ell - 1} := \psi.$$

We proceed to the ratio estimation:

$$\gamma/\binom{k}{\ell} \leqslant \psi/\binom{k}{\ell} = \text{const}\cdot\binom{m}{2\ell}\binom{k}{\ell-1}\Big/\binom{k}{\ell} = \text{const}\cdot\binom{m}{2\ell}\cdot\ell/(k-\ell+1) =$$
$$= \text{const}\cdot\binom{m}{2\ell}/(k-\ell+1) \leqslant \text{const}\cdot\frac{m^{2\ell}}{2k}.$$

After tending m to infinity we can claim that such $p = 2\ell + 1$ exists, that is, summand $\binom{m}{p} \geqslant m^p$ is an element of the sum representation of k. Then

$$\text{const}\cdot\frac{m^{2\ell}}{2k} \leqslant \text{const}\cdot\frac{m^{2\ell}}{m^{2\ell+1}} = \text{const}\cdot\frac{1}{m} \xrightarrow[m\to\infty]{} 0.$$

Theorem 5 is proven. ■

Future research

More accurate estimates on the minimal stable code sizes for general case are still required, as are better estimates of the ratio of stable subcodes. In addition, an idea for future research could be to find an analogues of the obtained results for an arbitrary basis of the Reed — Muller code.

Acknowledgments

The author thanks Ivan Chizhov for stating the problem and Lev Vysotsky for valuable help in preparing the text.

REFERENCES

1. *Shor P. V.* Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. SIAM J. Computing, 1997, vol. 26, no. 5, pp. 1484–1509.
2. *McEliece R. J.* A public-key cryptosystem based on algebraic coding theory. DSN Progress Report, 1978, vol. 4244, pp. 114–116.
3. *Niederreiter H.* Knapsack-type cryptosystems and algebraic coding theory. Problems Control Inform. Theory, 1986, vol. 15, no. 2, pp. 159–166.
4. `https://csrc.nist.gov/Projects/post-quantum-cryptography/Post-Quantum-Cryptography-Standardization/Call-for-Proposals` — NIST Call for Proposals, 2016.
5. *Wieschebrink C.* An attack on a modified Niederreiter encryption scheme. LNCS, 2006, vol. 3958, pp. 14–26.
6. *Wieschebrink C.* Cryptanalysis of the Niederreiter public key scheme based on GRS subcodes. LNCS, 2010, vol. 6061, pp. 61–72.
7. *Berger T. P. and Loidreau P.* How to mask the structure of codes. Designs, Codes, Cryptogr., 2005, vol. 35, no. 1, pp. 63–79.
8. *Couvreur A., Marquez-Corbella I., and Pellikaan R.* Cryptanalysis of public-key cryptosystems that use subcodes of algebraic geometry codes. Coding Theory Appl.ications. CIM Ser. Math. Sci., 2015, vol. 3, pp. 133–140.
9. *Lee Y., Lee W., Kim Y. S., and No J.-S.* Modified pqsigRM: RM code-based signature scheme. IEEE Access, 2020, vol. 8, pp. 177506–177518.
10. *Borodin M. A. and Chizhov I. V.* Effective attack on the McEliece cryptosystem based on Reed — Muller codes. Discr. Math. Appl., 2014, vol. 24, no. 5, pp. 273–280.
11. *Minder L. and Shokrollahi A.* Cryptanalysis of the Sidelnikov cryptosystem. LNCS, vol. 4515, pp. 347–360.

12. *Chizhov I. V. and Borodin M. A.* Hadamard products classification of subcodes of Reed — Muller codes codimension 1. Discr. Math. Appl., 2020, vol. 32, no. 1, pp. 115–134.
13. *Couvreur A., Gaborit P., Gauthier-Umãna V., et al.* Distinguisher-based attacks on public-key cryptosystems using Reed — Solomon codes. Designs, Codes, Cryptogr., 2014, vol. 73, no. 2, pp. 641–666.
14. *Erdös P. and Spencer J.* Probabilistic Methods in Combinatorics. Budapest, Akadèmiai Kiadó, 1974. 106 p.
15. *Rödl V.* On a Packing and Covering Problem. European J. Combinatorics, 1985, vol. 6, no. 1, pp. 69–78.

ПРИКЛАДНАЯ ТЕОРИЯ ГРАФОВ

УДК 510.52, 510.67, 519.17

АЛГОРИТМЫ РЕШЕНИЯ СИСТЕМ УРАВНЕНИЙ НАД РАЗЛИЧНЫМИ КЛАССАМИ КОНЕЧНЫХ ГРАФОВ[1]

А. В. Ильев*, В. П. Ильев**

*Институт математики им. С. Л. Соболева СО РАН, г. Омск, Россия,
**Омский государственный университет им. Ф. М. Достоевского, г. Омск, Россия

Целью работы является исследование и решение конечных систем уравнений над конечными неориентированными графами. Уравнениями над графами называются атомарные формулы языка L, состоящего из множества констант (вершин графа), бинарного предиката смежности вершин и предиката равенства. Доказано, что задача проверки совместности системы уравнений S от k переменных над произвольным обыкновенным n-вершинным графом Γ является $\mathcal{NP}$-полной. Подсчитана вычислительная сложность процедуры проверки совместности системы уравнений S над обыкновенным графом Γ и процедуры нахождения общего решения этой системы. Вычислительная сложность алгоритма решения системы уравнений S от k переменных над произвольным обыкновенным n-вершинным графом Γ, включающего эти процедуры, составляет $O(k^2n^{k/2+1}(k+n)^2)$ при $n \geqslant 3$. Выделены полиномиально разрешимые случаи: системы уравнений над деревьями, лесами, двудольными и полными двудольными графами, для решения которых предложены полиномиальные алгоритмы трудоёмкости $O(k^2n(k+n)^2)$.

Ключевые слова: *граф, система уравнений, вычислительная сложность.*

DOI 10.17223/20710410/53/6

ALGORITHMS FOR SOLVING SYSTEMS OF EQUATIONS OVER VARIOUS CLASSES OF FINITE GRAPHS

A. V. Il'ev*, V. P. Il'ev**

*Sobolev Institute of Mathematics SB RAS, Omsk, Russia
**Dostoevsky Omsk State University, Omsk, Russia

E-mail: artyom_iljev@mail.ru, iljev@mail.ru

The aim of the paper is to study and to solve finite systems of equations over finite undirected graphs. Equations over graphs are atomic formulas of the language L consisting of the set of constants (graph vertices), the binary vertex adjacency predicate and the equality predicate. It is proved that the problem of checking compatibility of a system of equations S with k variables over an arbitrary simple n-vertex graph Γ is $\mathcal{NP}$-complete. The computational complexity of the procedure for checking compatibility of a system of equations S over a simple graph Γ and the procedure for finding a general solution of this system is calculated. The computational complexity

[1]Работа первого автора поддержана грантом РНФ № 19-11-00209.

of the algorithm for solving a system of equations S with k variables over an arbitrary simple n-vertex graph Γ involving these procedures is $O(k^2n^{k/2+1}(k+n)^2)$ for $n \geqslant 3$. Polynomially solvable cases are distinguished: systems of equations over trees, forests, bipartite and complete bipartite graphs. Polynomial time algorithms for solving these systems with running time $O(k^2n(k+n)^2)$ are proposed.

Keywords: *graph, system of equations, computational complexity.*

Введение

Традиционным направлением исследований в алгебраической геометрии является решение систем уравнений над вещественными, комплексными, рациональными и целыми числами. Различным алгоритмическим проблемам, связанным с решением систем уравнений, посвящены многочисленные теоретические и практические исследования. Как правило, такие проблемы оказываются либо неразрешимыми, либо разрешимыми, но вычислительно трудными. К числу неразрешимых проблем относится проблема решения полиномиальных уравнений над кольцом целых чисел [1]. Среди разрешимых проблем, имеющих экспоненциальную сложность, можно выделить арифметику Пресбургера [2], проблему решения систем полиномиальных уравнений над полями комплексных и вещественных чисел [3] и проблему решения систем полиномиальных уравнений над конечными полями, которая эквивалентна $\mathcal{NP}$-полной проблеме выполнимости булевых формул [4].

В последние годы в алгебраической геометрии активно развивается направление исследований, в рамках которого изучаются системы уравнений над произвольными алгебраическими системами, где уравнениями являются атомарные формулы языка алгебраической системы [5]. И здесь также естественным образом возникают вопросы разрешимости и вычислительной сложности алгоритмических проблем. Например, доказано, что проблема совместности систем уравнений над конечными частично упорядоченными множествами является $\mathcal{NP}$-полной в том случае, когда проверяется существование решения, состоящего из попарно различных элементов частично упорядоченного множества [6], или что экзистенциальная теория класса всех конечных полей является $\mathcal{NP}$-трудной, а универсальная теория этого класса — со-$\mathcal{NP}$-трудной [7].

Многие алгоритмические проблемы связаны с проблемами разрешимости универсальных теорий различных классов алгебраических систем, изучение которых представляет значительный интерес в алгебраической геометрии. Для обыкновенных графов, являющихся важнейшими математическими объектами и используемых при решении большого числа практически важных задач оптимизации, вопросы разрешимости универсальных теорий подробно изучены в [8]. В свою очередь, эти вопросы имеют тесную связь с решением систем уравнений над конечными графами. Проверка принадлежности графа некоторому классу входит в качестве процедуры в алгоритмы, устанавливающие разрешимость универсальных теорий наследственных классов графов. Такая процедура может быть оформлена в виде решения конечной серии систем уравнений над графами.

В [9] предложены процедуры решения следующих задач:

1) проверка совместности системы уравнений S над конечным графом;
2) вычисление радикала системы S;
3) построение координатного графа системы S.

Отметим, что роль координатного графа системы уравнений S над графом аналогична роли общего решения системы линейных уравнений над полем в линейной алгебре. При этом координатный граф однозначно определяется радикалом системы S.

В настоящей работе доказано, что задача проверки совместности системы уравнений над произвольным конечным обыкновенным графом является $\mathcal{NP}$-полной. Выделены полиномиально разрешимые случаи: системы уравнений над деревьями, лесами, двудольными и полными двудольными графами. Конструктивно улучшена процедура проверки совместности системы уравнений S от k переменных над произвольным n-вершинным графом Γ и подобраны структуры данных, позволяющие реализовать все процедуры алгоритма решения системы уравнений S над графом Γ на ЭВМ. Подсчитана вычислительная сложность этого алгоритма, которая составляет $O(k^2n^{k/2+1}(k+n)^2)$ при $n \geqslant 3$. Кроме того, предложены полиномиальные процедуры проверки совместности систем уравнений и нахождения их общего решения для класса двудольных графов, в который входят также деревья и леса, и класса полных двудольных графов. Доказано, что вычислительная сложность алгоритмов решения систем уравнений над двудольными и полными двудольными графами составляет $O(k^2n(k+n)^2)$.

1. Предварительные сведения

Граф — это пара $\Gamma = (V, E)$, где V — непустое множество элементов, называемых *вершинами*; E — множество неупорядоченных пар различных элементов из V, называемых *рёбрами*. Если $(u, v) \in E$, то вершины u и v называются *смежными*. Граф *конечен*, если множество его вершин конечно, и *счётно бесконечен*, если множество его вершин счётно бесконечно. В данной работе рассматриваются только конечные обыкновенные графы, т. е. графы, не содержащие петель и кратных рёбер.

Граф называется *деревом*, если он не содержит циклов и является связным, и *лесом*, если он не содержит циклов и может быть несвязным. Граф называется *двудольным*, если множество его вершин можно разбить на два непересекающихся подмножества V_1 и V_2 так, что каждое ребро графа соединяет вершины из разных долей. Двудольный граф называется *полным двудольным*, если любая вершина из одной его доли смежна со всеми вершинами из другой доли.

Теперь дадим определение графа как алгебраической системы.

Граф — это алгебраическая система $\Gamma = \langle V, \mathrm{L}\rangle$, носитель которой V — непустое не более чем счётное множество, а язык $\mathrm{L} = \{E(x, y) \cup (x = y) \cup \mathrm{C}\}$ состоит из множества констант $\mathrm{C} \subseteq V$, бинарного предиката смежности вершин и предиката равенства, причём предикат смежности $E(x, y)$ является *иррефлексивным и симметричным*, т. е. удовлетворяет условиям:

1) $\forall x \big(\neg E(x, x)\big)$ (иррефлексивность);
2) $\forall x \forall y \big(E(x, y) \to E(y, x)\big)$ (симметричность).

Поскольку язык L обыкновенных графов не содержит функциональных символов, то здесь и далее рассматриваются только предикатные системы, а все определения адаптированы под предикатный случай.

Пусть $\Gamma = \langle V, \mathrm{L}\rangle$ — фиксированный обыкновенный граф, X — конечное множество переменных. Множество $T_{\mathrm{L}}(X)$ *термов* языка L от переменных из множества X состоит из всех переменных $x \in X$ и всех констант C. Множество $At_{\mathrm{L}}(X)$ *атомарных формул* языка L от переменных из множества X состоит из всех формул вида $t_1 = t_2$ и $E(t_1, t_2)$, где $t_1, t_2 \in T_{\mathrm{L}}(X)$. Атомарные формулы называются *уравнениями языка*, а произвольные подмножества $S \subseteq At_{\mathrm{L}}(X)$ — *системами уравнений* языка L. В данной

работе рассматриваются только конечные системы уравнений *диофантовых языков*, т. е. таких языков, в которых множество констант совпадает с множеством вершин графа.

Любая система уравнений S от k переменных над фиксированным обыкновенным графом определяет в аффинном k-мерном пространстве V^k множество своих решений $V_\Gamma(S) = \{\overline{v} \in V^k : \Gamma \models S(\overline{v})\}$, которое называется *алгебраическим множеством*. Если $V_\Gamma(S) = \varnothing$, то система уравнений S *несовместна* над Γ; иначе она является *совместной*. Две системы уравнений S_1 и S_2 называются *эквивалентными* над Γ, если $V_\Gamma(S_1) = V_\Gamma(S_2)$. Для любой системы уравнений S над Γ существует единственная эквивалентная ей максимальная система уравнений над Γ, которая называется *радикалом* системы S и обозначается $\mathrm{Rad}_\Gamma(S)$. Если система S несовместна над Γ, то $\mathrm{Rad}_\Gamma(S) = At_{\mathrm{L}}(X)$.

Роль координатного графа $CG_\Gamma(S)$ системы уравнений S аналогична роли общего решения системы линейных уравнений над полем в линейной алгебре. Координатный граф однозначно определяется радикалом $\mathrm{Rad}_\Gamma(S)$ следующим образом.

Отношение θ_S на множестве термов $T_{\mathrm{L}}(X)$, заданное по правилу

$$t_1 \sim_{\theta_S} t_2 \;\Leftrightarrow\; (t_1 = t_2) \in \mathrm{Rad}_\Gamma(S),\; t_1, t_2 \in T_{\mathrm{L}}(X),$$

является отношением эквивалентности, а константные и предикатные символы языка L интерпретируются на фактор-множестве $T_{\mathrm{L}}(X)/\theta_S$ следующим образом:

1) $c/\theta_S = c$ для любого символа $c \in \mathrm{C}$;
2) $E(t_1/\theta_S, t_2/\theta_S) = \mathtt{и} \;\Leftrightarrow\; E(t_1, t_2) \in \mathrm{Rad}_\Gamma(S)$.

Построенный на фактор-множестве $T_{\mathrm{L}}(X)/\theta_S$ граф $CG_\Gamma(S)$ называется *координатным графом* алгебраического множества $V_\Gamma(S)$. Если система S несовместна над Γ, то $CG_\Gamma(S)$ является тривиальной системой $\mathcal{E}$, т. е. графом, состоящим из единственной вершины и петли.

Следует отметить, что введённые отношение эквивалентности θ_S и определение координатного графа являются частными случаями обобщённых понятий конгруэнции и фактор-системы, предложенных В. А. Горбуновым и В. И. Тумановым [10].

2. Процедура проверки совместности системы уравнений в общем случае

Пусть $\Gamma = \big(V(\Gamma), E(\Gamma)\big)$ — обыкновенный граф с n вершинами $v_1, \ldots, v_n$, S — конечная система уравнений над Γ от переменных из множества $X = \{x_1, \ldots, x_k\}$.

Теорема 1. Задача проверки совместности системы уравнений S над произвольным конечным графом Γ является $\mathcal{NP}$-полной задачей.

Доказательство. Достаточно показать, что к задаче проверки совместности системы уравнений S над конечным графом Γ полиномиально сводится какая-нибудь другая $\mathcal{NP}$-полная задача распознавания [11], например распознавательный вариант задачи о клике.

Задача о клике. Дан n-вершинный граф $\Gamma = (V, E)$ и натуральное число $k \leqslant n$. Существует ли в графе Γ *клика* размера, не меньшего k, т. е. такое множество $V' \subseteq V$, что $(u, v) \in E$ для любых $u, v \in V'$ и $|V'| \geqslant k$?

Для ответа на поставленный вопрос нужно определить, совместна ли над графом Γ система уравнений $S = \big\{E(x_i, x_j) : i, j \in \{1, \ldots, k\},\; i < j\big\}$.

Таким образом, задача проверки совместности системы уравнений S над конечным графом Γ является $\mathcal{NP}$-полной. ■

Следствие 1. Если $\mathcal{P} \neq \mathcal{NP}$, то не существует полиномиального алгоритма проверки совместности системы уравнений S над произвольным конечным графом Γ.

2.1. Процедура проверки совместности системы уравнений S над произвольным конечным графом

Ранее А. В. Ильевым и В. Н. Ремесленниковым была предложена процедура проверки совместности системы уравнений S над конечным графом Γ [9]. Мы приведём её конструктивно улучшенный вариант без каких-либо существенных и структурных изменений. Важным предварительным шагом для работы этой процедуры является нахождение информационной базы системы уравнений S.

Информационная база системы S состоит из набора конечных множеств и натуральных чисел, определяемых по группам:

1) $X = \{x_1, \ldots, x_k\}$ — множество переменных, $S = \{s_1, \ldots, s_l\}$ — множество уравнений с переменными из X; k, l — числовые параметры.
2) $W_1, \ldots, W_k$ — подмножества $V(\Gamma)$; W_i состоит из вершин графа Γ, которые содержатся в записи уравнений вида $E(x_i, v_j)$ системы S; $\alpha_i = |W_i|$ — числовые параметры, $i = 1, \ldots, k$.
3) $W_1^{\perp}, \ldots, W_k^{\perp}$ — подмножества $V(\Gamma)$; $W_i^{\perp}$ состоит из вершин графа Γ, которые смежны с каждой из вершин множества W_i (если $W_i = \varnothing$, то по определению полагаем $W_i^{\perp} = V(\Gamma)$); $\beta_i = |W_i^{\perp}|$ — числовые параметры, $i = 1, \ldots, k$.
4) $W_1^{\perp\perp}, \ldots, W_k^{\perp\perp}$ — подмножества $V(\Gamma)$; $W_i^{\perp\perp} = (W_i^{\perp})^{\perp}$; $\gamma_i = |W_i^{\perp\perp}|$ — числовые параметры, причём $\gamma_i \geqslant \alpha_i$ для любых $i = 1, \ldots, k$.

Если в информационной базе хотя бы одно из чисел β_i равно нулю при $\alpha_i \neq 0$, то информационная база является *несогласованной*, а система S несовместна над Γ.

Поскольку случай $n = 1$ тривиальный, а при $n = 2$ обыкновенный граф Γ является двудольным и для проверки совместности системы уравнений S над ним уместно использовать полиномиальную процедуру 1.1 из п. 3.1, то всюду далее предполагается, что $n \geqslant 3$.

Процедура 1. Данная процедура строит классы эквивалентности $Y(t_i)$, где t_i — переменная x_i либо константа v_i, и преобразует систему уравнений S в эквивалентную систему $\overline{S}$ на графе Γ. В начале работы процедуры каждое множество $Y(t_i)$ состоит только из одного терма t_i, а система $\overline{S}$ совпадает с S.

В процессе работы процедура использует следующие структуры данных:

— Граф Γ представлен в виде матрицы смежности $A = \big(a_{ij}\big)$ размера $n \times n$, состоящей из нулей и единиц.
— Каждому типу уравнений отвечает своя матрица, состоящая из нулей и единиц, в которой единицами отмечены соответствующие уравнения. Равенства вида $x_i = x_j$ записаны в виде матрицы размера $k \times k$, равенства $x_i = v_j$ — в виде матрицы размера $k \times n$, а равенства $v_i = v_j$ — в виде матрицы размера $n \times n$. Матрицы, соответствующие уравнениям вида $E(t_i, t_j)$, выглядят аналогично.
— Семейства множеств $\{W_i\}$ и $\{W_i^{\perp}\}$, также записаны в виде матриц W и $W^{\perp}$, состоящих из нулей и единиц, размера $k \times n$. Строками этих матриц являются характеристические векторы множеств W_i и $W_i^{\perp}$.
— Классы эквивалентности $Y(t_i)$ хранятся в виде списков термов. Изначально количество таких списков $k + n$, а терм t_i — первый и единственный элемент в каждом списке. В дальнейшем при работе процедуры количество списков может сокращаться, при этом длина каждого из них ограничена значением $k + n$.

Шаг 0. Производится заполнение матриц W и $W^{\perp}$ и построение классов эквивалентности $Y(t_i)$. Матрица W сначала совпадает с матрицей, соответствующей уравнениям вида $E(x_i, v_j)$. Чтобы заполнить матрицу $W^{\perp}$, нужно для каждого её элемента перемножить элементы матрицы A графа Γ из тех строк, которые соответствуют вершинам из множеств W_i, т. е. в матрице $W^{\perp}$ каждый элемент определяется по формуле $w_{ij}^{\perp} = \prod\limits_{m:\ v_m \in W_i} a_{mj}$. Если хотя бы одна из строк матрицы $W^{\perp}$ состоит только из нулей, то исходная система S несовместна над Γ и процедура завершает свою работу.

Шаг 1. Для каждого равенства $t_i = t_j$ из $\overline{S}$ процедура объединяет множества $Y(t_p)$ и $Y(t_q)$, содержащие термы t_i и t_j. Полученное множество обозначается $Y(t_m)$, где t_m — константа с наименьшим номером, а при её отсутствии — переменная с наименьшим номером из множества $Y(t_p) \cup Y(t_q)$. Равенство $t_i = t_j$ при этом удаляется из $\overline{S}$.

Шаг 2. Процедура последовательно просматривает все множества $Y(t)$.

1) Если существует множество $Y(t)$, в котором содержатся две различные константы v_i и v_j, то исходная система S несовместна над Γ.
2) Если в множестве $Y(t)$ содержится только одна константа v_j и при этом $\bigcap\limits_{i:\ x_i \in Y(t)} W_i^{\perp} \cap \{v_j\} = \varnothing$, то исходная система S несовместна над Γ.
3) Если множество $Y(t)$ не содержит констант v_j, но при этом $\bigcap\limits_{i:\ x_i \in Y(t)} W_i^{\perp} = \varnothing$, то исходная система S несовместна над Γ.

   В каждом из случаев 1–3 процедура завершает работу.
4) Если в множестве $Y(t)$ содержится только одна константа v_j и при этом $\bigcap\limits_{i:\ x_i \in Y(t)} W_i^{\perp} \cap \{v_j\} = \{v_j\}$, то во всех уравнениях $\overline{S}$ каждая переменная $x_i \in Y(t)$ заменяется на v_j и для каждой из них множество $W_i^{\perp}$ переопределяется следующим образом: $W_i^{\perp} := \{v_j\}$.
5) Если множество $Y(t)$ не содержит констант, но при этом $\bigcap\limits_{i:\ x_i \in Y(t)} W_i^{\perp} = \{v_j\}$, то во всех уравнениях $\overline{S}$ каждая переменная $x_i \in Y(t)$ заменяется на v_j и в $\overline{S}$ добавляются равенства $x_i = v_j$.
6) Если множество $Y(t)$ не содержит констант и при этом $\left| \bigcap\limits_{i:\ x_i \in Y(t)} W_i^{\perp} \right| > 1$, то алгоритм выбирает переменную $x_j \in Y(t)$ с наименьшим номером и во всех уравнениях $\overline{S}$ заменяет все остальные переменные из $Y(t)$ на x_j.

Шаг 3. Процедура просматривает уравнения $E(t_i, t_j)$.

1) Если в $\overline{S}$ имеются уравнения $E(x_i, x_i)$ либо $E(v_i, v_j)$, такие, что $(v_i, v_j) \notin E(\Gamma)$, то исходная система S несовместна над Γ и процедура заканчивает работу. В противном случае уравнения $E(v_i, v_j)$ удаляются из системы $\overline{S}$.
2) Для каждой переменной, входящей в уравнения вида $E(x_i, v_j)$, процедура ищет элемент v_j в множестве W_i. Если такой элемент не найден, т. е. уравнение $E(x_i, v_j)$ было получено на шаге 2, то процедура переопределяет множества W_i и $W_i^{\perp}$ следующим образом: $W_i := W_i \cup \{v_j\}$ и $W_i^{\perp} := W_i^{\perp} \cap \{v_j\}^{\perp}$, где множество $\{v_j\}^{\perp}$ состоит из всех вершин графа Γ, смежных с v_j. Если получили $W_i^{\perp} = \varnothing$, то исходная система S несовместна над Γ; если $W_i^{\perp} = \{v_m\}$, то в $\overline{S}$ добавляется равенство $x_i = v_m$.

Если после выполнения шага 3 в $\overline{S}$ осталось хотя бы одно равенство, то процедура возвращается на шаг 1; если в $\overline{S}$ равенств нет, но есть уравнения вида $E(x_i, x_j)$, то процедура переходит на шаг 4. В противном случае процедура завершает работу.

Шаг 4. Процедура просматривает уравнения $E(x_i, x_j)$.

1) Процедура ищет минимальное доминирующее множество в графе $\mathrm{H} = (V_X, E_X)$, где V_X — множество переменных, входящих в запись уравнений $E(x_i, x_j)$, а множество рёбер E_X определяется этими уравнениями. Напомним, что подмножество $V' \subseteq V$ вершин графа (V, E) называется *доминирующим*, если каждая вершина из $V \setminus V'$ смежна по крайней мере с одной вершиной из V'. Для поиска минимального доминирующего множества можно использовать любые известные алгоритмы, имеющие трудоёмкость не хуже, чем $O(3^{|V_X|/2})$, например, предложенные в [12, 13]. Найденное доминирующее множество обозначим X'. Без ограничения общности можно считать, что $X' = \{x_1, \ldots, x_p\}$. Несложно заметить, что $p \leqslant k/2$.
2) Далее рассматриваются всевозможные наборы вида $(v_{1q}, v_{2q}, \ldots, v_{pq})$, где $v_{jq} \in W_j^{\perp}$. Всего таких наборов $|W_1^{\perp}| \cdot |W_2^{\perp}| \cdot \ldots \cdot |W_p^{\perp}| \leqslant n^p$. Для каждого такого набора составляется система уравнений

$$S_q = \overline{S} \cup \Big(\bigcup_{j=1}^{p} (x_j = v_{jq}) \Big)$$

с собственной информационной базой, включающей свои множества W_{iq} и $W_{iq}^{\perp}$. Каждая такая система проверяется на совместность, т. е. для неё выполняются шаги 1–3 процедуры. Если все системы S_q окажутся несовместными над Γ, то система $\overline{S}$ тоже несовместна над Γ и процедура завершает работу.
3) В случае, когда некоторые системы уравнений S_m из множества $\{S_q\}$ окажутся совместными над графом Γ, система уравнений $\overline{S} \cup \big(\bigcap\limits_m \overline{S}_m\big)$ эквивалентна исходной системе S, где $\overline{S}_m$ — системы уравнений, полученные после выполнения шагов 1–3 процедуры для совместных систем S_m. В этом случае происходит доопределение системы $\overline{S}$ вместе с множествами W_i и $W_i^{\perp}$ следующим образом: $\overline{S} := \overline{S} \cup \big(\bigcap\limits_m \overline{S}_m\big)$, $W_i := W_i \cup \big(\bigcap\limits_m W_{im}\big)$ и $W_i^{\perp} := W_i^{\perp} \cup \big(\bigcap\limits_m W_{im}^{\perp}\big)$. Заметим, что системы $\overline{S}_m$ могут состоять только из уравнений вида $E(x_i, v_j)$.
4) Кроме того, если некоторые классы эквивалентности $Y(t)$ объединялись в каждой совместной системе $\overline{S}_m$, то эти классы эквивалентности объединяются и в системе $\overline{S}$. Для этого составляются специальные матрицы $Y' = \big(y'_{ij}\big)$ размера $k \times k$ и $Y'' = \big(y''_{ij}\big)$ размера $k \times n$, изначально заполненные единицами. Затем для каждой системы $\overline{S}_m$ данные матрицы преобразуются следующим образом. Элемент y'_{ij} сохраняет свое текущее значение, если равенство $x_i = x_j$ имеет место в системе $\overline{S}_m$, и $y'_{ij} := 0$ в противном случае. Аналогично элемент y''_{ij} сохраняет свое текущее значение, если равенство $x_i = v_j$ имеет место в системе $\overline{S}_m$, и $y''_{ij} := 0$ в противном случае. В итоге матрицы Y' и Y'' будут содержать информацию обо всех равенствах вида $x_i = x_j$ и $x_i = v_j$, имевших место в любой совместной системе $\overline{S}_m$. Далее для каждого равенства, определяемого этими матрицами, происходит объединение соответствующих классов эквивалентности системы $\overline{S}$.

После прохождения шагов 0–4 процедура завершает работу. В результате система $\overline{S}$ состоит только из уравнений вида $E(t_i, t_j)$.

2.2. Корректность процедуры проверки совместности системы уравнений S в общем случае

Доказательство следующей теоремы приведено в [9]. Для полноты картины воспроизведём его здесь.

Теорема 2. Процедура 1 корректно проверяет систему уравнений S на совместность.

Доказательство. Прежде всего докажем, что система уравнений $\overline{S}$, построенная процедурой 1, эквивалентна S.

По окончании шага 1 это очевидно.

Этапы 4 и 5 шага 2 означают, что только v_j является вершиной графа Γ, которая может быть значением переменных из класса $Y(t)$. Поэтому замена каждой из этих переменных на v_j сохраняет эквивалентность системы уравнений $\overline{S}$ исходной системе над Γ.

Этап 6 шага 2 означает, что множество значений переменных из класса $Y(t)$ непусто, и потому замена каждой из этих переменных на x_j сохраняет эквивалентность системы уравнений $\overline{S}$ исходной системе S над Γ.

На этапе 2 шага 3 уточняется множество значений переменной x_i на графе Γ, и если оно состоит только из одной вершины v_j, то добавление в $\overline{S}$ уравнения $x_i = v_j$ сохраняет эквивалентность системы уравнений $\overline{S}$ исходной системе S над Γ.

На шаге 4 осуществляется переход от системы уравнений $\overline{S}$, содержащей уравнения с двумя переменными $E(x_i, x_j)$, к системам уравнений S_q, таким, что $\overline{S}$ эквивалентна объединению $\bigcup\limits_q S_q$ над графом Γ. Если в каждую совместную систему S_m из множества $\{S_q\}$ после её преобразования в систему $\overline{S}_m$ попадут какие-либо уравнения, не содержащиеся в $\overline{S}$, то добавление этих уравнений в $\overline{S}$ сохранит эквивалентность системы уравнений $\overline{S}$ исходной системе S над графом Γ. Если в процессе проверки совместности каждой системы S_m происходило объединение одних и тех же классов эквивалентности $Y(t)$, то эти классы эквивалентности должны быть объединены и в системе $\overline{S}$.

Очевидно, что для любой переменной x_i множество её значений содержится в $W_i^{\perp}$. Несовместность системы $\overline{S}$, устанавливаемая на этапе 1 шага 2, следует из попадания двух констант v_i и v_j в один класс эквивалентности, что противоречит условию попарного различия всех вершин графа. Несовместность системы $\overline{S}$, устанавливаемая на этапах 2 и 3 шага 2, следует из того, что никакая вершина графа Γ не может быть значением переменной x_i из класса эквивалентности $Y(t)$.

Несовместность системы $\overline{S}$, устанавливаемая на этапе 1 шага 3, следует из отсутствия в графе Γ петель и возможного отсутствия рёбер (v_i, v_j), существование которых вытекает из уравнений системы $\overline{S}$. Несовместность системы $\overline{S}$, устанавливаемая на этапе 2 шага 3, следует из того, что переменной x_i не могут быть присвоены никакие значения на графе Γ.

Несовместность системы $\overline{S}$, устанавливаемая на шаге 4 процедуры, следует из того, что переменным из доминирующего множества X' не могут быть всем одновременно присвоены никакие конкретные значения на графе Γ. ■

2.3. Трудоёмкость процедуры проверки совместности системы уравнений S в общем случае

Теорема 3. Трудоёмкость процедуры 1 составляет $O(k^2 n^{k/2+1}(k+n)^2)$.

Доказательство. Распишем вычислительную сложность процедуры 1 по шагам:

Сложность шага 0 составляет $O(kn + kn^2)$, или, что то же, $O(kn^2)$.

Сложность шага 1 — $O(k^2 + n^2 + kn)$, или, что то же, $O((k + n)^2)$.

Сложность этапа 1 шага 2 в худшем случае составляет $O((k + n)^2)$.

Сложность этапов 2 и 3 шага 2 — $O((k+n)(k+n+kn))$, или, что то же, $O(kn(k+n))$.

Сложность этапов 4–6 шага 2 — $O((k+n)(k+n+kn+k^2)$, или, что то же, $O(k(k+n)^2)$.

Таким образом, трудоёмкость шага 2 составляет $O(k(k + n)^2)$.

Сложность этапа 1 шага 3 составляет $O(k + n^2)$.

Сложность этапа 2 шага 3 — $O(kn + kn^2)$, или, что то же, $O(kn^2)$.

Таким образом, трудоёмкость всех шагов 1–3 процедуры за одну итерацию составляет $O(k(k + n)^2)$.

Поскольку шаги 1–3 могут быть повторены для каждого из kn возможных равенств $x_i = v_j$, то общая вычислительная сложность их прохождения в худшем случае составляет $O(k^2n(k + n)^2)$.

Сложность этапа 1 шага 4 в худшем случае составляет $O(3^{k/2})$, что не хуже, чем $O(n^{k/2})$ при $n \geqslant 3$.

Поскольку на этапе 2 шага 4 количество всевозможных комбинаций $(v_{1q}, v_{2q}, \ldots, v_{pq})$ при $p \leqslant k/2$ и $q \leqslant n$ в худшем случае равно $n^{k/2}$ и для каждой полученной новой системы уравнений S_q осуществляется прохождение шагов 1–3, то сложность этапа 2 шага 4 составляет $O(k^2n^{k/2+1}(k + n)^2)$.

Сложность этапа 3 шага 4 в худшем случае — $O(kn^{k/2+1})$.

Сложность этапа 4 шага 4 в худшем случае — $O(k^2 + kn + n^{k/2}(k + n)^2)$, или, что то же, $O(n^{k/2}(k + n)^2)$.

Таким образом, трудоёмкость всей процедуры 1 составляет $O(k^2n^{k/2+1}(k + n)^2)$. ■

3. Процедура проверки совместности системы уравнений над двудольными графами

Поскольку задача проверки совместности системы уравнений в общем случае $\mathcal{NP}$-полна, т. е. не является полиномиально разрешимой при $\mathcal{P} \neq \mathcal{NP}$, то возникает естественный вопрос о выделении её полиномиально разрешимых случаев. В число таких случаев входят задачи проверки совместности системы уравнений над деревьями, лесами, всеми двудольными графами и полными двудольными графами.

Нетрудно заметить, что деревья и леса являются двудольными графами, но не обязательно полными двудольными. Поэтому предложена единая процедура проверки совместности системы уравнений S над конечными двудольными графами, которая применима к деревьям и лесам, и отдельная процедура проверки совместности системы уравнений S над полными двудольными графами.

Пусть $\Gamma = \big(V(\Gamma), E(\Gamma)\big)$ — обыкновенный двудольный граф с n вершинами и S — конечная система уравнений над Γ от переменных из множества $X = \{x_1, \ldots, x_k\}$.

3.1. Процедура проверки совместности системы уравнений S над конечным двудольным графом

Процедура 1.1. Идея этой процедуры аналогична идее процедуры 1. В процессе работы используются те же структуры данных, что и в общем случае.

Шаги 0–3 в точности повторяют соответствующие шаги процедуры 1, а шаг 4 выглядит следующим образом:

Шаг 4. Процедура просматривает граф $\mathrm{H} = (V_X, E_X)$, где V_X — множество переменных, входящих в запись уравнений $E(x_i, x_j)$, а множество рёбер E_X определяется этими уравнениями.

1) В графе H ищутся циклы нечётной длины. Для этого производится расслоение каждой его компоненты связности, начиная с произвольной вершины, — создаётся специальный список, в котором каждый элемент содержит номер своего слоя и номер своей вершины. Если цикл нечётной длины найден, т. е. какие-то две вершины из одного слоя оказались смежны, то исходная система уравнений S несовместна над графом Γ.
2) Для каждого уравнения вида $E(x_i, x_j)$ множества $W_i^{\perp}$ и $W_j^{\perp}$ переопределяются следующим образом: $W_i^{\perp} := W_i^{\perp} \cap \Big(\bigcup\limits_{v_j \in W_j^{\perp}} \{v_j\}^{\perp} \Big)$ и $W_j^{\perp} := W_j^{\perp} \cap \Big(\bigcup\limits_{v_i \in W_i^{\perp}} \{v_i\}^{\perp} \Big)$.
   Если хотя бы одно множество $W_p^{\perp} = \varnothing$, то исходная система S несовместна над Γ; иначе — совместна. Если какое-либо множество $W_p^{\perp} = \{v_q\}$, то в $\overline{S}$ добавляется равенство $x_p = v_q$.
3) Для каждого равенства $x_p = v_q$ из $\overline{S}$, полученного на предыдущем этапе, процедура объединяет множества $Y(t_r)$ и $Y(v_q)$, содержащие переменную x_p и константу v_q, а равенства удаляются из системы. Полученное множество обозначается $Y(v_q)$. На этом процедура завершает свою работу.

Теорема 4. Процедура 1.1 корректно проверяет систему уравнений S на совместность.

Доказательство. Корректность шагов 1–3 процедуры доказывается аналогично тому, как это сделано в теореме 2.

На шаге 4 возможно уточнение множества значений переменной x_p на графе Γ, и если оно состоит только из одной вершины v_q, то добавление в $\overline{S}$ уравнения $x_p = v_q$ сохраняет эквивалентность системы уравнений $\overline{S}$ исходной системе S над Γ.

Несовместность системы $\overline{S}$, устанавливаемая на шаге 4, следует из того, что в графе Γ не существует циклов нечётной длины и при этом переменные, входящие в уравнения $E(x_i, x_j)$, могут иметь своими значениями только смежные вершины в графе Γ, т. е. находящиеся в разных его долях. ■

Теорема 5. Трудоёмкость процедуры 1.1 составляет $O(k^2 n(k+n)^2)$.

Доказательство. Распишем вычислительную сложность по шагам.

Сложность каждого из шагов 0–3 процедуры 1.1 такая же, как и у процедуры 1; общая вычислительная сложность их прохождения в худшем случае составляет $O(k^2 n(k+n)^2)$.

Сложность этапа 1 шага 4 составляет $O(k^3)$; этапа 2 — $O(k^2 n^2)$; этапа 3 — $O(kn)$.

Таким образом, трудоёмкость всей процедуры 1.1 составляет $O(k^2 n(k+n)^2)$. ■

3.2. Процедура проверки совместности системы уравнений S над конечным полным двудольным графом

Процедура 1.2. Идея этой процедуры полностью аналогична идее процедуры 1.1. Полный двудольный граф Γ хранится в виде матрицы размера $2 \times n$, состоящей из нулей и единиц, а в остальном в процессе работы используются те же структуры данных, что и в общем случае.

Шаги 0–3 в точности повторяют соответствующие шаги процедуры 1, а шаг 4 выглядит следующим образом:

Шаг 4. Процедура просматривает граф $H = (V_X, E_X)$, где V_X — множество переменных, входящих в запись уравнений $E(x_i, x_j)$, а множество рёбер E_X определяется этими уравнениями.

1) В графе H ищутся циклы нечётной длины. Для этого производится расслоение каждой его компоненты связности, начиная с произвольной вершины. Если цикл нечётной длины найден, то исходная система уравнений S несовместна над графом Γ. Иначе граф H является двудольным и для каждой его компоненты связности формируются списки вершин долей.
2) Процедура просматривает списки долей графа H. Если две переменные x_i и x_j лежат в одной доле графа H, а вершины из множеств $W_i^{\perp}$ и $W_j^{\perp}$ принадлежат разным долям графа Γ, то система уравнений S несовместна над Γ. Если две переменные x_i и x_j лежат в разных долях графа H, но в одной его компоненте связности, а вершины из множеств $W_i^{\perp}$ и $W_j^{\perp}$ принадлежат одной доле графа Γ, то система S также несовместна над Γ. В противном случае система уравнений S является совместной над графом Γ.

На этом процедура завершает свою работу.

Теорема 6. Процедура 1.2 корректно проверяет систему уравнений S на совместность.

Доказательство полностью аналогично доказательству теоремы 4.

Теорема 7. Трудоёмкость процедуры 1.2 составляет $O(k^2n(k+n)^2)$.

Доказательство. Как и у процедуры 1.1, общая трудоёмкость прохождения шагов 0–3 в худшем случае составляет $O(k^2n(k+n)^2)$.

Сложность этапа 1 шага 4 составляет $O(k^3)$; этапа 2 — $O(k^2n)$.

Таким образом, трудоёмкость всей процедуры 1.2 составляет $O(k^2n(k+n)^2)$. ■

4. Процедуры построения радикала и координатного графа

Обе эти процедуры, приведенные в [9] с полным доказательством их корректности, без изменений используются как при решении систем уравнений над двудольными графами, так и в общем случае.

4.1. Построение радикала системы уравнений S

Процедура 2. Если система S несовместна над Γ, то $\mathrm{Rad}_\Gamma(S) = At_{\mathrm{L}}(X)$. В противном случае процедура работает с информационной базой системы уравнений S.

Шаг 0. Процедура рассматривает последние версии множеств $W_1^{\perp}, \ldots, W_k^{\perp}$, полученные при выполнении процедуры проверки совместности системы S. С их помощью заново определяются множества $W_1^{\perp\perp}, \ldots, W_k^{\perp\perp}$.

Далее радикал $\mathrm{Rad}_\Gamma(S)$ строится следующим образом.

Шаг 1. $\mathrm{Rad}_\Gamma(S) := \overline{S}$, для чего определяются новые матрицы уравнений.

Шаг 2. Уравнения вида $E(x_i, v_j)$ добавляются в $\mathrm{Rad}_\Gamma(S)$ для любых $v_j \in W_i^{\perp\perp}$.

Шаг 3. Если $t_1, t_2 \in Y(t)$, то уравнения $t_1 = t_2$ добавляются в $\mathrm{Rad}_\Gamma(S)$ для любых $t_1, t_2 \in T_{\mathrm{L}}(X)$.

Шаг 4. Уравнения $t = t$ добавляются в $\mathrm{Rad}_\Gamma(S)$ для любого $t \in T_{\mathrm{L}}(X)$.

Шаг 5. Если $(t_1 = t_2) \in \mathrm{Rad}_\Gamma(S)$, то уравнения $t_2 = t_1$ добавляются в $\mathrm{Rad}_\Gamma(S)$ для любых $t_1, t_2 \in T_{\mathrm{L}}(X)$. Поскольку матрица, содержащая уравнения вида $x_i = v_j$, не является квадратной, то для данного типа уравнений считаем действия на шаге 5 выполненными автоматически.

Шаг 6. Если $(t_1 = t_2) \in \mathrm{Rad}_\Gamma(S)$ и $(t_2 = t_3) \in \mathrm{Rad}_\Gamma(S)$, то уравнения $t_1 = t_3$ добавляются в $\mathrm{Rad}_\Gamma(S)$ для любых $t_1, t_2, t_3 \in T_{\mathrm{L}}(X)$.

Шаг 7. Если $E(t_1, t_2) \in \mathrm{Rad}_\Gamma(S)$, то уравнения $E(t_2, t_1)$ добавляются в $\mathrm{Rad}_\Gamma(S)$ для любых $t_1, t_2 \in T_{\mathrm{L}}(X)$. Поскольку матрица, содержащая уравнения вида $E(x_i, v_j)$,

не является квадратной, то для данного типа уравнений считаем действия на шаге 7 выполненными автоматически.

Шаг 8. Если $(t_1 = t'_1), (t_2 = t'_2) \in \mathrm{Rad}_\Gamma(S)$ и $E(t_1, t_2) \in \mathrm{Rad}_\Gamma(S)$, то уравнения $E(t'_1, t'_2)$ добавляются в $\mathrm{Rad}_\Gamma(S)$ для любых $t_1, t_2, t'_1, t'_2 \in T_{\mathrm{L}}(X)$.

Шаг 9. Уравнения $E(v_i, v_j)$ добавляются в $\mathrm{Rad}_\Gamma(S)$ для любых $(v_i, v_j) \in E(\Gamma)$.

Теорема 8. Процедура 2 корректно строит радикал $\mathrm{Rad}_\Gamma(S)$.

Корректность процедуры 2 для построения радикала $\mathrm{Rad}_\Gamma(S)$ следует непосредственно из его определения.

Теорема 9. Трудоёмкость процедуры 2 составляет $O((k+n)^3)$.

Доказательство. Распишем вычислительную сложность процедуры 2 по шагам.

Переопределение множеств $W_1^{\perp\perp}, \ldots, W_k^{\perp\perp}$ на шаге 0 осуществляется за $O(kn^2)$ операций.

Сложность шага 1 составляет $O((k+n)^2)$; шага 2 — $O(kn)$; шага 3 — $O((k+n)^3)$; шага 4 — $O(k+n)$; шага 5 — $O(k^2+n^2)$; шага 6 — $O((k+n)^3)$; шага 7 — $O(k^2+n^2)$; шага 8 — $O((k+n)^2)$; шага 9 — $O(n^2)$.

Таким образом, трудоёмкость всей процедуры 2 составляет $O((k+n)^3)$. ■

4.2. Построение координатного графа системы уравнений S

Процедура 3. Множество вершин координатного графа $\Delta = CG_\Gamma(S)$ совпадает с множеством индикаторов t_i классов эквивалентности $Y(t_i)$ и является подмножеством $V(\Gamma) \cup X$, а множество рёбер выглядит следующим образом:

$$E(\Delta) = E(\Gamma) \cup E(x_i, x_j) \cup E(x_i, w_m), \text{ где } E(x_i, x_j), E(x_i, w_m) \in \mathrm{Rad}_\Gamma(S).$$

Если система S несовместна над Γ, то $\Delta = \mathcal{E}$, где $\mathcal{E}$ — граф, состоящий из одной вершины и петли. В противном случае процедура выполняет следующие построения:

Шаг 1. К множеству вершин графа Γ добавляются все вершины из множества X.

Шаг 2. К полученному графу добавляются всевозможные рёбра (x_i, x_j) и (x_i, w_m), для которых $E(x_i, x_j), E(x_i, w_m) \in \mathrm{Rad}_\Gamma(S)$.

Шаг 3. В полученном графе для каждого класса эквивалентности $Y(t_i)$ все вершины, находящиеся в этом классе, стягиваются в одну вершину, а кратные рёбра заменяются одним ребром.

В результате получается координатный граф Δ.

Теорема 10. Процедура 3 корректно строит координатный граф Δ.

Построение координатного графа процедурой 3 осуществляется непосредственно по определению.

Теорема 11. Трудоёмкость процедуры 3 составляет $O((k+n)^2)$.

Как следствия теорем 3, 5, 7, 9 и 11, можно получить трудоёмкости алгоритмов решения конечных систем уравнений над конечными обыкновенными графами, включающие процедуры проверки совместности этих систем, построения их радикала и общего решения — координатного графа.

Следствие 2. Трудоёмкость алгоритма решения систем уравнений над произвольными графами составляет $O(k^2 n^{k/2+1}(k+n)^2)$.

Следствие 3. Трудоёмкость алгоритма решения систем уравнений над двудольными графами, в том числе деревьями и лесами, составляет $O(k^2 n(k+n)^2)$.

Следствие 4. Трудоёмкость алгоритма решения систем уравнений над полными двудольными графами составляет $O(k^2 n(k+n)^2)$.

ЛИТЕРАТУРА

1. *Матиясевич Ю. В.* Диофантовость перечислимых множеств // Доклады Академии наук СССР. 1970. Т. 191. № 2. С. 279–282.
2. *Fischer M. J. and Rabin M. O.* Super-exponential complexity of Presburger arithmetic // Proc. SIAM-AMS Symp. Appl. Math. 1974. V. 7. P. 27–41.
3. *Mayr E. W. and Meyer A. R.* The complexity of the word problems for commutative semigroups and polynomial ideals // Adv. Math. 1982. V. 46. Iss. 3. P. 305–329.
4. *Cook S. A.* The complexity of theorem proving procedures // Proc. 3rd Ann. ACM Symp. on Theory of Computing. 1971. P. 151–158.
5. *Даниярова Э. Ю., Мясников А. Г., Ремесленников В. Н.* Алгебраическая геометрия над алгебраическими системами. Новосибирск: Изд-во СО РАН, 2016. 243 с.
6. *Никитин А. Ю., Рыбалов А. Н.* О сложности проблемы разрешимости систем уравнений над конечными частичными порядками // Прикладная дискретная математика. 2018. № 39. С. 94–98.
7. *Рыбалов А. Н.* О сложности экзистенциальной и универсальной теорий конечных полей // Прикладная дискретная математика. 2019. № 45. С. 85–89.
8. *Ильев А. В.* Разрешимость универсальных теорий и аксиоматизируемость наследственных классов графов // Труды института математики и механики УрО РАН. 2016. Т. 22. № 1. С. 100–111.
9. *Ильев А. В., Ремесленников В. Н.* Исследование совместности систем уравнений над графами и нахождение их общих решений // Вестник Омского университета. 2017. № 4(86). С. 26–32.
10. *Горбунов В. А.* Алгебраическая теория квазимногообразий. Новосибирск: Научная книга, 1999. 368+xii с.
11. *Гэри М., Джонсон Д.* Вычислительные машины и труднорешаемые задачи. М.: Мир, 1982. 416 с.
12. *Fomin F. V., Grandoni F., and Kratsch D.* A measure & conquer approach for the analysis of exact algorithms // J. ACM. 2009. V. 56. Iss. 5. P. 25–32.
13. *Van Rooij J. M. M., Nederlof J., and van Dijk T. C.* Inclusion/exclusion meets measure and conquer: Exact algorithms for counting dominating sets // LNCS. 2009. V. 5757. P. 554–565.

REFERENCES

1. *Matiyasevich Y. V.* Diofantovost perechislimykh mnozhestv [Diophantineity of enumerable sets]. Doklady Akademii Nauk USSR, 1970, vol. 191, no. 2, pp. 279–282. (in Russian)
2. *Fischer M. J. and Rabin M. O.* Super-exponential complexity of Presburger arithmetic. Proc. SIAM-AMS Symp. Appl. Math., 1974, vol. 7, pp. 27–41.
3. *Mayr E. W. and Meyer A. R.* The complexity of the word problems for commutative semigroups and polynomial ideals. Adv. Math., 1982, vol. 46, iss. 3, pp. 305–329.
4. *Cook S. A.* The complexity of theorem proving procedures. Proc. 3rd Ann. ACM Symp. on Theory of Computing, 1971, pp. 151–158.
5. *Daniyarova E. Y., Myasnikov A. G., and Remeslennikov V. N.* Algebraicheskaya geometriya nad algebraicheskimi sistemami [Algebraic Geometry over Algebraic Structures]. Novosibirsk, SB RAS Publishing House, 2016. 243 p. (in Russian)
6. *Nikitin A. Y. and Rybalov A. N.* O slozhnosti problemy razreshimosti sistem uravneniy nad konechnymi chastichnymi poryadkami [On complexity of the satisfiability problem of systems over finite posets]. Prikladnaya Diskretnaya Matematika, 2018, no. 39, pp. 94–98. (in Russian)

7. *Rybalov A. N.* O slozhnosti ekzistentsialnoy i universalnoy teoriy konechnykh poley [On complexity of the existential and universal theories of finite fields]. Prikladnaya Diskretnaya Matematika, 2019, no. 45, pp. 85–89. (in Russian)
8. *Ilev A. V.* Razreshimost universalnykh teory i aksiomatiziruyemost nasledstvennykh klassov grafov [Decidability of universal theories and axiomatizability of hereditary classes of graphs]. Trudy Instituta Matematiki i Mekhaniki UrO RAN, 2016, vol. 22, no. 1, pp. 100–111. (in Russian)
9. *Ilev A. V. and Remeslennikov V. N.* Issledovaniye sovmestnosti sistem uravneny nad grafami i nakhozhdeniye ikh obshchikh resheny [Study of the compatibility of systems of equations over graphs and finding their general solutions]. Vestnik Omskogo Universiteta, 2017, no. 4(86), pp. 26–32. (in Russian)
10. *Gorbunov V. A.* Algebraic Theory of Quasivarieties. New York, Plenum, 1998. 298+xii p.
11. *Garey M. R. and Johnson D. S.* Computers and Intractability: A Guide to the Theory of NP-Completeness. San Francisco, W. H. Freeman and Co, 1979. 338+x p.
12. *Fomin F. V., Grandoni F., and Kratsch D.* A measure & conquer approach for the analysis of exact algorithms. J. ACM, 2009, vol. 56, iss. 5, pp. 25–32.
13. *Van Rooij J. M. M., Nederlof J., and van Dijk T. C.* Inclusion/exclusion meets measure and conquer: Exact algorithms for counting dominating sets. LNCS, 2009, vol. 5757, pp. 554–565.

МАТЕМАТИЧЕСКИЕ ОСНОВЫ ИНФОРМАТИКИ И ПРОГРАММИРОВАНИЯ

УДК 519.714

О СЛОЖНОСТИ РЕАЛИЗАЦИИ СИСТЕМЫ МОНОМОВ ОТ ДВУХ ПЕРЕМЕННЫХ СХЕМАМИ КОМПОЗИЦИИ

С. А. Корнеев

Московский государственный университет имени М. В. Ломоносова, г. Москва, Россия
Институт прикладной математики имени М. В. Келдыша Российской академии наук, г. Москва, Россия

Исследуется сложность реализации систем мономов схемами композиции. Для этой вычислительной модели установлена сложность реализации системы из p мономов от двух переменных с точностью до слагаемого порядка p. Показано, что для схем композиции, в отличие от других моделей, асимптотика роста сложности реализации системы из ограниченного числа мономов от двух переменных, вообще говоря, не определяется сложностью никакого несобственного подмножества мономов.

Ключевые слова: *система мономов, сложность вычисления, схемная сложность, схема композиции.*

DOI 10.17223/20710410/53/7

THE COMPLEXITY OF IMPLEMENTATION OF A SYSTEM OF MONOMIALS IN TWO VARIABLES BY COMPOSITION CIRCUITS

S. A. Korneev

Lomonosov Moscow State University, Moscow, Russia
Keldysh Institute of Applied Mathematics, Russian Academy of Sciences, Moscow, Russia

E-mail: korneev.sa.42@gmail.com

The computational complexity of systems of monomials by compositional circuits is investigated. In such a model, complexity is understood as the smallest number of composition operations required to compute a system of monomials. For this model we have found the computational complexity of a system of p monomials in two variables up to a term that grows as p. By $l_{sh}(A)$ we mean the complexity of implementation of the system of monomials defined by the matrix A by composition circuits. For any integer a we assume $a^+ = \max(a, 1)$.

Theorem. Let $A = (a_{ij})$ be a $(p \times q)$-matrix without zero rows and columns consisting of non-negative integers. Then $G(A) \leqslant l_{sh}(A) \leqslant G(A) + 2p - 3$, where

$$G(A) = \max_{(i_1,\ldots,i_p)\in S_p} \sum_{k=1}^{p} \left\lceil \log \max \left(\frac{a^+_{i_k 1}}{\max\limits_{l:l<k} a^+_{i_l 1}}, \frac{a^+_{i_k 2}}{\max\limits_{l:l<k} a^+_{i_l 2}}, 1 \right) \right\rceil.$$

We also show that for composition circuits, in contrast to other models, the asymtotical growth of the computational complexity of system of monomials in two variables in the general case is not determined by any improper subset of this system of monomials.

Keywords: *set of monomials, computation complexity, circuit complexity, composition circuit.*

1. Введение, основные определения

Исследуется сложность вычисления системы мономов схемами композиции [1–4].

Операция композиции, предложенная в [5], является обобщением операции умножения: композицией мономов $U = {x_1}^{a_{11}} {x_2}^{a_{12}} \dots {x_q}^{a_{1q}}$ и $V = {x_1}^{a_{21}} {x_2}^{a_{22}} \dots {x_q}^{a_{2q}}$ относительно монома $R = {x_1}^{b_1} {x_2}^{b_2} \dots {x_q}^{b_q}$ называется моном

$$(U, V)_R = \frac{UV}{R} = {x_1}^{a_{11}+a_{21}-b_1} {x_2}^{a_{12}+a_{22}-b_2} \dots {x_q}^{a_{1q}+a_{2q}-b_q},$$

при этом степени монома R не могут превосходить соответствующих степеней мономов U и V. Если R — нулевой моном (моном с нулевым набором степеней), то соответствующая операция композиции является просто операцией умножения.

Схемой композиции будем называть такую последовательность мономов

$$S = (x_1, \dots, x_q, U_1, \dots, U_n),$$

что каждый из мономов U_k, $k = 1, \dots, n$, получается композицией двух каких-либо предшествующих мономов (не обязательно различных) относительно некоторого монома R_k (вообще говоря, своего для каждой операции композиции). Под *сложностью* $l_{sh}(S)$ *схемы композиции* S будем понимать число n.

Схема композиции S *реализует моном* U, если U является одним из мономов последовательности S (в дальнейшем будем записывать это в виде $U \in S$). Аналогично, *схема композиции* S *реализует систему мономов* M, если для каждого монома U из системы M выполнено условие $U \in S$.

Часто бывает удобно строить схемы поэтапно: например, для реализации системы мономов M сначала построить схему, реализующую некоторую вспомогательную систему мономов M_0, а затем добавить элементы, используя мономы системы M_0. Этот метод неоднократно используется и в данной работе. Из этих соображений обобщим понятие схемы следующим образом.

Схемой композиции над системой мономов $M = \{V_1, \dots, V_r\}$ будем называть такую последовательность мономов

$$S = (V_1, \dots, V_r, U_1, \dots, U_n),$$

что каждый из мономов U_k, $k = 1, \dots, n$, получается композицией двух предшествующих мономов относительно некоторого монома R_k. Под *сложностью* $l_{sh}(S)$ *схемы композиции* S *над системой мономов* M будем понимать число n.

Для системы мономов M, следуя [2], положим $l_{sh}(M) = \min l_{sh}(S)$, где минимум берётся по всем схемам композиции, реализующим систему M. Величину $l_{sh}(M)$ будем называть *сложностью реализации системы мономов* M *схемами композиции*. Аналогично, для систем мономов M и M_0 положим $l_{sh}(M/M_0) = \min l_{sh}(S)$, где минимум берётся по всем схемам композиции, реализующим систему мономов M над системой

мономов M_0. Величину $l_{sh}(M/M_0)$ будем называть *сложностью реализации системы мономов M над системой мономов M_0 схемами композиции*. Если выполнено условие $l_{sh}(S) = l_{sh}(M/M_0)$, то будем называть схему S *минимальной схемой композиции* (реализующей систему мономов M над системой мономов M_0).

Заметим, что систему мономов

$$M = \{{x_1}^{a_{11}}{x_2}^{a_{12}} \dots {x_q}^{a_{1q}}, {x_1}^{a_{21}}{x_2}^{a_{22}} \dots {x_q}^{a_{2q}}, \dots, {x_1}^{a_{p1}}{x_2}^{a_{p2}} \dots {x_q}^{a_{pq}}\}$$

можно однозначно задать с помощью матрицы (без нулевых строк) из целых неотрицательных чисел

$$A = \begin{pmatrix} a_{11} & a_{12} & \dots & a_{1q} \\ a_{21} & a_{22} & \dots & a_{2q} \\ \dots & \dots & \dots & \dots \\ a_{p1} & a_{p2} & \dots & a_{pq} \end{pmatrix}.$$

Сложностью $l_{sh}(A)$ реализации матрицы A схемами композиции будем называть сложность реализации схемами композиции соответствующей этой матрице системы мономов. Таким образом, если матрице A соответствует система мономов M_A, то $l_{sh}(A) = l_{sh}(M_A)$. В дальнейшем во всех терминах и обозначениях сложности будем допускать замену системы мономов на соответствующую матрицу (и наоборот).

2. Сравнение схем композиции с другими вычислительными моделями

Интерес к задаче о сложности вычисления систем мономов схемами композиции обусловлен несколькими факторами.

Во-первых, задача об исследовании различных свойств, связанных в том числе с вопросами сложности, известной в алгебре операции композиции [5], возникает естественным образом. Операция умножения является частным случаем операции композиции, и при вычислении одного монома от нескольких переменных, а также при вычислении системы мономов от одной переменной сложность в классе схем композиции устанавливается очень просто [1], что помогает при исследовании сложности аналогичных задач в классе схем умножения выделить содержательную часть, отделив её от технически сложной и трудоёмкой, но малосодержательной части.

Во-вторых, исследование этой задачи даёт надежду на разработку новых методов для асимптотически точного решения задачи Пиппенджера [6], т. е. задачи о сложности реализации систем мономов схемами из элементов, реализующих операцию умножения. В этом направлении есть достижения — результат об асимптотике роста функции шенноновского типа, характеризующей максимальную сложность реализации систем мономов с индивидуальными ограничениями на показатели степеней в мономах, удалось при некоторых слабых ограничениях перенести с модели схем композиции на классическую модель схем умножения [4].

В-третьих, при изучении сложности реализации систем мономов схемами композиции были обнаружены новые неожиданные эффекты, представляющие значительный интерес [2]. Ещё один из таких эффектов описан в данной работе.

Отметим особенности модели схем композиции в сравнении с другими моделями. В первую очередь рассмотрим классическую задачу о сложности возведения в степень [7, разд. 4.6.3] и её обобщения [8].

Для простейшего случая (вычисление одного монома от одной переменной x^a) задача о сложности реализации схемами композиции имеет тривиальное решение — сложность равна $\lceil \log a \rceil$ (здесь и далее под записью $\log x$ понимается $\log_2 x$). Для классиче-

ской модели установлено [9], что для вычисления величины x^a при $a \to \infty$ достаточно

$$\log a + \frac{\log a}{\log\log a}(1 + o(1))$$

операций умножения; мощностным (неконструктивным) методом доказано [10, 11], что при любом положительном ε для почти всех значений a для вычисления величины x^a требуется не менее

$$\log a + (1 - \varepsilon)\frac{\log a}{\log\log a}$$

операций умножения; при этом не известно конструктивной нижней оценки лучше, чем доказанная в [12] оценка

$$l(x^a) \geqslant \log a + \log s(a) - 2{,}13,$$

где $s(a)$ — количество единиц в двоичной записи числа a. Таким образом, для схем умножения при вычислении одной степени x^a для почти всех значений a разность между верхней и нижней оценками (а также между реальной сложностью и нижней оценкой) имеет порядок $\log a/\log\log a$.

Задачи о сложности реализации матриц размеров $1 \times q$ и $p \times 1$ (т. е. о сложности вычисления одного монома от q переменных и p степеней одной переменной соответственно) схемами композиции также имеют простое решение с точным ответом [1]. Аналогичные задачи для схем умножения — это задачи Беллмана [13] и Кнута [7, разд. 4.6.3, упр. 32]. Для них в работах [14, 15] найдена сложность при фиксированных параметрах p и q, а в [16 – 18] — асимптотика роста сложности. Здесь и далее, говоря об асимптотике роста сложности матрицы $A = (a_{ij})$, мы подразумеваем асимптотику роста сложности при условии $\sum a_{ij} \to \infty$.

Для классической модели в случае матриц размера $2 \times q$ и $p \times 2$ даже поиск асимптотики роста сложности становится значительно труднее. В работе [19] установлено, что при некоторых слабых ограничениях асимптотика роста сложности реализации матрицы A размера $p \times 2$ схемами умножения равна $\log D(A)$, где $D(A)$ — максимум абсолютных величин миноров матрицы A. При этом для схем умножения работает принцип двойственности [20]: если в матрице A размера $p \times q$ из целых неотрицательных чисел нет нулевых строк и столбцов, то $l(A) + p = l(A^{\mathrm{T}}) + q$, где $l(A)$ — сложность реализации матрицы A схемой умножения. Таким образом, применение принципа двойственности к оценке для матрицы размера $p \times 2$ сразу доставляет аналогичную оценку для матрицы размера $2 \times q$.

Как показано в работе [21], для модели с двумя операциями — умножением и делением — асимптотика сложности матрицы произвольного размера также определяется величиной $D(A)$. Таким образом, для моделей с операцией умножения и с операциями умножения и деления сложность матриц размера $2 \times q$ и $p \times 2$ определяется некоторой подматрицей размера 2×2.

Для модели с одной операцией умножения, но допускающей использование отрицательных степеней переменных, в [22] также установлен результат, показывающий, что сложность матрицы размера $2 \times q$ определяется некоторой её подматрицей размера 2×2.

Возвращаясь к задаче реализации матриц схемами композиции, отметим, что для реализации матрицы размера $2 \times q$ в [3] установлено точное значение сложности, а также показано, что эта сложность, как и во всех перечисленных случаях, определяется подматрицей размера 2×2. Аналогичное утверждение для сложности реализации

матриц размера $p \times 2$ схемами композиции не может быть получено из соображений двойственности, так как для схем композиции они не работают в достаточной степени [2]. Более того, оказывается, что в этом случае оно, вообще говоря, неверно.

Следующий пример демонстрирует, что при фиксированном p даже удаление одного произвольного монома может изменить асимптотику роста сложности.

Пусть p — чётное число. Рассмотрим матрицу

$$A = \begin{pmatrix} 2^{2n} & 1 \\ 2^{2n} & 2^{3n} \\ 2^{4n} & 2^{3n} \\ 2^{4n} & 2^{5n} \\ \dots & \dots \\ 2^{pn} & 2^{(p-1)n} \\ 2^{pn} & 2^{(p+1)n} \end{pmatrix}.$$

Пусть $\{U_1, U_2, \dots, U_p\}$ — система мономов, соответствующая матрице A. Тогда $l_{sh}(A) = l_{sh}(\{U_1, U_2, \dots, U_p\})$. Рассмотрим последовательность мономов:

$$\begin{aligned}
&S = \Big(x, y, xy, \\
&x^2y, x^4y, \dots, x^{2^{2n}-1}y, x^{2^{2n}}y, \\
&x^{2^{2n}}y^2, x^{2^{2n}}y^4, \dots, x^{2^{2n}}y^{2^{3n}-1}, x^{2^{2n}}y^{2^{3n}}, \\
&x^{2^{2n}+1}y^{2^{3n}}, x^{2^{2n}+2}y^{2^{3n}}, \dots, x^{2^{4n}-1}y^{3n}, x^{2^{4n}}y^{3n}, \\
&x^{2^{4n}}y^{2^{3n}+1}, x^{2^{4n}}y^{2^{3n}+2}, \dots, x^{2^{4n}}y^{2^{5n}-1}, x^{2^{4n}}y^{2^{5n}}, \\
&\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots \\
&x^{2^{(p-2)n}+1}y^{2^{(p-1)n}}, x^{2^{(p-2)n}+2}y^{2^{(p-1)n}}, \dots, x^{2^{pn}-1}y^{2^{(p-1)n}}, x^{2^{pn}}y^{(p-1)n}, \\
&x^{2^{pn}}y^{2^{(p-1)n}+1}, x^{2^{pn}+1}y^{2^{(p-1)n}+2}, \dots, x^{2^{pn}+1}y^{2^{(p+1)n}-1}, x^{2^{pn}+1}y^{2^{(p+1)n}}\Big).
\end{aligned}$$

Нетрудно заметить, что эта последовательность является схемой, реализующей матрицу A. Найдём сложность схемы. Так как мономы x и y «даны изначально», т. е. не учитываются при подсчёте сложности, то

$$l_{sh}(S) = 1 + 3n + 2n(p-1) = (2p+1)n + 1$$

и, поскольку схема S реализует матрицу A, то

$$l_{sh}(A) \leqslant l_{sh}(S) = (2p+1)n + 1.$$

Введём обозначение: для произвольных мономов U_1 и U_2 будем говорить, что *моном U_1 содержится в мономе U_2* (или что *моном U_2 содержит моном U_1*), и писать $U_1 \leqslant U_2$, если все показатели степеней монома U_1 не превосходят соответствующих показателей степеней монома U_2. Если это условие не выполнено, то будем говорить, что *моном U_1 не содержится в мономе U_2* (*моном U_2 не содержит моном U_1*), и писать $U_1 \nleqslant U_2$.

Установим нижнюю оценку для величины $l_{sh}(A)$. Пусть S_0 — минимальная схема, реализующая матрицу A. Рассмотрим систему мономов $M_1 = \{U \in S_0 : U \leqslant U_1\}$ и последовательность мономов S_1, состоящую из всех мономов системы M_1, упорядоченных так же, как в схеме S_0. Легко видеть, что последовательность S_1 является схемой,

реализующей моном U_1. Очевидно, для реализации этого монома потребуется как минимум $2n+1$ операций: $2n$ операций для увеличения степени с 1 до $2n$ и ещё одна операция для «объединения» переменных x и y. Таким образом,

$$l_{sh}(S_1) \geqslant 2n+1.$$

Теперь рассмотрим систему мономов $M_2 = \{U \in S_0 : U \leqslant U_2\}$ и последовательность мономов S_2, состоящую из всех мономов системы M_2, упорядоченных так же, как и в схеме S_0. Легко видеть, что последовательность S_2 является схемой, реализующей моном U_2 над системой мономов M_1. Очевидно, для такой схемы потребуется по крайней мере $3n$ операций, то есть

$$l_{sh}(S_1) \geqslant 3n.$$

Продолжая это рассуждение, будем для $i = 3, \ldots, p$ рассматривать систему мономов $M_i = \{U \in S_0 : U \leqslant U_i\}$ и последовательность мономов S_i, состоящую из всех мономов системы M_i, упорядоченных так же, как и в схеме S_0. Последовательность S_i является схемой, реализующей моном U_i над системой мономов $M_1 \cup M_2 \cup \ldots \cup M_{i-1}$. При этом

$$l_{sh}(S_i) \geqslant 2n, \quad i = 3, \ldots, p.$$

Так как любой моном схемы S_0 учитывается при подсчёте сложности только одной из схем $S_1, S_2, \ldots, S_p$, то

$$l_{sh}(S_0) \geqslant \sum_{i=1}^{p} l_{sh}(S_i).$$

Отсюда, используя минимальность схемы S_0 и нижние оценки сложности схем S_i, $i = 1, \ldots, p$, получаем

$$l_{sh}(A) = l_{sh}(S_0) \geqslant \sum_{i=1}^{p} l_{sh}(S_i) \geqslant 2n + 1 + 3n + 2n(p-1) = (2p+1)n + 1.$$

Верхняя и нижняя оценки совпали. Таким образом, мы нашли сложность реализации матрицы A:

$$l_{sh}(A) = (2p+1)n + 1.$$

Обозначим через $A^{(k)}$ матрицу, полученную из A удалением k-й строки. Находя сложность этой матрицы аналогичным способом, получим

$$l_{sh}(A^{(k)}) = (2p-1)n + 1.$$

Вычислим соотношение сложностей данных матриц:

$$\frac{l_{sh}(A)}{l_{sh}(A^{(k)})} = \frac{(2p+1)n+1}{(2p-1)n+1} = 1 + \frac{2n}{(2p-1)n+1} = 1 + \frac{1}{p - 1/2 + 1/(2n)}.$$

Рассмотрим последовательности матриц A_n и $A_n^{(k_n)}$, которые получаются из матриц A и $A^{(k)}$ соответственно при $n = 1, 2, 3, \ldots$ и произвольных $k_n \in \{1, 2, \ldots, p\}$. Тогда

$$\lim_{n\to\infty} \frac{l_{sh}(A_n)}{l_{sh}(A_n^{(k_n)})} = 1 + \frac{1}{p - 1/2},$$

т. е. асимптотика роста сложности последовательности матриц $A_n^{(k_n)}$ отличается от асимптотики роста сложности последовательности матриц A_n.

Этот пример легко обобщается на случай нечётного p.

Приведённый пример показывает, что при реализации системы мономов, задаваемой матрицей размера $p \times 2$, схемами композиции, в отличие от моделей с операцией умножения и с операциями умножения и деления, асимптотика роста сложности не только не определяется подматрицей размера 2×2, но даже, вообще говоря, не определяется никакой подматрицей размера $(p-1) \times 2$. Поэтому ожидать простой и компактной формулы, определяющей асимптотику роста сложности реализации матриц размера $p \times 2$ схемами композиции, не приходится. В данной работе эта асимптотика найдена с точностью до слагаемого порядка p, и соответствующий результат сформулирован в терминах соотношений элементов в каждом столбце.

3. Вспомогательные утверждения

Для доказательства основных результатов потребуется ряд вспомогательных утверждений (леммы 1–10). Доказательство леммы 2 можно найти в работе [2], доказательства лемм 3–5, а также другое доказательство леммы 2 — в [3].

Для произвольного числа a будем использовать обозначение $a^+ = \max(a, 1)$. В лемме 5 и всюду далее S_p обозначает группу перестановок множества $\{1, 2, \ldots, p\}$, а максимум, берущийся по пустому множеству элементов, считается равным единице.

Пусть даны схема

$$S_1 = (U_1, \ldots, U_s, U_{s+1}, \ldots, U_{s+l_1})$$

над системой мономов $\{U_1, \ldots, U_s\}$ и схема

$$S_2 = (V_1, \ldots, V_t, V_{t+1}, \ldots, V_{t+l_2})$$

над системой мономов $\{V_1, \ldots, V_t\}$, и при этом каждый моном системы $\{V_1, \ldots, V_t\}$ содержится в схеме S_1. Легко видеть, что тогда последовательность мономов

$$(U_1, \ldots, U_s, U_{s+1}, \ldots, U_{s+l_1}, V_{t+1}, \ldots, V_{t+l_2})$$

является схемой сложности $l_1 + l_2$ над системой мономов $\{U_1, \ldots, U_s\}$. Построенную таким образом схему будем обозначать через $S(S_1; S_2)$. Отмеченный факт сформулируем в виде утверждения.

Лемма 1. Пусть S_1 — схема, реализующая систему мономов M_1 над системой мономов M_0, а S_2 — схема, реализующая систему мономов M_2 над системой мономов M_1', причём $M_1' \subseteq M_1$. Тогда существует такая схема S_{12}, реализующая систему мономов M_2 над системой мономов M_0, что

$$l_{sh}(S_{12}) = l_{sh}(S_1) + l_{sh}(S_2).$$

Приведём несколько известных утверждений.

Лемма 2 [1, теорема 2]. Для монома $U = {x_1}^{a_1} \ldots {x_q}^{a_q}$ справедливо равенство

$$l_{sh}(U) = \left\lceil \log \max_{k:1 \leqslant k \leqslant q} a_k \right\rceil + q - 1.$$

Лемма 3 [3, лемма 3]. Пусть S — минимальная схема, реализующая систему мономов M над системой мономов M_0, мономы из системы M_0 не имеют общих переменных, моном U_0 содержится в каждом из мономов системы M, а также является

элементом схемы S. Тогда

$$l_{sh}(M/M_0) = l_{sh}(U_0/M_0') + l_{sh}(M/M_0''),$$

где $M_0' = \{U \in M_0 : U \leqslant U_0\}$, $M_0'' = (M_0 \setminus M_0') \cup \{U_0\}$.

Лемма 4 [3, лемма 4]. Пусть мономы $U = {x_1}^{a_1}{x_2}^{a_2}\dots{x_q}^{a_q}$ и $V = {x_1}^{b_1}{x_2}^{b_2}\dots{x_q}^{b_q}$ удовлетворяют условиям $a_k \geqslant b_k > 0$, $k = 1, \dots, q$. Тогда

$$l_{sh}(U/V) = \left\lceil \log \max_{k:1\leqslant k\leqslant q} \frac{a_k}{b_k} \right\rceil.$$

Лемма 5 [3, лемма 7]. Пусть в матрице

$$A = \begin{pmatrix} a_{11} & a_{12} & \dots & a_{1q} \\ a_{21} & a_{22} & \dots & a_{2q} \\ \dots & \dots & \dots & \dots \\ a_{p1} & a_{p2} & \dots & a_{pq} \end{pmatrix}$$

все элементы — натуральные числа. Тогда

$$l_{sh}(A/x_1x_2\dots x_q) \geqslant \max_{(i_1,\dots,i_p)\in S_p} \sum_{k=1}^{p} \left\lceil \log \max \left(\max_{j:1\leqslant j\leqslant q} \left(\frac{a_{i_kj}}{\max\limits_{l:l<k} a_{i_lj}} \right), 1 \right) \right\rceil.$$

Лемма 6. Пусть A — матрица размера $p \times 2$ из натуральных чисел. Тогда

$$l_{sh}(A) = l_{sh}(A/xy) + 1.$$

Доказательство. Верхняя оценка следует из леммы 1.

Докажем нижнюю оценку. Пусть $\{U_1, \dots, U_p\}$ — система мономов, соответствующая матрице A, и S — минимальная схема, реализующая A. Рассмотрим множества

$$S_k = \{U \in S : U \leqslant U_k\}, \quad k = 1, \dots, p.$$

В каждом из этих множеств есть хотя бы один моном вида $x^a y$ $(a > 0)$. Среди всех мономов такого вида в схеме S выберем моном с наименьшим показателем a и обозначим его через U_0. Очевидно, тогда $U_0 \leqslant U_k$, $k = 1, \dots, p$. Используя леммы 1–3, получаем

$$l_{sh}(A) = l_{sh}(A/U_0) + l_{sh}(U_0) = l_{sh}(A/U_0) + l_{sh}(U_0/xy) + 1 \geqslant l_{sh}(A/xy) + 1,$$

что и требовалось. ■

Следующая лемма обобщает утверждение леммы 5 на случай матрицы с нулевыми элементами.

Лемма 7. Пусть в матрице

$$A = \begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \\ \dots & \dots \\ a_{p1} & a_{p2} \end{pmatrix}$$

все элементы — целые неотрицательные числа. Тогда

$$l_{sh}(A) \geqslant \max_{(i_1,\dots,i_p)\in S_p} \sum_{k=1}^{p} \left\lceil \log \max \left(\frac{a^+_{i_k1}}{\max\limits_{l:l<k} a^+_{i_l1}}, \frac{a^+_{i_k2}}{\max\limits_{l:l<k} a^+_{i_l2}}, 1 \right) \right\rceil.$$

Доказательство. Пусть $S = (x, y, U_1, U_2, \dots U_l)$ — минимальная схема, реализующая матрицу A. Для произвольного монома $U = x^a y^b$ положим

$$U' = \begin{cases} xy^b, & \text{если } a = 0; \\ x^a y, & \text{если } b = 0; \\ x^a y^b & \text{иначе.} \end{cases}$$

Легко видеть, что тогда последовательность мономов $S' = (xy, U'_1, U'_2, \dots U'_l)$ является схемой, реализующей матрицу $A' = (a^+_{ij})$ над мономом xy. Отсюда, используя леммы 5 и 6, получаем

$$l_{sh}(A) = l_{sh}(S) = l_{sh}(S') \geqslant l_{sh}(A'/xy) \geqslant \max_{(i_1,\dots,i_p)\in S_p} \sum_{k=1}^{p} \left\lceil \log \max \left(\frac{a^+_{i_k 1}}{\max\limits_{l:l<k} a^+_{i_l 1}}, \frac{a^+_{i_k 2}}{\max\limits_{l:l<k} a^+_{i_l 2}}, 1 \right) \right\rceil,$$

что и требовалось. ■

Лемма 8. Пусть для мономов $V = x^{a_1} y^{a_2}$, $V_0 = x^{b_1} y^{b_2}$, $V_1 = x^{a_{11}} y^{a_{12}}$ и $V_2 = x^{a_{21}} y^{a_{22}}$ выполнены условия

$$0 < \max(a_{11}, a_{21}) \leqslant b_1 \leqslant a_1, \quad 0 < \max(a_{12}, a_{22}) \leqslant b_2 \leqslant a_2, \quad \frac{b_1}{2} < a_{11}, \quad \frac{b_2}{2} < a_{22}.$$

Тогда существует схема S, реализующая моном V над системой мономов $\{V_1, V_2\}$ и удовлетворяющая следующим условиям:

1) Для сложности схемы S верно равенство

$$l_{sh}(S) = \left\lceil \log \max \left(\frac{a_1}{b_1}, \frac{a_2}{b_2} \right) \right\rceil + 2.$$

2) Для любого натурального числа c_1, для которого справедливы неравенства

$$b_1 \leqslant c_1 \leqslant a_1, \quad \frac{a_2}{b_2} \leqslant \frac{c_1}{b_1}, \tag{1}$$

существуют такие натуральные числа d_1, d_2, что

$$\frac{c_1}{2} < d_1 \leqslant c_1, \quad \frac{a_2}{2} < d_2 \leqslant a_2 \tag{2}$$

и $x^{d_1} y^{d_2} \in S$.

Доказательство. Без ограничения общности будем считать, что $\dfrac{a_1}{b_1} \geqslant \dfrac{a_2}{b_2}$ (отметим, что это неравенство следует из условий (1)). Положим $n = \left\lceil \log \dfrac{a_1}{b_1} \right\rceil$. Искомая схема будет иметь вид

$$S = (x^{a_{11}} y^{a_{12}}, x^{a_{21}} y^{a_{22}}, x^{a_{11}} y^{a_{22}}, x^{b_1} y^{b_2}, x^{b_{11}} y^{b_{12}}, x^{b_{21}} y^{b_{22}}, \dots, x^{b_{n1}} y^{b_{n2}}),$$

где $b_{ij} = \min(2^i b_j, a_j)$, $i = 1, \dots, n$, $j = 1, 2$. Легко видеть, что $b_{n1} = a_1$ и $b_{n2} = a_2$, т. е. схема S реализует моном V.

Пусть теперь для натурального числа c_1 выполнены условия (1). Положим $m = \left\lfloor \log \frac{c_1}{b_1} \right\rfloor$ и заметим, что из условий леммы следует неравенство $m \leqslant n$. Покажем, что для чисел $d_1 = b_{m1}$ и $d_2 = b_{m2}$ выполнены условия (2). Действительно,

$$b_{m1} \leqslant 2^m b_1 = 2^{\lfloor \log(c_1/b_1) \rfloor} b_1 \leqslant 2^{\log(c_1/b_1)} b_1 = c_1,$$

$$b_{m1} = \min\left(2^{\lfloor \log(c_1/b_1) \rfloor} b_1, a_1\right) > \min\left(2^{\log(c_1/b_1)-1} b_1, a_1\right) = \min\left(\frac{c_1}{2}, a_1\right) = \frac{c_1}{2},$$

$$b_{m2} \leqslant a_2,$$

$$b_{m2} = \min\left(2^{\lfloor \log(c_1/b_1) \rfloor} b_2, a_2\right) > \min\left(2^{\log(c_1/b_1)-1} b_2, a_2\right) = \min\left(\frac{c_1 b_2}{2b_1}, a_2\right) \geqslant \min\left(\frac{a_2}{2}, a_2\right) = \frac{a_2}{2},$$

что и требовалось. ■

Следующая лемма уточняет утверждение леммы 9 для случая, когда мономы V_0, V_1 и V_2 совпадают.

Лемма 9. Пусть для мономов $V = x^{a_1} y^{a_2}$ и $V_0 = x^{b_1} y^{b_2}$ выполнены условия

$$0 < b_1 \leqslant a_1, \quad 0 \leqslant b_2 \leqslant a_2.$$

Тогда существует схема S, реализующая моном V над мономом V_0 и удовлетворяющая следующим условиям:

1) Для сложности схемы S верно равенство

$$l_{sh}(S) = \left\lceil \log \max\left(\frac{a_1}{b_1}, \frac{a_2}{b_2}\right) \right\rceil.$$

2) Для любого натурального числа c_1, для которого справедливы неравенства

$$b_1 \leqslant c_1 \leqslant a_1, \quad \frac{a_2}{b_2} \leqslant \frac{c_1}{b_1},$$

существуют такие натуральные числа d_1, d_2, что

$$\frac{c_1}{2} < d_1 \leqslant c_1, \quad \frac{a_2}{2} < d_2 \leqslant a_2$$

и $x^{d_1} y^{d_2} \in S$.

Также сформулируем аналог леммы 9 для случая мономов от одной переменной.

Лемма 10. Пусть $0 < b \leqslant a$. Тогда существует такая схема S, реализующая моном x^a над мономом x^b, что

$$l_{sh}(S) = \left\lceil \log \frac{a}{b} \right\rceil.$$

Кроме того, если для натурального числа c выполнено условие $b \leqslant c \leqslant a$, то схема S содержит моном x^d, для которого выполнено условие $c/2 < d \leqslant c$.

4. Формулировка и доказательство основных результатов

Сформулируем и докажем основной результат для случая матрицы без нулевых элементов.

Теорема 1. Пусть в матрице

$$A = \begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \\ \dots & \dots \\ a_{p1} & a_{p2} \end{pmatrix}$$

все элементы — натуральные числа. Тогда

$$F(A) + 1 \leqslant l_{sh}(A) \leqslant F(A) + 2p - 3,$$

где

$$F(A) = \max_{(i_1,\dots,i_p)\in S_p} \sum_{k=1}^{p} \left\lceil \log \max \left(\frac{a_{i_k 1}}{\max\limits_{l:l<k} a_{i_l 1}}, \frac{a_{i_k 2}}{\max\limits_{l:l<k} a_{i_l 2}}, 1 \right) \right\rceil.$$

Доказательство. Нижняя оценка следует из лемм 5 и 6.

Докажем верхнюю оценку. Без ограничения общности будем считать, что в матрице A все строки различны. Определим перестановку $(j_1, j_2, \dots, j_p)$ строк матрицы A по следующим индуктивным правилам.

Рассмотрим все строки матрицы A, содержащие её минимальный элемент. Среди них выберем такую строку, чтобы другой (оставшийся) элемент этой строки был минимально возможным (если таких строк две, то берём любую из них). Обозначим номер этой строки через j_1.

Пусть уже определён набор $(j_1, \dots, j_{i-1})$. Положим

$$K_i = \{1, 2, \dots, p\} \setminus \{j_1, \dots, j_{i-1}\}, \quad b_{il} = \max_{j\in\{j_1,\dots,j_{i-1}\}} a_{jl}, \quad l = 1, 2.$$

Рассмотрим все пары (j, l), $j \in K_i$, $l \in \{1, 2\}$, на которых достигается минимум величины a_{jl}/b_{il}. Среди этих пар выберем такую, чтобы величина $a_{j,3-l}$ принимала минимально возможное значение (если таких пар две, то берём любую из них). Обозначим первый элемент выбранной пары через j_i.

Так как изменение порядка строк не влияет на сложность реализации матрицы, то без ограничения общности будем считать, что $j_1 = 1$, $j_2 = 2$, …, $j_p = p$.

Пусть $\{U_1, U_2, \dots, U_p\}$ — система мономов, соответствующая матрице A. Заданный порядок удовлетворяет следующим условиям:

1) Если $1 \leqslant i < j \leqslant p$, то $U_j \not\leqslant U_i$.
2) Для всех $i = 1, \dots, p$ выполнено равенство

$$\min_{l=1,2} \frac{a_{il}}{b_{il}} = \min_{j:i\leqslant j\leqslant p} \min_{l=1,2} \frac{a_{jl}}{b_{il}}. \tag{3}$$

Обозначим схему (x, y, xy) через $\widetilde{S}_0$ и положим

$$t_1 = \lceil \log \max (a_{11}, a_{12}) \rceil, \quad t_2 = \lceil \log \max (a_{21}, a_{22}) \rceil,$$
$$t_{12} = \left\lceil \log \max \left(\frac{a_{11}}{a_{21}}, \frac{a_{12}}{a_{22}} \right) \right\rceil, \quad t_{21} = \left\lceil \log \max \left(\frac{a_{21}}{a_{11}}, \frac{a_{22}}{a_{12}} \right) \right\rceil.$$

Рассмотрим все возможные случаи соотношения мономов U_1 и U_2:

1. Пусть один из мономов U_1, U_2 не превосходит другого.

1.1) Пусть выполняется соотношение $U_1 \leqslant U_2$.

Применяя лемму 9, построим схему S_1, реализующую моном U_1 над мономом xy, и схему S_2, реализующую моном U_2 над мономом U_1. Положим $\widetilde{S}_1 = S(\widetilde{S}_0; S_1)$, $\widetilde{S}_2 = S(\widetilde{S}_1; S_2)$. Тогда

$$l_{sh}(S_1) + l_{sh}(S_2) = t_1 + t_{21}.$$

1.2) Пусть выполняется соотношение $U_2 \leqslant U_1$.

Применяя лемму 9, построим схему S_2, реализующую моном U_2 над мономом xy, и схему S_1, реализующую моном U_1 над мономом U_2. Положим $\widetilde{S}_1 = S(\widetilde{S}_0; S_2)$, $\widetilde{S}_2 = S(\widetilde{S}_1; S_1)$. Тогда

$$l_{sh}(S_1) + l_{sh}(S_2) = t_2 + t_{12}.$$

2. Пусть выполняются соотношения $U_1 \not\leqslant U_2$ и $U_2 \not\leqslant U_1$.

Положим $c = 2^{\max(t_1 - t_{12}, t_2 - t_{21})}$, $U_0 = x^{\min(c, a_{11}, a_{21})} y^{\min(c, a_{12}, a_{22})}$.

Применяя лемму 9, построим схему S_0, реализующую моном U_0 над мономом xy, схему S_{01}, реализующую моном U_1 над мономом U_0, и схему S_2, реализующую моном U_2 над мономом U_0. Положим $S_1 = S(\widetilde{S}_0; S_{01})$, $\widetilde{S}_1 = S(\widetilde{S}_0; S_1)$, $\widetilde{S}_2 = S(\widetilde{S}_1; S_2)$.

2.1) Пусть выполняется неравенство $t_1 - t_{12} \geqslant t_2 - t_{21}$. В этом случае

$$l_{sh}(S_0) = \lceil \log \max(\min(c, a_{11}, a_{21}), \min(c, a_{12}, a_{22})) \rceil \leqslant \log c = t_1 - t_{12},$$

$$l_{sh}(S_{01}) = \left\lceil \log \max\left(\frac{a_{11}}{\min(c, a_{11}, a_{21})}, \frac{a_{12}}{\min(c, a_{12}, a_{22})} \right) \right\rceil = \left\lceil \log \max\left(\frac{a_{11}}{c}, \frac{a_{11}}{a_{12}}, \frac{a_{12}}{c}, \frac{a_{12}}{a_{22}} \right) \right\rceil =$$

$$= \max\left(\left\lceil \log \max\left(\frac{a_{11}}{c}, \frac{a_{12}}{c} \right) \right\rceil, \left\lceil \log \max\left(\frac{a_{11}}{a_{21}}, \frac{a_{12}}{a_{22}} \right) \right\rceil \right) = \max(t_1 - (t_1 - t_{12}), t_{12}) = t_{12},$$

$$l_{sh}(S_2) = \left\lceil \log \max\left(\frac{a_{21}}{\min(c, a_{11}, a_{21})}, \frac{a_{22}}{\min(c, a_{12}, a_{22})} \right) \right\rceil = \left\lceil \log \max\left(\frac{a_{21}}{c}, \frac{a_{21}}{a_{11}}, \frac{a_{22}}{c}, \frac{a_{22}}{a_{12}} \right) \right\rceil =$$

$$= \max\left(\left\lceil \log \max\left(\frac{a_{21}}{c}, \frac{a_{22}}{c} \right) \right\rceil, \left\lceil \log \max\left(\frac{a_{21}}{a_{11}}, \frac{a_{22}}{a_{12}} \right) \right\rceil \right) =$$

$$= \max(t_2 - (t_1 - t_{12}), t_{21}) \leqslant \max(t_2 - (t_2 - t_{21}), t_{21}) = t_{21},$$

и поэтому

$$l_{sh}(S_1) + l_{sh}(S_2) = l_{sh}(S_0) + l_{sh}(S_{01}) + l_{sh}(S_2) \leqslant t_1 + t_{21}.$$

2.2) Пусть выполняется неравенство $t_1 - t_{12} \geqslant t_2 - t_{21}$. В этом случае аналогично получаем

$$l_{sh}(S_1) + l_{sh}(S_2) \leqslant t_2 + t_{12}.$$

Таким образом, в любом случае схема $\widetilde{S}_2$ реализует систему мономов $\{U_1, U_2\}$ и при этом справедливо неравенство

$$l_{sh}(\widetilde{S}_2) \leqslant \max(t_1 + t_{21}, t_2 + t_{12}) + 1. \tag{4}$$

Будем индуктивно строить последовательность схем $\widetilde{S}_3, \widetilde{S}_4, \ldots, \widetilde{S}_p$, где схема $\widetilde{S}_i$, $i = 3, \ldots, p$, реализует систему мономов $\{U_1, U_2, \ldots, U_i\}$.

Пусть схема $\widetilde{S}_{i-1}$ уже построена. Построим схему S_i, реализующую моном U_i над некоторой системой мономов $\{V_1, V_2\}$ (вообще говоря, своей для каждого шага), содержащейся в схеме $\widetilde{S}_{i-1}$. Выберем такое $k < i$, что U_k — наименьший из мономов вида $x^{b_{i1}} y^b$. Опять рассмотрим все возможные случаи:

1. Если $a_{k1} \leqslant a_{i1}$ и $a_{k2} \leqslant a_{i2}$, то положим $V_1 = U_k$.

2. Если $a_{k1} \leqslant a_{i1}$ и $a_{k2} > a_{i2}$, то из условия (3) следует, что $a_{k1}/b_{k1} \leqslant a_{i2}/b_{k2}$. Отсюда, используя лемму 8 и учитывая построение схемы S_k, получаем, что эта схема содержит моном $V_1 = x^{d_1}y^{d_2}$, где

$$a_{k1}/2 < d_1 \leqslant a_{k1}, \quad a_{i2}/2 < d_2 \leqslant a_{i2}.$$

3. Если $a_{k1} > a_{i1}$ и $a_{k2} \leqslant a_{i2}$, то из условия (3) следует, что $a_{k2}/b_{k2} \leqslant a_{i1}/b_{k1}$. Отсюда, используя лемму 8 и учитывая построение схемы S_k, получаем, что эта схема содержит моном $V_1 = x^{d_1}y^{d_2}$, где

$$a_{i1}/2 < d_1 \leqslant a_{i1}, \quad a_{k2}/2 < d_2 \leqslant a_{k2}.$$

Аналогично, рассматривая степени при переменной y, выберем в схеме $\widetilde{S}_i$ моном V_2.

Применяя лемму 8 для мономов $V = U_i$, $V_0 = x^{\min(a_{i1},b_{i1})}y^{\min(a_{i2},b_{i2})}$, V_1 и V_2, построим схему S_i, реализующую моном U_i над системой мономов $\{V_1, V_2\}$, для которой выполнено равенство

$$l_{sh}(S_i) = \left\lceil \log \max\left(\frac{a_{i1}}{\min(a_{i1}, b_{i1})}, \frac{a_{i2}}{\min(a_{i2}, b_{i2})}\right)\right\rceil + 2. \tag{5}$$

Положим $\widetilde{S}_i = S(\widetilde{S}_{i-1}; S_i)$. Тогда схема $\widetilde{S}_i$ реализует систему мономов $\{U_1, \ldots, U_i\}$, при этом

$$l_{sh}(\widetilde{S}_i) = l_{sh}(\widetilde{S}_{i-1}) + l_{sh}(S_i). \tag{6}$$

Наконец, используя (4)–(6), получаем

$$l_{sh}(A) \leqslant l_{sh}(\widetilde{S}_p) = l_{sh}(\widetilde{S}_2) + \sum_{i=3}^{p} l_{sh}(S_i) \leqslant$$

$$\leqslant \sum_{i=1}^{p}\left\lceil \log \max\left(\frac{a_{i1}}{\min(a_{i1}, b_{i1})}, \frac{a_{i2}}{\min(a_{i2}, b_{i2})}\right)\right\rceil + 2p - 3 =$$

$$= \sum_{i=1}^{p}\left\lceil \log \max\left(\frac{a_{i1}}{b_{i1}}, \frac{a_{i2}}{b_{i2}}, 1\right)\right\rceil + 2p - 3 = \sum_{i=1}^{p}\left\lceil \log \max\left(\frac{a_{i1}}{\max\limits_{j<i} a_{j1}}, \frac{a_{i2}}{\max\limits_{j<i} a_{j2}}, 1\right)\right\rceil + 2p - 3 \leqslant$$

$$\leqslant \max_{(i_1,\ldots,i_p)\in S_p} \sum_{k=1}^{p}\left\lceil \log \max\left(\frac{a_{i_k 1}}{\max\limits_{l:l<k} a_{i_l 1}}, \frac{a_{i_k 2}}{\max\limits_{l:l<k} a_{i_l 2}}, 1\right)\right\rceil + 2p - 3,$$

что и требовалось. ■

Следующая теорема обобщает результат теоремы 1 на случай матрицы с нулевыми элементами.

Теорема 2. Пусть в матрице

$$A = \begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \\ \ldots & \ldots \\ a_{p1} & a_{p2} \end{pmatrix}$$

нет нулевых строк и столбцов и все её элементы — целые неотрицательные числа. Тогда

$$G(A) \leqslant l_{sh}(A) \leqslant G(A) + 2p - 3,$$

где

$$G(A) = \max_{(i_1,\ldots,i_p)\in S_p} \sum_{k=1}^{p} \left\lceil \log \max \left(\frac{a^+_{i_k 1}}{\max\limits_{l:l<k} a^+_{i_l 1}}, \frac{a^+_{i_k 2}}{\max\limits_{l:l<k} a^+_{i_l 2}}, 1 \right) \right\rceil.$$

Доказательство. Нижняя оценка следует из леммы 7. Верхняя оценка доказывается аналогично верхней оценке теоремы 1: в первую очередь нумеруются строки, содержащие нуль; для соответствующих мономов вместо лемм 8 и 9 используется лемма 10; если моном U_i содержит обе переменные, а все предыдущие мономы не содержат переменной x (или y), то для построения схемы S_i в качестве монома V_1 (соответственно V_2) используется моном x (соответственно y). ■

5. Достижимость верхних и нижних оценок

Легко видеть, что нижняя оценка теорем 1 и 2 достигается на матрицах

$$\begin{pmatrix} 2 & 2 \\ 4 & 4 \\ 8 & 8 \\ \ldots & \ldots \\ 2^{p-2} & 2^{p-2} \\ 2^{p-1} & 2^{p-1} \\ 2^p & 2^p \end{pmatrix} \quad \text{и} \quad \begin{pmatrix} 2 & 0 \\ 4 & 0 \\ 8 & 0 \\ \ldots & \ldots \\ 2^{p-2} & 0 \\ 2^{p-1} & 0 \\ 0 & 2 \end{pmatrix}$$

соответственно. Приведём пример матрицы, на которой достигается верхняя оценка. Пусть

$$B = (b_{ij}) = \begin{pmatrix} 2^p & 1 \\ 1 & 2^p \\ 7 & 7 \\ 15 & 15 \\ \ldots & \ldots \\ 2^{p-1}-1 & 2^{p-1}-1 \\ 2^p-1 & 2^p-1 \end{pmatrix}.$$

Нетрудно проверить, что максимум в верхней оценке теоремы для матрицы B достигается при $(i_1, i_2, \ldots, i_p) = (1, 2, \ldots, p)$ и равен $4p-3$. Покажем, что $l_{sh}(A) \geqslant$ $\geqslant 4p-3$. Пусть S — минимальная схема, реализующая матрицу B над мономом xy, и $\{U_1, U_2, \ldots U_p\}$ — система мономов, соответствующая матрице B. Положим

$$M_1 = \{U \in S : U \leqslant U_1\}, \quad M_2 = \{U \in S : U \leqslant U_2\}.$$

Рассмотрим последовательности мономов S_1 и S_2, состоящие из всех мономов систем M_1 и M_2 соответственно, упорядоченных так же, как в схеме S. Легко видеть, что последовательности S_1 и S_2 являются схемами, реализующими мономы U_1 и U_2 соответственно, над мономом xy.

Если для некоторого k, $k = 3, \ldots, p$, выполнено условие, что схема S_1 содержит хотя бы один из мономов $x^{2^k-1}y$ или $x^{2^k-2}y$, а схема S_2 содержит хотя бы один из мономов xy^{2^k-1} или xy^{2^k-2}, то

$$l_{sh}(U_k/M_1 \cup M_2 \cup \{U_{k-1}\}) = 1.$$

Если это условие не выполнено, то

$$l_{sh}(U_k/M_1 \cup M_2 \cup \{U_{k-1}\}) = 2.$$

Пусть условие выполнено ровно для s чисел из набора $\{3, \dots, p\}$. Тогда, применяя лемму 4 для всех соответствующих мономов схем S_1 и S_2, получаем

$$l_{sh}(S_1) \geqslant p + s, \quad l_{sh}(S_2) \geqslant p + s.$$

Из этих оценок следует, что

$$l_{sh}(S) = l_{sh}(S_1) + l_{sh}(S_2) + \sum_{k=3}^{p} l_{sh}(U_k/M_1 \cup M_2 \cup \{U_{k-1}\}) \geqslant$$
$$\geqslant 2(p + s) + s + 2(p - 2 - s) = 4p + s - 4 \geqslant 4p - 4.$$

Отсюда, используя лемму 6 и минимальность схемы S, получаем

$$l_{sh}(B) = l_{sh}(B/xy) + 1 = l_{sh}(S) + 1 \geqslant 4p - 3,$$

что и требовалось.

ЛИТЕРАТУРА

1. *Мерекин Ю. В.* О порождении слов с использованием операции композиции // Дискретный анализ и исследование операций. Сер. 1. 2003. Т. 10. № 4. С. 70–78.
2. *Трусевич Е. Н.* О сложности вычисления некоторых систем одночленов схемами композиции // Вестник Московского университета. Сер. 1. Математика. Механика. 2014. № 5. С. 18–22.
3. *Корнеев С. А.* О сложности реализации системы из двух мономов схемами композиции // Дискретная математика. 2020. Т. 32. № 2. С. 15–31.
4. *Корнеев С. А.* Об асимптотическом поведении функций шенноновского типа, характеризующих сложность вычисления систем мономов // Ученые записки Казанского университета. Сер. Физико-математические науки. 2020. Т. 162. № 3. С. 300–311.
5. *Ширшов А. И.* Некоторые алгоритмические проблемы для алгебр Ли // Сибирский математический журнал. 1962. Т. 3. С. 292–296.
6. *Pippenger N.* On the evaluation of powers and monomials // SIAM J. Comput. 1980. V. 9. No. 2. P. 230–250.
7. *Кнут Д. Е.* Искусство программирования для ЭВМ. Т. 2. М.: Мир, 1977.
8. *Кочергин В. В.* О задачах Беллмана и Кнута и их обобщениях // Фундаментальная и прикладная математика. 2015. Т. 20. № 6. С. 159–188.
9. *Brauer A.* On addition chains // Bull. AMS. 1939. V. 45. P. 736–739.
10. *Erdos P.* Remarks on number theory, III: On addition chains // Acta Arithmetica. 1960. V. 6. P. 77–81.
11. *Кочергин В. В., Кочергин Д. В.* Уточнение нижней оценки сложности возведения в степень // Прикладная дискретная математика. 2017. № 38. С. 119–132.
12. *Schönhage A.* A lower bound for the lenght of addition chains // Theor. Comput. Sci. 1975. V. 1. P. 1–12.
13. *Bellman R. E.* Addition chains of vectors (Advanced problems 5125) // Amer. Math. Monthly. 1963. V. 70. P. 765.
14. *Straus E. G.* Addition chains of vectors // Amer. Math. Monthly. 1964. V. 71. P. 806–808.
15. *Yao A. C.-C.* On the evaluation of powers // SIAM J. Computing. 1976. V. 5. P. 100–103.
16. *Гашков С. Б., Кочергин В. В.* Об аддитивных цепочках векторов, вентильных схемах и сложности вычисления степеней // Методы дискретного анализа в теории графов и сложности. 1992. Вып. 52. С. 22–40.

17. *Кочергин В. В.* О сложности вычислений одночленов и наборов степеней // Тр. Ин-та математики СО РАН. 1994. Т. 27. С. 94–107.
18. *Кочергин В. В.* Уточнение оценок сложности вычисления одночленов и наборов степеней в задачах Беллмана и Кнута // Дискретный анализ и исследование операций. 2014. Т. 21. № 6. С. 51–72.
19. *Кочергин В. В.* О сложности вычисления систем одночленов от двух переменных // Труды VII Междунар. конф. «Дискретные модели в теории управляющих систем» (Покровское, 4–6 марта 2006). М.: МАКС Пресс, 2006. С. 185–190.
20. *Сидоренко А. Ф.* Сложность аддитивных вычислений семейства целочисленных линейных форм // Записки научных семинаров ЛОМИ. Л.: Наука, 1981. Т. 105. С. 53–61.
21. *Кочергин В. В.* Об асимптотике сложности аддитивных вычислений систем целочисленных линейных форм // Дискретный анализ и исследование операций. Сер. 1. 2006. Т. 13. № 2. С. 38–58.
22. *Кочергин В. В.* О сложности совместного вычисления трех элементов свободной абелевой группы с двумя образующими // Дискретный анализ и исследование операций. Сер. 1. 2008. Т. 15. № 2. С. 23–64.

REFERENCES

1. *Merekin Yu. V.* O porozhdenii slov s ispol'zovaniyem operatsii kompozitsii [On the generation of words using the composition operation]. Diskretnyy Analiz i Issledovaniye Operatsiy. Ser. 1, 2003. vol. 10, no. 4., pp. 70–78. (in Russian)
2. *Trusevich E. N.* Complexity of certain systems of monomials in calculation by composition circuits. Moscow University Math. Bull., 2014, vol. 69, iss. 5, pp. 193–197.
3. *Korneev S. A.* On the complexity of implementation of a system of two monomials by composition curcuits // Discr. Math. Appl., 2021, vol. 31, no. 2, pp. 113–125.
4. *Korneev S. A.* Ob asimptoticheskom povedenii funktsiy shennonovskogo tipa, kharakterizuyushchikh slozhnost' vychisleniya sistem monomov [On the asymptotic behavior of Shannon-type functions characterizing the computing complexity of systems of monomials]. Uchenye Zapiski Kazanskogo Universiteta. Ser. Fiziko-Matematicheskie Nauki, 2020, vol. 162, no. 3, pp. 300–311. (in Russian)
5. *Shirshov A. I.* Nekotorye algoritmicheskie problemy dlya algebr Li [Some algorithmic problems for Lie algebras]. Sibirskiy Matematicheskiy Zhurnal, 1962, vol. 3, pp. 292–296. (in Russian)
6. *Pippenger N.* On the evaluation of powers and monomials // SIAM J. Comput. 1980. vol. 9, no. 2, pp. 230–250.
7. *Knuth D. E.* The Art of Computer Programming, vol. 2. Reading, Addison-Wesley, 1969.
8. *Kochergin V. V.* On Bellman's and Knuth's problems and their generalizations. J. Math. Sci., 2018, vol. 233, no. 1, pp. 103–124.
9. *Brauer A.* On addition chains. Bull. AMS, 1939, vol. 45, pp. 736–739.
10. *Erdos P.* Remarks on number theory, III: On addition chains. Acta Arithmetica, 1960, vol. 6, pp. 77–81.
11. *Kochergin V. V. and Kochergin D. V.* Utochneniye nizhney otsenki slozhnosti vozvedeniya v stepen' [Improvement of the lower bound for the complexity of exponentiation]. Prikladnaya Diskretnaya Matematika, 2017, no. 38, pp. 119–132. (in Russian)
12. *Schönhage A.* A lower bound for the lenght of addition chains. Theor. Comput. Sci., 1975, vol. 1, pp. 1–12.
13. *Bellman R. E.* Addition chains of vectors (Advanced problems 5125). Amer. Math. Monthly, 1963, vol. 70, p. 765.
14. *Straus E. G.* Addition chains of vectors. Amer. Math. Monthly, 1964, vol. 71, pp. 806–808.

15. *Yao A. C.-C.* On the evaluation of powers. SIAM J. Computing, 1976, vol. 5, pp. 100–103.
16. *Gashkov S. B. and Kochergin V. V.* On addition chains of vectors, gate circuits, and the complexity of computations of powers. Siberian Adv. Math., 1994, vol. 4, no. 4, pp. 1–16.
17. *Kochergin V. V.* On the complexity of computations of monomials and tuples of powers. Siberian Adv. Math., 1996, vol. 6, no. 1, pp. 71–86.
18. *Kochergin V. V.* Improvement of the estimates of the computational complexity for monomials and sets of powers in Bellman's and Knuth's problems. J. Appl. Indust. Math., 2015, vol. 9, no. 1, pp. 68–82.
19. *Kochergin V. V.* O slozhnosti vychisleniya sistem odnochlenov ot dvukh peremennykh [On the complexity of computing systems of monomials in two variables]. Proc. VII Int. Conf. "Diskretnye Modeli v Teorii Upravlyayushchikh Sistem" (Pokrovskoe, March 4–6, 2006). Moscow, MAKS Press Publ., 2006, pp. 185–190. (in Russian)
20. *Sidorenko A. F.* Slozhnost' additivnykh vychisleniy semeystva tselochislennykh lineynykh form [Complexity of additive computations of a family of integer linear forms]. Zapiski Nauchnykh Seminarov LOMI, 1981, vol. 105, pp. 53–61. (in Russian)
21. *Kochergin V. V.* Asymptotics of the complexity of systems of integer linear forms for additive computations. J. Appl. Indust. Math., 2007, vol. 1, no. 3, pp. 328–342.
22. *Kochergin V. V.* O slozhnosti sovmestnogo vychisleniya trekh elementov svobodnoy abelevoy gruppy s dvumya obrazuyushchimi [On the complexity of joint computation of three elements of a free abelian group with two generators]. Diskretnyy Analiz i Issledovaniye Operatsiy. Ser. 1, 2008, vol. 15, no. 2, pp. 23–64. (in Russian)

УДК 510.52

О ГЕНЕРИЧЕСКОЙ СЛОЖНОСТИ ПРОБЛЕМЫ РАСПОЗНАВАНИЯ ГАМИЛЬТОНОВЫХ ПУТЕЙ[1]

А. Н. Рыбалов

Институт математики им. С. Л. Соболева СО РАН, г. Омск, Россия

Изучается генерическая сложность проблемы распознавания гамильтоновых путей в конечных графах. Путь в графе называется гамильтоновым, если он проходит через все вершины ровно по одному разу. Доказывается, что при условии $\mathrm{P} \neq \mathrm{NP}$ и $\mathrm{P} = \mathrm{BPP}$ для этой проблемы не существует полиномиального сильно генерического алгоритма.

Ключевые слова: *генерическая сложность, гамильтонов путь.*

DOI 10.17223/20710410/53/8

THE GENERAL COMPLEXITY OF THE PROBLEM TO RECOGNIZE HAMILTONIAN PATHS

A. N. Rybalov

Sobolev Institute of Mathematics, Omsk, Russia

E-mail: alexander.rybalov@gmail.com

Generic-case approach to algorithmic problems has been offered by A. Miasnikov, V. Kapovich, P. Schupp, and V. Shpilrain in 2003. This approach studies an algorithm behavior on typical (almost all) inputs and ignores the rest of inputs. In this paper, we study the generic complexity of the problem of recognition of Hamiltonian paths in finite graphs. A path in graph is called Hamiltonian if it passes through all vertices exactly once. We prove that under the conditions $\mathrm{P} \neq \mathrm{NP}$ and $\mathrm{P} = \mathrm{BPP}$ for this problem there is no polynomial strongly generic algorithm. A strongly generic algorithm solves a problem not on the whole set of inputs, but on a subset, the sequence of frequencies of which exponentially quickly converges to 1 with increasing size. To prove the theorem, we use the method of generic amplification, which allows to construct generically hard problems from the problems hard in the classical sense. The main component of this method is the cloning technique, which combines the inputs of a problem together into sufficiently large sets of equivalent inputs. Equivalence is understood in the sense that the problem is solved similarly for them.

Keywords: *generic complexity, Hamiltonian path.*

Введение

Гамильтонов путь в графе — это путь, проходящий через все вершины графа ровно по одному разу. Проблема распознавания гамильтоновых путей в конечных графах является классической комбинаторной проблемой, изучаемой многие десятилетия. Она содержится в знаменитом списке из двадцати одной NP-полной проблемы [1]. При

[1]Работа поддержана грантом РНФ № 19-11-00209.

условии $\mathrm{P} \neq \mathrm{NP}$ не существует полиномиального алгоритма для решения этой задачи. Здесь P — это класс алгоритмических проблем, распознаваемых за полиномиально ограниченное время на детерминированных машинах Тьюринга, а класс NP состоит из проблем, распознаваемых за полиномиальное время на недетерминированных машинах Тьюринга. Эквивалентное определение класса NP — это класс проблем, решения которых можно проверить за полиномиальное время на детерминированных машинах Тьюринга. Большинство исследователей считает, что имеет место неравенство $\mathrm{P} \neq \mathrm{NP}$, однако это до сих пор не доказано. При условии совпадения классов P и BPP (класс проблем, решаемых за полиномиальное время вероятностными алгоритмами) для проблемы распознавания гамильтоновых путей не существует и полиномиальных вероятностных алгоритмов. Имеются серьёзные доводы в пользу равенства $\mathrm{P} = \mathrm{BPP}$. В частности, доказано [2], что это равенство следует из весьма правдоподобных гипотез о вычислительной сложности некоторых трудных проблем.

В [3] предложена теория генерической вычислимости и сложности вычислений. В рамках этого 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. С точки зрения практики алгоритмы, решающие быстро проблему на генерическом множестве, так же хороши, как и быстрые алгоритмы для всех входов. Отметим, что похожий подход для изучения проблем оптимизации был предложен ранее в [4].

Данная работа посвящена изучению генерической сложности распознавания гамильтоновых путей. Доказывается, что при условии $\mathrm{P} \neq \mathrm{NP}$ и $\mathrm{P} = \mathrm{BPP}$ для этой проблемы не существует полиномиального сильно генерического алгоритма. Сильно генерический алгоритм решает проблему не на всём множестве входов, а на подмножестве, последовательность частот которого при увеличении размера экспоненциально быстро сходится к 1.

1. Предварительные сведения

Пусть I — некоторое множество входов, I_n — подмножество входов размера n. 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 назовём предел

$$\rho(S) = \overline{\lim_{n\to\infty}} \rho_n(S).$$

Верхний предел здесь нужен для того, что часто при кодировании входных данных не для каждого n существуют коды размера n. Множество S называется *пренебрежимым*, если $\rho(S) = 0$, и *сильно пренебрежимым*, если последовательность $\rho_n(S)$ экспоненциально быстро сходится к 0, т. е. существуют константы σ, $0 < \sigma < 1$, и $C > 0$, такие, что для любого n имеет место $\rho_n(S) < C\sigma^n$.

Алгоритм $\mathcal{A}$ с множеством входов I и множеством выходов $J \cup \{?\}$ $(? \notin J)$ называется (*сильно*) *генерическим*, если:

1) $\mathcal{A}$ останавливается на всех входах из I;

2) множество $\{x \in I : \mathcal{A}(x) = ?\}$ является (сильно) пренебрежимым.

Генерический алгоритм $\mathcal{A}$ вычисляет функцию $f : I \to J$, если для всех $x \in I$ выполнено

$$(\mathcal{A}(x) = y \in J) \Rightarrow (f(x) = y).$$

Ситуация $\mathcal{A}(x) = ?$ означает, что $\mathcal{A}$ не может вычислить функцию f на аргументе x. Но условие 2 гарантирует, что $\mathcal{A}$ корректно вычисляет f на почти всех входах (входах из генерического множества). Различие между генерически разрешимыми проблемами и сильно генерически разрешимыми проблемами поясняется в работе [5].

2. Гамильтоновы пути

Здесь и далее будем рассматривать неориентированные графы без петель и кратных рёбер. Пусть имеется граф G с множеством вершин $\{v_1, \ldots, v_n\}$. Путь с началом в вершине v_1 и концом в вершине v_n, проходящий через каждую вершину ровно по одному разу, называется *гамильтоновым*. *Проблема распознавания гамильтоновых путей* состоит в следующем. Задан произвольный граф G. Определить, существует ли в графе G гамильтонов путь.

Обычно в определении гамильтонова пути не накладывается условие на начало и конец. Однако, как показывает следующая лемма, это условие не делает проблему более простой, то есть проблема распознавания таких гамильтоновых путей тоже неразрешима за полиномиальное время при условии $\mathrm{P} \neq \mathrm{NP}$.

Лемма 1. Если для проблемы распознавания гамильтоновых путей с выделенными началом и концом существует полиномиальный алгоритм, то существует полиномиальный алгоритм и для проблемы распознавания гамильтоновых путей без выделенных начала и конца.

Доказательство. Допустим, что существует полиномиальный алгоритм $\mathcal{A}$ для проблемы распознавания наличия гамильтоновых путей с выделенными началом и концом. Построим полиномиальный алгоритм $\mathcal{B}$ для проблемы распознавания наличия гамильтоновых путей без выделенных начала и конца.

Алгоритм $\mathcal{B}$ работает на графе G с вершинами $v_1, \ldots, v_n$ следующим образом. Для всевозможных $i, j \leqslant n$, таких, что $i \neq j$, строится граф $G(i, j)$, который получается из графа G добавлением двух новых вершин u, w и двух новых рёбер (u, v_i) и (v_j, w). Вершина u объявляется первой, а вершина w — последней вершинами нового графа $G(i, j)$. Затем запускается алгоритм $\mathcal{A}$ на графе $G(i, j)$. Если хотя бы на одном графе $G(i, j)$ алгоритм $\mathcal{A}$ определил, что существует гамильтонов путь с началом в u и концом в w, то существует и гамильтонов путь в графе G. Если же для всех графов $G(i, j)$ таких путей нет, то нет гамильтонова пути и в графе G.

Этот алгоритм является полиномиальным, так как алгоритм $\mathcal{A}$ является полиномиальным и число графов $G(i, j)$ равно $n^2 - n$. ■

Пусть G_1 и G_2 — два графа с непересекающимися множествами вершин и v_1 — некоторая вершина графа G_1, а v_2 — некоторая вершина графа G_2. Обозначим через $G_1 \cup G_2 \cup (v_1, v_2)$ граф, являющийся объединением графов G_1 и G_2 с добавленным ребром (v_1, v_2).

Лемма 2. Пусть G_1 и G_2 — два графа с непересекающимися множествами вершин $\{v_1, \ldots, v_k\}$ и $\{w_1, \ldots, w_m\}$ соответственно. Пусть в графе G_2 существует гамильтонов путь. Тогда в графе G_1 существует гамильтонов путь тогда и только тогда, когда в графе $G_1 \cup G_2 \cup (v_k, w_1)$ существует гамильтонов путь.

Доказательство. Пусть $\pi = w_1 w'_2 \dots w'_{m-1} w_m$ — гамильтонов путь в графе G_2, $\tau = v_1 v'_2 \dots v'_{k-1} v_k$ — гамильтонов путь в графе G_1. Тогда легко видеть, что в графе $G_1 \cup G_2 \cup (v_k, w_1)$ гамильтоновым путём является путь

$$\psi = v_1 v'_2 \dots v'_{k-1} v_k w_1 w'_2 \dots w'_{m-1} w_m.$$

Обратно, пусть в графе $G_1 \cup G_2 \cup (v_k, w_1)$ есть гамильтонов путь $\tau = v_1 u_1 \dots u_{k+m-2} w_k$, где $u_i \in \{v_2, \dots, v_k, w_1, \dots, w_{k-1}\}$, $i = 1, \dots, k+m-2$. Тогда этот путь обязательно проходит через ребро (v_k, w_1), так как это единственное ребро, соединяющее графы G_1 и G_2. Поэтому этот путь имеет вид

$$\tau = v_1 v'_2 \dots v'_{k-1} v_k w_1 w'_2 \dots w'_{m-1} w_m,$$

где $v'_i \in \{v_2, \dots, v_{k-1}\}$, а $w'_i \in \{w_2, \dots, w_{m-1}\}$. Отсюда следует, что путь $\pi = v_1 v'_2 \dots v'_{k-1} v_k$ является гамильтоновым для графа G_1. ■

Лемма 3. Пусть $\mathcal{G}_n$ — множество всех графов, а $\mathcal{H}_n$ — множество графов, имеющих гамильтонов путь, с вершинами $v_1, \dots, v_n$. Тогда для достаточно больших n имеет место оценка

$$\frac{|\mathcal{H}_n|}{|\mathcal{G}_n|} > \frac{0{,}99}{n^2}.$$

Доказательство. Известно [6, 7], что почти во всех графах при достаточно большом n существует гамильтонов путь с началом в некоторой вершине v_i и концом в некоторой вершине v_j, $i \neq j$, $i, j \leqslant n$. Обозначим множество таких графов через $\mathcal{H}'_n$. Имеем

$$\frac{|\mathcal{H}'_n|}{|\mathcal{G}_n|} > 0{,}99. \tag{1}$$

Нас интересует множество графов $\mathcal{H}_n$, в которых гамильтонов путь начинается в вершине v_1 и заканчивается в вершине v_n. Обозначим через $\mathcal{H}_n(i, j)$ множество графов с гамильтоновым путём, начинающимся в вершине v_i и заканчивающимся в вершине v_j, $i \neq j$. Таким образом, $\mathcal{H}_n = \mathcal{H}_n(1, n)$. Заметим, что

$$\mathcal{H}'_n = \bigcup_{i,j \leqslant n,\ i \neq j} \mathcal{H}_n(i, j).$$

Отсюда

$$|\mathcal{H}'_n| \leqslant \sum_{i,j \leqslant n,\ i \neq j} |\mathcal{H}_n(i, j)|. \tag{2}$$

Для любых i, j, $i \neq j$, имеет место $|\mathcal{H}_n(i, j)| = |\mathcal{H}_n(1, n)|$. Действительно, перестановка на множестве индексов, которая переводит i в 1, j в n, а остальные индексы оставляет на месте, задаёт биекцию между множествами $\mathcal{H}_n(i, j)$ и $\mathcal{H}_n(1, n)$. Поэтому неравенство (2) можно переписать как

$$|\mathcal{H}'_n| \leqslant (n^2 - n)|\mathcal{H}_n(1, n)| = (n^2 - n)|\mathcal{H}_n|.$$

С учётом (1) получаем

$$\frac{|\mathcal{H}_n|}{|\mathcal{G}_n|} \geqslant \frac{0{,}99}{n^2 - n} > \frac{0{,}99}{n^2}.$$

Лемма 3 доказана. ■

3. Основной результат

Для изучения генерической сложности проблемы распознавания гамильтоновых путей будем использовать представление графов с помощью матриц смежности. Так как графы неориентированные, для кодирования графа с n вершинами достаточно верхней части такой матрицы, состоящей из $n(n-1)/2$ бит. Таким образом, будем считать, что размер графа с n вершинами равен $n(n-1)/2$.

Теорема 1. Если существует сильно генерический полиномиальный алгоритм, решающий проблему распознавания гамильтоновых путей, то существует вероятностный полиномиальный алгоритм, решающий эту проблему на всём множестве входов.

Доказательство. Допустим, что существует сильно генерический полиномиальный алгоритм $\mathcal{A}$, решающий проблему распознавания гамильтоновых путей. Построим вероятностный полиномиальный алгоритм $\mathcal{B}$, разрешающий эту проблему на всём множестве входов. Пусть имеется граф G с n вершинами $v_1, \ldots, v_n$. Он имеет размер $n(n-1)/2$. Алгоритм $\mathcal{B}$ будет работать на графе G следующим образом:

1. Генерируется случайный граф H, имеющий гамильтонов путь, с n^2-n вершинами $v_{n+1}, \ldots, v_{n^2}$. Это делается за $n(n^2-n)^2$ раундов:
   а) генерируется случайный граф H с n^2-n вершинами $v_{n+1}, \ldots, v_{n^2}$;
   б) с помощью алгоритма $\mathcal{A}$ проверяется, есть ли в графе H гамильтонов путь;
   в) если $\mathcal{A}(H) =?$, то возвращаемся на шаг а;
   г) если $\mathcal{A}(H) = 0$, то возвращаемся на шаг а;
   д) если $\mathcal{A}(H) = 1$, то генерация заканчивается.
2. Если генерация прошла успешно, переходим на следующий шаг, иначе выдаём ответ «НЕТ».
3. Запускается алгоритм $\mathcal{A}$ на графе $G' = G \cup H \cup (v_n, v_{n+1})$.
4. Если $\mathcal{A}(G') = 1$, то, по лемме 2, в графе G есть гамильтонов путь. Выдаём ответ «ДА».
5. Если $\mathcal{A}(G') = 0$, то, по лемме 2, в графе G нет гамильтонова пути. Выдаём ответ «НЕТ».
6. Если $\mathcal{A}(G') =?$, то выдаём ответ «НЕТ».

Заметим, что полиномиальный вероятностный алгоритм $\mathcal{B}$ выдаёт правильный ответ на шагах 4 и 5, а на шагах 2 и 6 может выдать неправильный ответ. Нужно доказать, что вероятность того, что ответ выдаётся на шагах 2 и 6, меньше $1/2$.

Граф H имеет n^2-n вершин, то есть его размер равен $m = (n^2-n)(n^2-n-1)/2$. Так как множество $\{G \in \mathcal{G} : \mathcal{A}(G) \neq ?\}$ сильно пренебрежимое, то существует константа $\alpha > 0$, такая, что

$$\frac{|\{H \in \mathcal{G} : \mathcal{A}(H) \neq ?\}_m|}{|\mathcal{G}_m|} < \frac{1}{2^{\alpha m}} = \frac{1}{2^{\alpha(n^2-n)(n^2-n-1)/2}}$$

для любого n.

Ответ будет выдан на шаге 2 в том случае, если за все $n(n^2-n)^2$ раундов генерации не был получен граф H с гамильтоновым путём. Эта вероятность, по лемме 3, для достаточно больших n не превосходит

$$\left(1 - \frac{0{,}99}{(n^2-n)^2} + \frac{1}{2^{\alpha(n^2-n)(n^2-n-1)/2}}\right)^{n(n^2-n)^2} < \left(1 - \frac{0{,}98}{(n^2-n)^2}\right)^{n(n^2-n)^2} < \frac{1}{\mathrm{e}^{0{,}98n}} < \frac{1}{4}.$$

Оценим вероятность выдать ответ на шаге 6. Граф $G \cup H \cup (v_n, v_{n+1})$ имеет n^2 вершин, то есть его размер равен $m = (n^4 - n^2)/2$. Вероятность того, что для случайного графа $G \cup H \cup (v_n, v_{n+1})$ имеет место $\mathcal{A}(G \cup H \cup (v_n, v_{n+1})) =?$, не больше

$$\frac{|\{G \in \mathcal{G} : \mathcal{A}(G) \neq ?\}_m|}{|\{G \cup H \cup (v_n, v_{n+1}) : H \in \mathcal{G}\}_m|} = \frac{|\{G \in \mathcal{G} : \mathcal{A}(G) \neq ?\}_m|}{|\mathcal{G}_m|} \frac{|\mathcal{G}_m|}{|\{G \cup H \cup (v_n, v_{n+1}) : H \in \mathcal{G}\}_m|}.$$

С другой стороны, так как граф H имеет $n^2 - n$ вершин, то

$$|\{G \cup H \cup (v_n, v_{n+1}) : H \in \mathcal{G}\}_m| = |\{H : H \in \mathcal{G}\}_{((n^2-n)^2-(n^2-n))/2}| = 2^{(n^4-2n^3+n)/2}.$$

Отсюда

$$\frac{|\mathcal{G}_m|}{|\{G \cup H \cup (v_n, v_{n+1}) : H \in \mathcal{G}\}_m|} = \frac{2^{(n^4-n^2)/2}}{2^{(n^4-2n^3+n)/2}} = 2^{(2n^3-n^2+n)/2}.$$

Поэтому искомая вероятность не больше

$$\frac{2^{(2n^3-n^2+n)/2}}{2^{\alpha(n^4-n^2)/2}} < \frac{1}{4}$$

при больших n.

Итого, получаем, что вероятность выдать ответ на шагах 2 и 6 не превосходит $1/4 + 1/4 = 1/2$. ■

Напомним, что класс BPP состоит из проблем, разрешимых за полиномиальное время на вероятностных машинах Тьюринга. Одной из важных гипотез в теории сложности вычислений является гипотеза о совпадении классов P и BPP. Из неё следует, что любой полиномиальный вероятностный алгоритм $\mathcal{A}$ можно эффективно дерандомизировать, то есть построить полиномиальный алгоритм $\mathcal{B}$, не использующий генератор случайных чисел и решающий ту же проблему, что и алгоритм $\mathcal{A}$. В работе [2] доказано, что равенство P = BPP следует из весьма правдоподобных гипотез о вычислительной сложности некоторых трудных проблем.

Теорема 2. Если P $\neq$ NP и P=BPP, то не существует сильно генерического полиномиального алгоритма для решения проблемы распознавания гамильтоновых путей.

Доказательство. Пусть существует сильно генерический алгоритм, решающий проблему распознавания гамильтоновых путей. Тогда по теореме 1 существует вероятностный полиномиальный алгоритм, решающий её на всём множестве входов, т. е. эта проблема лежит в классе BPP. Так как P = BPP, она лежит и в классе P, откуда, по лемме 1, P $\neq$ NP, что противоречит посылке теоремы. ■

Автор выражает благодарность рецензенту за полезные замечания и предложения по улучшению текста статьи.

ЛИТЕРАТУРА

1. *Karp R.* Reducibility among combinatorial problems // R. E. Miller and J. W. Thather (eds.). Complexity of Computer Computations. The IBM Research Symposia Ser., 1972. P. 85–103.
2. *Impagliazzo R. and Wigderson A.* P = BPP unless E has subexponential circuits: Derandomizing the XOR Lemma // Proc. 29th STOC. El Paso: ACM, 1997. P. 220–229.
3. *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.
4. *Гимади Э.Х., Глебов Н.И., Перепелица В.А.* Алгоритмы с оценками для задач дискретной оптимизации // Проблемы кибернетики. 1975. Т. 31. С. 35–42.

5. *Рыбалов А.* О генерической сложности проблемы общезначимости булевых формул // Прикладная дискретная математика. 2016. № 2(32). С. 119–126.
6. *Перепелица В.А.* О двух задачах из теории графов // Докл. АН СССР. 1970. Т. 194. № 6. С. 1269–1272.
7. *Posa L.* Hamiltonian circuits in random graphs // Discrete Math. 1976. V. 14. P. 359–364.

REFERENCES

1. *Karp R.* Reducibility among combinatorial problems. R. E. Miller and J. W. Thather (eds.), Complexity of Computer Computations. The IBM Research Symposia Ser., 1972, pp. 85–103.
2. *Impagliazzo R. and Wigderson A.* P = BPP unless E has subexponential circuits: Derandomizing the XOR Lemma. Proc. 29th STOC, El Paso, ACM, 1997, pp. 220–229.
3. *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.
4. *Gimadi E.H., Glebov N.I., and Perepelitsa V.A.* Algoritmy s ocenkami dlya zadach diskretnoi optimizacii [Algorithms with bounds for problems of discrete optimization]. Problemy Kibernetiki, 1975, vol. 31, pp. 35–42. (in Russian)
5. *Rybalov A.* O genericheskoy slozhnosti problemy obshcheznachimosti bulevykh formul [On generic complexity of the validity problem for Boolean formulas]. Prikladnaya Diskretnaya Matematika, 2016, no. 2(32), pp. 119–126. (in Russian)
6. *Perepelitsa V.A.* O dvuh zadachah iz teorii grafov [On two problems from graph theory]. Doklady AN SSSR, 1970, vol. 194. no. 6, pp. 1269–1272. (in Russian)
7. *Posa L.* Hamiltonian circuits in random graphs. Discrete Math., 1976, vol. 14, pp. 359–364.

СВЕДЕНИЯ ОБ АВТОРАХ

ВЫСОЦКАЯ Виктория Владимировна — аспирантка Московского государственного университета им. М. В. Ломоносова, специалист-исследователь НПК «Криптонит», г. Москва. E-mail: **vysotskaya.victory@gmail.com**

ГАМАЮНОВ Денис Юрьевич — кандидат физико-математических наук, доцент кафедры информационной безопасности ВМК МГУ имени М. В. Ломоносова, г. Москва. E-mail: **gamajun@seclab.cs.msu.ru**

ДЕНИСОВ Олег Викторович — кандидат физико-математических наук, доцент, ООО «Инновационные телекоммуникационные технологии», г. Москва.
E-mail: **denisovOleg@yandex.ru**

ДЕУНДЯК Владимир Михайлович — кандидат физико-математических наук, доцент кафедры алгебры и дискретной математики Института математики, механики и компьютерных наук ЮФУ, старший научный работник ФГАНУ НИИ «Спецвузавтоматика», г. Ростов-на-Дону. E-mail: **vl.deundyak@gmail.com**

ЗАГУМЕННОВ Денис Владимирович — аспирант кафедры алгебры и дискретной математики Института математики, механики и компьютерных наук ЮФУ, г. Ростов-на-Дону. E-mail: **zagumionnov.denis@yandex.ru**

ИЛЬЕВ Артём Викторович — кандидат физико-математических наук, научный сотрудник Института математики им. С. Л. Соболева СО РАН, г. Омск.
E-mail: **artyom_iljev@mail.ru**

ИЛЬЕВ Виктор Петрович — доктор физико-математических наук, профессор, профессор Омского государственного университета им. Ф. М. Достоевского, г. Омск.
E-mail: **iljev@mail.ru**

КОРНЕЕВ Сергей Александрович — аспирант кафедры дискретной математики Московского государственного университета имени М. В. Ломоносова, инженер-исследователь Института прикладной математики имени М. В. Келдыша РАН, г. Москва. E-mail: **korneev.sa.42@gmail.com**

РЫБАЛОВ Александр Николаевич — кандидат физико-математических наук, старший научный сотрудник лаборатории комбинаторных и вычислительных методов алгебры и логики Института математики им. С. Л. Соболева СО РАН, г. Омск.
E-mail: **alexander.rybalov@gmail.com**

СИГАЛОВ Даниил Алексеевич — младший научный сотрудник МГУ имени М. В. Ломоносова, г. Москва. E-mail: **asterite@seclab.cs.msu.ru**

ХАШАЕВ Артур Акрамович — младший научный сотрудник МГУ имени М. В. Ломоносова, г. Москва. E-mail: **arthur@seclab.cs.msu.ru**

ШЕВЛЯКОВ Артем Николаевич — доктор физико-математических наук, старший научный сотрудник Института математики им. С. Л. Соболева СО РАН; Омский государственный технический университет, г. Омск.
E-mail: **art.shevlyakov@gmail.com**

Журнал «Прикладная дискретная математика» входит в перечень ВАК рецензируемых научных изданий, в которых должны быть опубликованы основные результаты диссертаций на соискание учёной степени кандидата и доктора наук по специальностям 01.01.06 — «Математическая логика, алгебра и теория чисел», 01.01.09 — «Дискретная математика и математическая кибернетика», 05.13.11 — «Математическое и программное обеспечение вычислительных машин,комплексов и компьютерных сетей», 05.13.17 — «Теоретические основы информатики», 05.13.19 — «Методы и системы защиты информации, информационная безопасность», а также в перечень журналов, рекомендованных ФУМО ВО ИБ в качестве учебной литературы по специальности «Компьютерная безопасность».

Журнал индексируется в базах данных Web of Science (Emerging Sources Citation Index (ESCI) и Russian Science Citation Index (RSCI)), Scopus, MathSciNet и Zentralblatt MATH.

Журнал «Прикладная дискретная математика» распространяется по подписке; его подписной индекс 38696 в объединённом каталоге «Пресса России». Полнотекстовые электронные версии вышедших номеров журнала доступны на его сайте journals.tsu.ru/pdm и на Общероссийском математическом портале www.mathnet.ru. На сайте журнала можно найти также правила подготовки рукописей статей для публикации в журнале.

Тематика публикаций журнала:

- *Теоретические основы прикладной дискретной математики*
- *Математические методы криптографии*
- *Математические методы стеганографии*
- *Математические основы компьютерной безопасности*
- *Математические основы надёжности вычислительных и управляющих систем*
- *Прикладная теория кодирования*
- *Прикладная теория автоматов*
- *Прикладная теория графов*
- *Логическое проектирование дискретных автоматов*
- *Математические основы информатики и программирования*
- *Вычислительные методы в дискретной математике*
- *Дискретные модели реальных процессов*
- *Математические основы интеллектуальных систем*
- *Исторические очерки по дискретной математике и её приложениям*