МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

ПРИКЛАДНАЯ ДИСКРЕТНАЯ МАТЕМАТИКА

Научный журнал

2010 №2(8)

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

ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

РЕДАКЦИОННАЯ КОЛЛЕГИЯ ЖУРНАЛА
«ПРИКЛАДНАЯ ДИСКРЕТНАЯ МАТЕМАТИКА»

Агибалов Г. П., д-р техн. наук, проф. (председатель); Девянин П. Н., д-р техн. наук, проф. (зам. председателя); Парватов Н. Г., канд. физ.-мат. наук, доц. (зам. председателя); Черемушкин А. В., д-р физ.-мат. наук, чл.-корр. Академии криптографии (зам. председателя); Панкратова И. А., канд. физ.-мат. наук, доц. (отв. секретарь); Алексеев В. Б., д-р физ.-мат. наук, проф.; Бандман О. Л., д-р техн. наук, проф.; Евдокимов А. А., канд. физ.-мат. наук, проф.; Евтушенко Н. В., д-р техн. наук, проф.; Закревский А. Д., д-р техн. наук, проф., чл.-корр. НАН Беларуси; Костюк Ю. Л., д-р техн. наук, проф.; Логачев О. А., канд. физ.-мат. наук, доц.; Матросова А. Ю., д-р техн. наук, проф.; Микони С. В., д-р техн. наук, проф.; Салий В. Н., канд. физ.-мат. наук, проф.; Сафонов К. В., д-р физ.-мат. наук, проф.; Фомичев В. М., д-р физ.-мат. наук, проф.; Чеботарев А. Н., д-р техн. наук, проф.; Шоломов Л. А., д-р физ.-мат. наук, проф.

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

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

Периодичность выхода журнала: 4 номера в год.

ООО «Издательство научно-технической литературы»
634050, Томск, пл. Ново-Соборная, 1, тел. (3822) 533-335
Редактор *Н. И. Шидловская*
Верстка *Д. А. Стефанцова*

Изд. лиц. ИД. №04000 от 12.02.2001. Подписано к печати 10.06.2010.
Формат $60 \times 84\frac{1}{8}$. Бумага офсетная. Печать офсетная. Гарнитура «Таймс».
Усл. п. л. 13,4. Уч.-изд. л. 15. Тираж 300 экз. Заказ №15.

Отпечатано в типографии «М-Принт», г. Томск, ул. Пролетарская, 38/1

СОДЕРЖАНИЕ

ТЕОРЕТИЧЕСКИЕ ОСНОВЫ ПРИКЛАДНОЙ ДИСКРЕТНОЙ МАТЕМАТИКИ

МАТЕМАТИЧЕСКИЕ МЕТОДЫ КРИПТОГРАФИИ

МАТЕМАТИЧЕСКИЕ МЕТОДЫ СТЕГАНОГРАФИИ

МАТЕМАТИЧЕСКИЕ ОСНОВЫ НАДЁЖНОСТИ ВЫЧИСЛИТЕЛЬНЫХ И УПРАВЛЯЮЩИХ СИСТЕМ

МАТЕМАТИЧЕСКИЕ ОСНОВЫ ИНФОРМАТИКИ И ПРОГРАММИРОВАНИЯ

ПРИКЛАДНАЯ ТЕОРИЯ АВТОМАТОВ

CONTENTS

ТЕОРЕТИЧЕСКИЕ ОСНОВЫ ПРИКЛАДНОЙ ДИСКРЕТНОЙ МАТЕМАТИКИ

DOI 10.17223/20710410/8/1

УДК 519.95

НЕОБХОДИМЫЕ И ДОСТАТОЧНЫЕ УСЛОВИЯ ТРИВИАЛЬНОСТИ ЛИНЕЙНОЙ СТРУКТУРЫ МОНОМИАЛЬНОГО ОТОБРАЖЕНИЯ НАД ПОЛЕМ ИЗ 2^{2^t} ЭЛЕМЕНТОВ

А. Н. Алексейчук, Р. В. Проскуровский

Институт специальной связи и защиты информации Национального технического университета Украины «Киевский политехнический институт», г. Киев, Украина

E-mail: alex-crypto@mail.ru, roman-crypto@mail.ru

В терминах двоичного представления натурального числа d получены необходимые и достаточные условия, при которых все ненулевые линейные комбинации координатных функций отображения $x \mapsto x^d$, $x \in \mathbf{GF}(2^{2^t})$, не имеют линейных трансляторов.

Ключевые слова: *линейная структура дискретного отображения, конечное поле, мономиальное отображение.*

Одним из известных требований к дискретным отображениям, используемым в современных симметричных криптосистемах, является условие отсутствия линейных трансляторов. Напомним (см., например, [1]), что ненулевой вектор $a \in \{0,1\}^n$ называется линейным транслятором отображения $\varphi : \{0,1\}^n \to \{0,1\}^m$, если существует элемент $c \in \{0,1\}^m$, такой, что равенство $\varphi(x \oplus a) \oplus \varphi(x) = c$ выполняется для всех $x \in \{0,1\}^n$. Как правило, наличие линейных трансляторов (или, как говорят, нетривиальность линейной структуры) отображения φ свидетельствует о его криптографических слабостях.

Исследованию строения множества линейных трансляторов булевых функций и подстановок на множестве $\{0,1\}^n$ посвящены работы [2–5] и др. В [6] предложено обобщение понятия линейного транслятора для отображений векторных пространств над конечными полями и исследованы распределения ряда числовых характеристик множества линейных трансляторов случайных равновероятных отображений указанного вида. В [7] описаны биективные преобразования конечных модулей над конечным ассоциативным кольцом с единицей, имеющие заданное непустое множество линейных трансляторов.

С криптографической точки зрения более естественным (и более жестким) является требование, состоящее в отсутствии линейных трансляторов не только у данного отображения $\varphi : \{0,1\}^n \to \{0,1\}^m$, но и у всех ненулевых линейных комбинаций его координатных функций. Формализация этого условия (применительно к более широкому классу отображений одной конечной абелевой группы в другую) приводит к понятию тривиальности линейной структуры дискретного отображения, введенному в работе [8]. Отметим, что это, более общее по сравнению с [6, 7], понятие возникает естественным образом при обосновании достаточных условий стойкости блочных шифров относительно алгебраических атак, основанных на гомоморфизмах (см. [8, 9]).

Определение [8]. Пусть G_1 и G_2 — конечные абелевы группы, $\varphi : G_1 \to G_2$ — произвольное отображение. Будем говорить, что φ *имеет тривиальную линейную структуру*, если не существует элемента $a \in G_1 \setminus \{0\}$ и комплексного характера $\psi \neq 1$ группы G_2, таких, что функция $\psi(\varphi(x+a) - \varphi(x))$, $x \in G_1$, является константой.

В [10] получены достаточные условия тривиальности линейной структуры преобразований аддитивной группы конечного поля и показано, что при $n = 2^t$, $t \geqslant 2$, тривиальную линейную структуру имеет отображение $\varphi(x) = x^{2^n-2}$, $x \in \mathbf{GF}(2^n)$, используемое при построении узлов замены современных блочных шифров [11, 12].

Полное описание полиномов над конечными полями, задающих отображения с тривиальной линейной структурой, представляет собой непростую задачу. В настоящей работе исследуются мономиальные отображения над полем порядка 2^{2^t}.

Введем следующие обозначения:
$F = \mathbf{GF}(2^n)$, $n = 2^t$, $t \geqslant 2$;
$F^* = F \setminus \{0\}$;
$\mathrm{Tr}(x) = x + x^2 + ... + x^{2^{n-1}}$ — абсолютный след элемента $x \in F$;
$\overline{i,j} = \{l \in \mathbb{Z} : i \leqslant l \leqslant j\}$, $i, j \in \mathbb{Z}$.

Пусть $l = \sum\limits_{i=0}^{n-1} 2^i l_i$ — двоичное представление числа $l \in \overline{0,\, 2^n - 1}$. Обозначим $I_l = \{i \in \overline{0, n-1} : l_i = 1\}$. Для любых $M \subseteq \overline{0, n-1}$, $i \in \overline{0, n-1}$, $l \in \overline{0,\, 2^n - 1}$ положим

$$M + i = \{(j + i) (\mathrm{mod}\, n) : j \in M\},\ s(l, i) = 2^i l (\mathrm{mod}(2^n - 1)).$$

Непосредственно из данных определений вытекает следующее утверждение.

Лемма 1. Для любых $i \in \overline{0, n-1}$, $l \in \overline{0,\, 2^n - 1}$ справедливо равенство

$$I_{s(l,i)} = I_l + i.$$

Сформулируем два вспомогательных утверждения, первое из которых является следствием условия $n = 2^t$ и теоремы 2.39, приведенной в [13], а второе следует из данного выше определения и известных свойств конечных полей (см. теоремы 5.7 и 2.21 в [13]).

Лемма 2. Для любого $x \in F$ элементы $x, x^2, ..., x^{2^{n-1}}$ образуют базис поля F над полем $\mathbf{GF}(2)$ тогда и только тогда, когда $\mathrm{Tr}(x) = 1$.

Лемма 3. Отображение $\varphi : F \to F$ тогда и только тогда имеет тривиальную линейную структуру, когда для любых $a, b \in F^*$ выполняется соотношение $\mathrm{Tr}(b\varphi(x+a) + b\varphi(x)) \neq \mathrm{const}$. При этом отображения $x \mapsto \varphi(x)$ и $x \mapsto \varphi(x^2)$, $x \in F$, имеют тривиальную или нетривиальную линейную структуру одновременно.

Сформулируем и докажем теорему, содержащую необходимые и достаточные условия тривиальности линейной структуры мономиального отображения

$$\varphi(x) = x^d,\ x \in F. \tag{1}$$

Отметим, что на основании второго утверждения леммы 3 можно ограничиться случаем нечетных значений d.

Теорема. Пусть $d \in \overline{3, 2^n - 2}$, $d \equiv 1 (\mathrm{mod}\, 2)$. Тогда отображение (1) имеет тривиальную линейную структуру в том и только том случае, когда существует множество $M \subseteq \overline{0, n-1}$, $M \notin \{\varnothing, I_d\}$, такое, что

$$\left|\{i \in \overline{0, n-1} : M + i \subseteq I_d\}\right| \equiv 1 (\mathrm{mod}\, 2). \tag{2}$$

Доказательство. Пусть $a, b \in F^*$; тогда для любого $x \in F$

$$b(\varphi(x+a)+\varphi(x)) = b((x+a)^d + x^d) = ba^d((z+1)^d + z^d),$$

где $z = xa^{-1}$. Отсюда на основании лемм 2 и 3 вытекает, что φ имеет тривиальную линейную структуру в том и только в том случае, когда

$$\exists\, z \in F^* \left(\mathrm{Tr}((z+1)^d + z^d) = 1\right). \tag{3}$$

Действительно, согласно лемме 2, при выполнении условия (3) множество $\{(z+1)^d + z^d : z \in F\}$ содержит некоторый базис поля F над полем $\mathbf{GF}(2)$, откуда следует, что равенство $\mathrm{Tr}(b(\varphi(x+a)+\varphi(x))) = \mathrm{const}$, $x \in F$, возможно лишь при условии $ba^d = 0$. Наоборот, если $\mathrm{Tr}((z+1)^d + z^d) = 0$ для любого $z \in F^*$, то в силу равенства $\mathrm{Tr}(1) = n \cdot 1 = 0$ справедливо соотношение $\mathrm{Tr}(\varphi(x+1)+\varphi(x))) = 0$, $x \in F$, означающее, что отображение φ имеет нетривиальную линейную структуру.

Используя определение функции след и равенство $\mathrm{Tr}(1) = 0$, получим

$$\mathrm{Tr}((z+1)^d + z^d) = \mathrm{Tr}\left(\sum_{j=1}^{d-1} z^j \binom{d}{j}\right) = \sum_{j=1}^{d-1}\sum_{i=0}^{n-1} \binom{d}{j} z^{j\cdot 2^i},$$

откуда на основании тождества $z^{2^n-1} = 1$, $z \in F^*$, вытекает следующее равенство:

$$\mathrm{Tr}((z+1)^d + z^d) = \sum_{i=0}^{2^n-2} A_l z^l,\ z \in F^*, \tag{4}$$

где

$$A_l = \sum_{\substack{(i,\,j):\\ i\in\overline{0,\,n-1},j\in\overline{1,\,d-1},\\ j\cdot 2^i \equiv l \bmod (2^n-1)}} \binom{d}{j},\ l \in \overline{0,\,2^n-2}. \tag{5}$$

Поскольку полином от z в правой части равенства (4) имеет степень, меньшую 2^n, то условие (3) равносильно следующему условию:

$$\exists\, l \in \overline{0,\,2^n-2}\ \ (A_l \neq 0). \tag{6}$$

Преобразуем выражение (5). Заметим, что сравнение $j2^i \equiv l \bmod(2^n-1)$ равносильно сравнению $j \equiv 2^{n-i}l \bmod(2^n-1)$; следовательно,

$$A_l = \sum_{\substack{i\in\overline{0,\,n-1}:\\ s(l,\,i)\in\overline{1,\,d-1}}} \binom{d}{s(l,i)},\ l \in \overline{0,\,2^n-2}. \tag{7}$$

Далее, по теореме Лукаса [14] $\binom{d}{s(l,\,i)} \equiv 1 (\mathrm{mod}\, 2)$ тогда и только тогда, когда множество единичных разрядов в двоичном представлении числа $s(l,\,i)$ содержится во множестве I_d, то есть $I_{s(l,\,i)} \subseteq I_d$. При этом, согласно лемме 1, $I_{s(l,\,i)} = I_l + i$, $i \in \overline{0,\,n-1}$, $l \in \overline{0,\,2^n-2}$. Отсюда на основании формулы (7) вытекает, что $A_l = 0$, если $l = 0$ или $|I_l| \geqslant |I_d|$. Таким образом, условие (6) равносильно следующему условию:

$$\exists\, l \in \overline{0,\,2^n-2}\ (|I_l| < |I_d|\ \&\ A_l \neq 0). \tag{8}$$

Заметим, наконец, что для любого $l \in \overline{1,\,2^n-2}$, такого, что $|I_l| < |I_d|$, справедлива импликация

$$\forall\, i \in \overline{0,\,n-1}\ \left[\binom{d}{s(l,\,i)} \equiv 1 (\mathrm{mod}\, 2)\ \Rightarrow\ s(l,\,i) \in \overline{1,\,d-1}\right].$$

Следовательно, для указанных l

$$A_l = \sum_{i=0}^{n-1} \binom{d}{s(l,i)} \equiv \left|\{i \in \overline{0,\, n-1} : I_l + i \subseteq I_d\}\right| (\mathrm{mod}\, 2). \qquad (9)$$

Теперь нетрудно завершить доказательство теоремы. Если отображение (1) имеет тривиальную линейную структуру, то на основании соотношений (8) и (9) множество $M = I_l \notin \{\oslash, I_d\}$ удовлетворяет условию (2).

Наоборот, пусть множество $M \subseteq \overline{0, n-1}$, $M \notin \{\oslash, I_d\}$, удовлетворяет условию (2). Тогда $M = I_l$ для некоторого $l \in \overline{1,\, 2^n - 1} \setminus \{d\}$ и существует $i \in \overline{0,\, n-1}$, такое, что $I_l + i \subseteq I_d$. Следовательно, $|I_l| < |I_d|$, в частности, $l \neq 2^n - 1$. Отсюда на основании соотношений (2) и (9) вытекает справедливость условия (8). Таким образом, отображение (1) имеет тривиальную линейную структуру, что и требовалось доказать. ■

Отметим, что полученный критерий тривиальности линейной структуры отображений вида (1) допускает простую практическую проверку.

Следствие 1. Если в условиях теоремы двоичное представление числа d содержит нечетное число единиц, то отображение (1) имеет тривиальную линейную структуру.

Для доказательства достаточно рассмотреть множество $M = \{0\}$.

Следствие 2. Пусть в условиях теоремы $d = 4^k - 2^k + 1$, $k \equiv 1 (\mathrm{mod}\, 2)$, $k \geqslant 3$. Тогда отображение (1) (известное как функция Касами; см. [1], с. 340) имеет тривиальную линейную структуру.

Для доказательства достаточно положить $M = \{0, 1, 2\}$ и воспользоваться соотношением $\left|\{i \in \overline{0,\, n-1} : M + i \subseteq I_d\}\right| = k - 2 \equiv 1 (\mathrm{mod}\, 2)$.

ЛИТЕРАТУРА

1. *Логачев О. А., Сальников А. А., Ященко В. В.* Булевы функции в теории кодирования и криптологии. М.: МЦНМО, 2004. 470 с.
2. *Evertse J. H.* Linear structures in blockciphers // Advances in Cryptology — EUROCRYPT'87. Springer Verlag, 1988. P. 249–266.
3. *Lai X.* Additive and linear structures of cryptographic functions // Fast Software Encryption — FSE'94. Springer Verlag, 1995. P. 75–86.
4. *Ященко В. В.* О критерии распространения для булевых функций и о бент-функциях // Проблемы передачи информации. 1997. Т. 33. № 1. С. 75–86.
5. *Dawson E., Wu Ch. K.* On the linear structures of symmetric functions // Australian J. of Combinatorics. Springer Verlag, 1997. V. 16. P. 239–243.
6. *Сачков В. Н.* Трансляторы и трансляции дискретных функций // Труды по дискретной математике. М.: Гелиос АРВ, 2006. Т. 9. С. 253–268.
7. *Пудовкина М. А.* Линейные структуры групп подстановок над конечным модулем // Прикладная дискретная математика. 2008. № 1. С. 25–28.
8. *Алексейчук А. Н.* Достаточные условия стойкости рандомизированных блочных систем шифрования относительно метода криптоанализа на основе коммутативных диаграмм // Реєстрація, зберігання і обробка даних. 2007. Т. 9. № 2. С. 61–68.
9. *Алексейчук А. Н.* Критерий примитивности группы подстановок, порожденной раундовыми преобразованиями Rijndael-подобного блочного шифра // Реєстрація, зберігання і обробка даних. 2004. Т. 6. № 2. С. 11–18.

10. *Алексейчук А. Н., Скрынник Е. В.* Классы отображений с тривиальной линейной структурой над конечным полем // Реєстрація, зберігання і обробка даних. 2008. Т. 10. № 3. С. 80–88.
11. *Daemen J., Rijmen D.* AES proposal: Rijndael // `htpp://csrc.nist.gov/encription/aes/rijndael/Rijndael.pdf`.
12. *Aoki A., Ichikawa T., Kanda M., et al.* Camellia: a 128-bit block cipher suitable for multiple platforms — Design and Analysis // Selected Areas in Cryptography. Springer Verlag, 2001. P. 39–56.
13. *Лидл Р., Нидеррайтер Г.* Конечные поля. М.: Мир, 1988. Т. 1, 2. 818 с.
14. *Берлекэмп Э.* Алгебраическая теория кодирования. М.: Мир, 1971. 477 с.

2010 Теоретические основы прикладной дискретной математики №2(8)

DOI 10.17223/20710410/8/2

УДК 519.7

СООТВЕТСТВИЕ ГАЛУА ДЛЯ ЗАМКНУТЫХ КЛАССОВ ДИСКРЕТНЫХ ФУНКЦИЙ[1]

Н. Г. Парватов

Томский государственный университет, г. Томск, Россия

E-mail: parvatov@mail.tsu.ru

Формулируется теория Галуа для S-замкнутых классов дискретных функций, совпадающих в частных случаях с замкнутыми суперпозицией классами функций многозначной логики, с клонами, с наследственными системами дискретных функций, а также с классами функций, вычисляемых схемами из переключательных элементов.

Ключевые слова: *замкнутый класс, клон, предикат, сохранение предиката, соответствие Галуа, дискретная функция.*

Введение

Один из классических объектов изучения дискретной математики — системы дискретных функций, рассматриваемые с замыканием относительно некоторых операций суперпозиции, таких, как операции перестановки и отождествления переменных, введения и удаления фиктивных переменных, операция композиции и другие. Подобными системами являются, например, замкнутые классы функций многозначной логики, клоны, наследственные системы, а также системы переключательных функций, вычисляемых схемами из переключательных элементов.

Следует отметить, что при изучении замыкания, заданного в некотором множестве P, часто важной оказывается возможность рассматривать это замыкание как замыкание Галуа, определяемое некоторым соответствием Галуа [1]. Последнее задаётся обычно посредством отношения $\alpha \subseteq P \times Q$ между элементами множества P и элементами некоторого множества Q. Тогда замкнутыми классами элементов множества P являются его галуа-замкнутые классы

$$\alpha(Y) = \bigcap_{y \in Y} \alpha(y) \text{ при } Y \subseteq Q,$$

где $\alpha(y) = \{x : (x, y) \in \alpha\}$ для любого y из Q, находящиеся во взаимно-однозначном соответствии с галуа-замкнутыми классами множества Q, то есть с классами

$$(X)\alpha = \bigcap_{x \in X} (x)\alpha \text{ при } X \subseteq P,$$

где $(x)\alpha = \{x : (x, y) \in \alpha\}$ для любого x из P. В этом случае изучение замыкания в множестве P сводится к изучению замыкания Галуа в множестве Q и наоборот.

Так, при изучении клонов существенную роль играет соответствие Галуа, определяемое отношением сохранения функцией $f : E^n \to E$ предиката $p : E^m \to \{\text{И, Л}\}$.

[1]Исследование выполнено при финансовой поддержке ФЦП «Научные и научно-педагогические кадры инновационной России» на 2009–2013 годы, гос. контракт № П1010.

В [2, 3] показано, что галуа-замкнутыми классами функций в этом случае являются всевозможные клоны на множестве E, там же описаны галуа-замкнутые классы предикатов. Впоследствии эти результаты неоднократно обобщались.

При изучении наследственных систем функций $f : E^n \to C$ (то есть систем, замкнутых операциями отождествления и перестановки переменных, введения фиктивных переменных) большое значение имеет соответствие Галуа, определяемое отношением сохранения такими функциями пар предикатов $p : E^m \to \{\text{И, Л}\}$ и $q : C^m \to \{\text{И, Л}\}$. В [4] показано, что галуа-замкнутыми классами функций в этом случае являются наследственные системы, там же описаны галуа-замкнутые классы пар предикатов.

Отметим также работы [5, 6]. В первой из них теория Галуа сформулирована для клонов многоосновных алгебр, состоящих из функций, вычисляемых термами с переменными нескольких сортов, а во второй — для систем функций $f : E^n \to C$, замкнутых операциями перестановки переменных и введения фиктивной переменной.

В настоящей работе рассматриваются функции $f : E^n \to C$, где E и C — фиксированные конечные множества, а n принимает произвольные натуральные значения. В множестве $P_{C,E}$ всех таких функций определяется операция S-замыкания. В частных случаях S-замкнутыми классами являются замкнутые классы функций многозначной логики, клоны, наследственные системы, а также классы переключательных функций, вычисляемых схемами из переключательных элементов. Теорию Галуа, сформулированную в [4] для наследственных систем, удаётся перенести на S-замкнутые классы. Таким образом удаётся построить обобщённую теорию Галуа, в частных случаях включающую известные результаты для замкнутых классов функций многозначной логики, клонов и наследственных систем, а также ранее не известные результаты о классах функций, вычисляемых переключательными схемами.

Статья организована следующим образом. В п. 1 вводится понятие S-замыкания. В п. 2 рассматривается отношение сохранения функцией пары предикатов. Это отношение рассматривается как отношение между функциями, принадлежащими множеству $P_{C,E}$, и парами предикатов, выбираемых из некоторого множества $\Pi^S_{C,E}$. Множество $\Pi^S_{C,E}$ удаётся определить таким образом, что галуа-замкнутыми классами функций оказываются точно все S-замкнутые классы функций из $P_{C,E}$. Это устанавливается теоремой 1. В п. 3 доказывается теорема 2, характеризующая галуа-замкнутые классы пар предикатов как классы, замкнутые некоторым набором операций.

Основная задача статьи — сообщить о возможности обобщения теории Галуа, единого для замкнутых классов функций многозначной логики, клонов, наследственных систем и систем переключательных функций. Само обобщение не является сложной задачей и, по всей видимости, может быть выполнено формально. Тем не менее автору удалось найти оригинальное доказательство теоремы 2, не повторяющее полностью схем рассуждений, использованных в аналогичной ситуации в [2] или в [3, 4].

1. Замкнутые классы дискретных функций

Будем обозначать через P_E множество всех функций $f : E^n \to E$, где E — фиксированное конечное множество и n — произвольное натуральное число. Замкнутые операциями суперпозиции из [7, 8] множества функций из P_E называются *замкнутыми классами*, а замкнутые классы, включающие класс S_E всех селекторных (тождественно равных некоторому своему аргументу) функций, называются *клонами*.

Будем обозначать через $P_{C,E}$ множество всех функций $f : E^n \to C$, где E, C — фиксированные конечные множества и n — произвольное натуральное число. Для заданных конечных множеств E, C и D, а также множеств X и Y функций из соответ-

ствующих классов $P_{D,C}$ и $P_{C,E}$ через XY будет обозначаться множество всех функций $f(g_1(x),\ldots,g_n(x))$, где f — n-местная функция из X; $g_1,\ldots,g_n$ — m-местные функции из Y; через x обозначен набор переменных $x_1,\ldots,x_m$, а числа n и m принимают произвольные натуральные значения.

Далее будем считать заданными конечные множества E и C и набор $S=(L,O,R,U)$, состоящий из клонов L, R и множеств O, U функций из соответствующих классов P_C, P_E и $P_{C,E}$, $P_{E,C}$. Множество M функций из $P_{C,E}$ назовём *S-замкнутым классом*, если выполняются включения

$$O\subseteq M,\quad LMR\subseteq M,\quad MUM\subseteq M.$$

Частными случаями S-замкнутых классов являются замкнутые классы и клоны функций из P_E (возникающие при выполнении равенств $E=C$ и $L=R=U=S_E$ и соответствующего равенства $O=\varnothing$ или $O=S_E$), а также изучавшиеся в [4, 9] (L,R)-замкнутые наследственные классы (возникающие при пустых множествах U и O). Это делает S-замкнутые классы привлекательными для изучения.

Помимо этого, S-замкнутые классы имеют приложения в теории управляющих систем как классы функций, вычисляемых схемами из переключательных элементов с неограниченным числом каскадов. При этом элементы множеств E и C интерпретируются соответственно как состояния узлов и как проводимости между узлами, возможные в схемах. Функции клонов L и R интерпретируются как допустимые используемой технологией синтеза операции над проводимостями и состояниями. В множестве O находятся функции переключательных элементов, используемых при синтезе наравне с базисными переключательными элементами. Наконец, множество U состоит из функций узлового соединения, каждая из которых определяет состояние произвольного узла схемы набором проводимостей от него до полюсов питания. Очень часто в схемах с r-полюсным источником питания состояние любого узла однозначно определяется набором проводимостей от него до полюсов питания; тогда множество E совпадает с C^r, а множество U состоит из единственной тождественной функции $u^{(r)}:C^r\to E$. Естественной также является ситуация, когда в дополнение к сказанному выше множество O содержит все селекторы $s^i:E\to C$, такие, что $s^i(x)=x^i$; при этом переменная $x=(x^1,\ldots,x^r)$ принимает значения в множестве $E=C^r$, её i-я компонента x^i принимает значения в множестве C и $1\leqslant i\leqslant r$. Следует отметить, однако, что в этом последнем случае при выполнении равенств $E=C^r$ и $L=S_E$, $O=\{s^1,\ldots,s^r\}$, $R=S_C$, $U=\{u^{(r)}\}$ изучение S-замкнутых классов сводится к изучению клонов функций из P_E, чего нельзя гарантировать в других случаях.

2. Сохраняемые пары предикатов

Обозначим через Π_E множество всех m-местных предикатов $p:E^m\to\{\text{И, Л}\}$ при всевозможных натуральных m. Пару (p_1,p_2) m-местных предикатов из соответствующих множеств Π_C и Π_E будем называть m-местным 2-предикатом на паре множеств C и E. Множество всех таких 2-предикатов при всевозможных натуральных m обозначим через $\Pi_{C,E}$. Говорят, что n-местная *функция f из $P_{C,E}$ сохраняет* m-арный *2-предикат* (p_1,p_2), если для любых наборов $X_1,\ldots,X_n$, удовлетворяющих предикату p_2, набор $Y=f^{[m]}(X_1,\ldots,X_n)$ удовлетворяет предикату p_1. При этом набор Y состоит из значений функции f, вычисленных последовательно от строк матрицы $(X_1^T,\ldots,X_n^T)$, где верхний индекс ^{T} означает транспонирование. Если при совпадающих множествах E и C предикаты p_1 и p_2 также совпадают, то говорят о *сохранении функцией f предиката p_1*. В этой ситуации предикат p удобно отождествить с 2-предикатом (p,p).

Обозначим через $\mathrm{pol}_{C,E}(Y)$ множество всех функций из $P_{C,E}$, сохраняющих все 2-предикаты множества $Y \subseteq \Pi_{C,E}$. Через $\mathrm{inv}_{C,E}(X)$ будем обозначать множество всех 2-предикатов из $\Pi_{C,E}$, сохраняемых всеми функциями из множества $X \subseteq P_{C,E}$. Положим также

$$\Pi^S_{C,E} = (\mathrm{inv}_C(L) \times \mathrm{inv}_E(R)) \cap \mathrm{inv}_{C,E}(O) \cap \mathrm{inv}'_{E,C}(U);$$

$$\mathrm{inv}^S_{C,E}(Y) = \Pi^S_{C,E} \cap \mathrm{inv}_{C,E}(Y),$$

где $\mathrm{inv}'_{E,C}(U)$ — множество 2-предикатов (p_2, p_1), полученных перестановкой координат у 2-предикатов (p_1, p_2) из класса $\mathrm{inv}_{E,C}(U)$.

Предметом данного исследования является соответствие Галуа между системами подмножеств в $P_{C,E}$ и $\Pi^S_{C,E}$, определяемое отношением сохранения функцией из $P_{C,E}$ 2-предиката из $\Pi^S_{C,E}$. При пустых множествах O и U это отношение изучалось в [2, 3]. А при выполнении равенств $C = E$ и $L = O = R = U = S_E$ это отношение по сути совпадает с изучавшимся в [2, 3] отношением сохранения функцией из P_E предиката из Π_E (и совпадает с ним формально, если всякий 2-предикат (p, p) из $\Pi^S_{C,E}$ отождествить с предикатом p из Π_E). В обоих упомянутых случаях галуа-замкнутыми классами функций являются S-замкнутые классы. Это свойство выполняется и в общем случае, так как имеет место

Теорема 1. Для любого множества N 2-предикатов из $\Pi^S_{C,E}$ множество $\mathrm{pol}_{C,E}(N)$ является S-замкнутым классом функций из $P_{C,E}$. Для любого S-замкнутого класса M функций из $P_{C,E}$ имеет место равенство $M = \mathrm{pol}_{C,E}(\mathrm{inv}^S_{C,E}(M))$.

Доказательство. Первое предложение теоремы проверяется непосредственно. Во втором — включение $M \subseteq \mathrm{pol}_{C,E}(\mathrm{inv}^S_{C,E}(M))$ очевидно. Докажем обратное включение. Для этого рассмотрим произвольную n-местную функцию f из множества $\mathrm{pol}_{C,E}(\mathrm{inv}^S_{C,E}(M))$. Пусть $m = |E|^n$ и наборы $X_1, \ldots, X_n$ выбраны в множестве E^m так, что множество строк матрицы $(X_1^T, \ldots, X_n^T)$ совпадает с E^n. Рассмотрим m-арный 2-предикат (p_1, p_2) из $\Pi_{C,E}$, такой, что предикату p_2 удовлетворяют всевозможные наборы $g^{[m]}(X_1, \ldots, X_n)$, где n-местная функция g принадлежит множеству $R(UM \cup S_E)$, а предикату p_1 удовлетворяют всевозможные наборы $g^{[m]}(X_1, \ldots, X_n)$, где n-местная функция g принадлежит множеству M. Непосредственно проверяется, что этот 2-предикат принадлежит множеству $\mathrm{inv}^S_{C,E}(M)$ и, следовательно, сохраняется функцией f. Тогда, поскольку наборы $X_1, \ldots, X_n$ удовлетворяют предикату p_2, набор $f^{[m]}(X_1, \ldots, X_n)$ удовлетворяет предикату p_1. Это означает, что для некоторой функции g из M выполняется равенство $f^{[m]}(X_1, \ldots, X_n) = g^{[m]}(X_1, \ldots, X_n)$. Так как в строках матрицы $(X_1^T, \ldots, X_n^T)$ содержатся все наборы из E^n, то функции f и g совпадают. Таким образом, функция f принадлежит классу M. Теорема доказана. ■

Итак, теорема 1 характеризует классы функций из $P_{C,E}$, сохраняющих множества 2-предикатов из $\Pi^S_{C,E}$. Классы 2-предикатов, сохраняемых множествами функций, будут охарактеризованы в п. 3.

3. Замкнутые классы пар предикатов

Введём в рассмотрение необходимые далее операции над 2-предикатами из $\Pi_{C,E}$. *Конъюнкцией* n- и m-местных 2-предикатов (p_1, p_2) и (q_1, q_2) из $\Pi_{C,E}$ будем называть $(n+m)$-местный 2-предикат (r_1, r_2) из $\Pi_{C,E}$, где

$$r_i(x_1, \ldots, x_{n+m}) \equiv p_i(x_1, \ldots, x_n) \wedge q_i(x_{n+1}, \ldots, x_{n+m})$$

для $i = 1, 2$. *Проекцией* n-местного 2-предиката (p_1, p_2) по j-й координате будем называть 2-предикат (p'_1, p'_2), где

$$p'_i(x_1, \ldots, x_{j-1}, x_{j+1}, \ldots, x_n) \equiv \exists x_j \ (p_i(x_1, \ldots, x_n))$$

для $i = 1, 2$. Для n-местного предиката p из Π_E и набора I чисел $i_1, \ldots, i_n$ из множества $\{1, \ldots, m\}$ положим

$$p_I^{(m)}(x_1, \ldots, x_m) \equiv p(x_{i_1}, \ldots, x_{i_n}).$$

Тогда для n-местного 2-предиката (p_1, p_2) будем говорить, что m-местный 2-предикат $(p_{1I}^{(m)}, p_{2I}^{(m)})$ *получен* из 2-предиката (p_1, p_2) *подстановкой переменных*. Наконец, 2-предикат (p_1, p_2) будем называть *2-диагональю*, если предикаты p_1, p_2 — тождественно истинные или тождественно ложные, или оба выражаются одной и той же формулой вида $x_i = x_j \wedge \ldots \wedge x_l = x_s$, интерпретируемой в соответствующих множествах C и E.

Множество 2-предикатов из класса $\Pi_{C,E}^S$, включающее все 2-диагонали, назовём *S-замкнутым*, если оно содержит конъюнкцию любых двух своих 2-предикатов, вместе с любым своим 2-предикатом содержит все его проекции и все 2-предикаты, получаемые из него подстановкой переменных, а также вместе с любым своим m-местным 2-предикатом (p_1, p_2) содержит всякий m-местный 2-предикат (q_1, q_2) из $\Pi_{C,E}^S$, такой, что $p_1 \subseteq q_1$ и $q_2 \subseteq p_2$. При этом включение $p \subseteq q$ для m-местных предикатов p и q из Π_E (или из Π_C) означает, что для любого набора x из множества E^m (соответственно из C^m) верна импликация $p(x) \Rightarrow q(x)$. Имеет место

Теорема 2. Для любого множества M функций из $P_{C,E}$ множество $\mathrm{inv}_{C,E}^S(M)$ является S-замкнутым классом 2-предикатов. Для любого S-замкнутого класса N 2-предикатов верно: $N = \mathrm{inv}_{C,E}^S(\mathrm{pol}_{C,E}(N))$.

Доказательство. Для доказательства можно воспользоваться схемой рассуждений из [2] или из [3, 4]. Приведём, однако, независимое доказательство. Первое предложение теоремы проверяется непосредственно; во втором — включение $N \subseteq \mathrm{inv}_{C,E}^S(\mathrm{pol}_{C,E}(N))$ очевидно. Докажем обратное включение. Для этого выберем произвольно l-арный 2-предикат (p_1, p_2) в множестве $\mathrm{inv}_{C,E}^S(\mathrm{pol}_{C,E}(N))$. Пусть предикату p_2 удовлетворяют наборы $X_1, \ldots, X_n$ из множества E^l, и только они. Дополним их до наборов $Z_1, \ldots, Z_n$ из множества E^m при некотором $m \geqslant l$ (добавив $m - l$ новых координат к каждому набору) так, чтобы в строках матрицы $(Z_1^T, \ldots, Z_n^T)$ содержались все наборы из множества E^n. Очевидно, это можно сделать. Рассмотрим m-местный 2-предикат (q_1, q_2), где предикату q_2 удовлетворяют всевозможные наборы $g^{[m]}(Z_1, \ldots, Z_n)$ для всевозможных n-местных функций g из $R(U\mathrm{pol}_{C,E}^S(N) \cup S_E)$, а предикату q_1 — наборы $g^{[m]}(Z_1, \ldots, Z_n)$ для всевозможных n-местных функций g из $\mathrm{pol}_{C,E}^S(N)$. Можно проверить непосредственно, что 2-предикат (q_1, q_2) принадлежит классу $\mathrm{inv}_{C,E}^S(\mathrm{pol}_{C,E}(N))$.

Для 2-предиката (q'_1, q'_2), полученного из (q_1, q_2) проектированием по добавленным $m - l$ координатам, верно, что $p_2 \subseteq q'_2$ и $q'_1 \subseteq p_1$. Отсюда, поскольку класс N S-замкнут, для завершения доказательства достаточно показать, что 2-предикат (q_1, q_2) принадлежит классу N. Для этого рассмотрим 2-предикат (q''_1, q''_2), такой, что

$$q''_i(x_1, \ldots, x_m) \equiv \bigwedge_A r_i(x_1, \ldots, x_m)$$

для $i = 1, 2$, где конъюнкция вычисляется по всем m-местным 2-предикатам (r_1, r_2) из N, таким, что $q_2 \subseteq r_2$. Множество таких 2-предикатов (r_1, r_2) обозначено выше

буквой A. Очевидно, что 2-предикат (q_1'', q_2'') вслед за 2-предикатами из множества A принадлежит классу N и выполняется включение $q_2 \subseteq q_2''$. Докажем включение $q_1'' \subseteq q_1$ (тогда из принадлежности 2-предиката (q_1'', q_2'') S-замкнутому классу N следует принадлежность 2-предиката (q_1, q_2) тому же классу, и всё доказано). Выберем произвольно набор Z_0, удовлетворяющий предикату q_1'', и определим n-местную функцию g из P_E при помощи выражения $g^{[m]}(Z_1, \ldots, Z_n) = Z_0$. Функция g определена корректно. Действительно, если в матрице $(Z_1^T, \ldots, Z_n^T)$ k-я и j-я строки совпадают, то множеству A принадлежит 2-предикат (r_1, r_2), где

$$r_i(x_1, \ldots, x_m) \equiv x_k = x_j$$

при $i = 1, 2$. Следовательно, k-я и j-я координаты совпадают у всех наборов, удовлетворяющих предикату q_1'', в частности у набора Z_0. Аналогично, функция g сохраняет все предикаты из N. Действительно, если t-местный 2-предикат (r_1', r_2') принадлежит классу N и предикату r_2' удовлетворяют наборы $(Z_{j,i_1}, \ldots, Z_{j,i_t})$ (через $Z_{j,i}$ обозначена i-я координата набора Z_j) при $1 \leqslant j \leqslant n$, то множеству A принадлежит 2-предикат (r_1, r_2), где

$$r_i(x_1, \ldots, x_m) \equiv r_i'(x_{i_1}, \ldots, x_{i_t})$$

при $i = 1, 2$. Отсюда набор Z_0 удовлетворяет предикату r_1, а набор $(Z_{0,i_1}, \ldots, Z_{0,i_t})$ — предикату r_1'. Функция g, таким образом, сохраняет 2-предикат (r_1', r_2').

Итак, функция g сохраняет все предикаты из множества N и принадлежит, таким образом, классу $\mathrm{pol}_{C,E}(N)$. Функция g сохраняет, следовательно, и принадлежащий множеству $\mathrm{inv}^S_{C,E}(\mathrm{pol}_{C,E}(N))$ 2-предикат (q_1, q_2). Отсюда, так как наборы $Z_1, \ldots, Z_n$ удовлетворяют предикату q_2, то набор $Z_0 = g^{[m]}(Z_1, \ldots, Z_n)$ удовлетворяет предикату q_1. Включение $q_1' \subseteq q_1$ доказано. Теорема доказана. ■

ЛИТЕРАТУРА

1. *Курош А. Г.* Лекции по общей алгебре. СПб.: Изд-во «Лань», 2005.
2. *Боднарчук В. Г., Калужнин Л. А., Котов В. Н., Ромов Б. А.* Теория Галуа для алгебр Поста // Кибернетика 1969. № 3. С. 1–10; № 5. С. 1–9.
3. *Geiger D.* Closed systems of functions and predicates // Pacific journal of mathematics. 1968. V. 27. No. 1. P. 95–100.
4. *Pippenger N.* Galois theory for minors of finite functions // Discrete Mathematics. 2002. V. 254. P. 405–419.
5. *Pöshel R., Kalužnin L. A.* Funktionen- und Relationenalgebren. Berlin: WEB Deutscher Verlag der Wissenschaften, 1979.
6. *Hellerstein L.* On generalized constraints and certificates // Discrete Mathematics. 2001. V. 226. P. 211–232.
7. *Мальцев А. И.* Итеративные алгебры Поста. Новосибирск: Изд-во Новосиб. ун-та, 1976.
8. *Мальцев А. И.* Итеративные алгебры и многообразия Поста // Алгебра и логика. 1966. Т. 5. № 2. С. 5–24.
9. *Парватов Н. Г.* Наследственные системы дискретных функций // Дискрет. анализ и исслед. операций. Сер. 2. 2007. Т. 14. № 2. С. 76–91.

DOI 10.17223/20710410/8/3
УДК 519.1

СВОЙСТВА h-ПЕРИОДИЧЕСКИХ ПОСЛЕДОВАТЕЛЬНОСТЕЙ

В. М. Фомичев

Институт проблем информатики РАН, г. Москва, Россия

E-mail: fomichev@nm.ru

Введено понятие h-периодичности последовательностей, связанное с отображением h мультиграмм последовательности в некоторое множество. Исследованы свойства h-периодических последовательностей, при аддитивных функциях h установлена связь длин периода и h-периода последовательности. При некоторых аддитивных функциях h исследована длина h-периода линейных рекуррентных последовательностей над конечным полем и последовательностей де Брёйна. Показано, что криптографические свойства ряда генераторов гаммы с неравномерным движением зависят от длины h-периода управляющей гаммы, где h — функция маркировки слов.

Ключевые слова: *период последовательности, аддитивная функция, линейная подстановка.*

Введение

Для генерации последовательностей (гаммы) в криптографических схемах часто используется последовательное соединение автономного автомата (управляющего блока), в котором информация продвигается равномерно в соответствии с заданной тактовой частотой, и неавтономного автомата (генерирующего блока), в котором продвижение информации зависит от знаков управляющей гаммы. Такие генераторы гаммы реализуют внешнее управление неравномерным движением информации.

Свойства выходной гаммы и преобразований состояний генератора с внешним управлением неравномерным движением существенным образом определяются свойствами управляющей гаммы. Например, в генераторах «δ-τ шагов» и в генераторах с перемежающимся шагом, построенных на основе линейных регистров сдвига (ЛРС) с максимальными длинами периодов, порядок линейной подгруппы циклической группы преобразований состояний генератора определяется длиной периода управляющей гаммы [1, разд. 18.4.2, 2]: линейные уравнения гаммообразования соответствуют всем тактам работы генератора, номера которых кратны длине периода управляющей гаммы.

В некоторых генераторах линейные уравнения гаммообразования могут встречаться весьма часто (это ослабляет свойства гаммы); например, если управляющая гамма является σ-периодической последовательностью [3], то есть если существует разбиение последовательности на слова одинаковой длины, при котором сумма элементов в каждом слове одинакова. Любая периодическая последовательность разбивается на такие отрезки, например на периоды. Для обеспечения положительных криптографических свойств важно, чтобы не существовало более глубокого разбиения последовательности на указанные слова.

В настоящей работе определяются h-периодические последовательности, где h — некоторое обобщение хеш-функции, позволяющее обобщить и свойство σ-периодичности.

Для аддитивных функций h доказано, что длина h-периода периодической последовательности делит длину периода, и исследованы длины h-периодов линейных рекуррентных последовательностей над конечным полем и последовательностей де Брёйна.

Для широкого класса генераторов гаммы с внешним управлением неравномерным движением показано, что доля линейных уравнений относительно знаков промежуточного состояния среди уравнений гаммообразования, соответствующих знакам периода гаммы, равна $1/\tau$, где τ — длина h-периода управляющей гаммы и h — аддитивная функция маркировки слов.

1. Периодические и h-периодические последовательности

Пусть $\mathbb{N}$ — множество натуральных чисел; $\mathbb{N}_0 = \mathbb{N} \cup \{0\}$; $X_\to = \{x_0, x_1, \ldots\}$ — последовательность над множеством X; $X^* = \bigcup\limits_{s\geqslant 1} X^s$ — множество всех слов натуральной длины в алфавите X; (t, τ) — наибольший общий делитель чисел $t, \tau \in \mathbb{N}$.

Множество X^* образует полугруппу относительно операции конкатенации. Результат конкатенации слова u длины ℓ и слова u' длины ℓ' есть слово uu' длины $\ell+\ell'$. Слово $x_\nu, x_{\nu+1}, \ldots, x_{\nu+s-1}$ длины s в алфавите X (обозначим его $x_{(\nu,s)}$) называют s-граммой последовательности $X_\to$, где $s \in \mathbb{N}$, $\nu \in \mathbb{N}_0$.

Последовательность $X_\to$ называют периодической, если $x_i = x_{i+t}$ при всех $i \geqslant \nu$. При этом говорят, что в $X_\to$ имеются совпадения на расстоянии t с начальным номером ν. Наименьшее из расстояний совпадения t называют длиной периода последовательности $X_\to$ (обозначается $t(X_\to)$), и при $t = t(X_\to)$ наименьший из начальных номеров совпадения ν называют длиной ее предпериода (обозначается $\nu(X_\to)$).

Обозначим для краткости $t = t(X_\to)$, $\nu = \nu(X_\to)$. Тогда период последовательности $X_\to$ есть слово $x_{\nu+i}, x_{\nu+i+1}, \ldots, x_{\nu+i+t-1}$ при любом $i \geqslant 0$, а предпериод — слово $x_0, x_1, \ldots, x_{\nu-1}$ при $\nu > 0$. Если $\nu(X_\to) = 0$, то $X_\to$ предпериода не имеет и называется чисто периодической последовательностью. Напомним элементарные свойства периодических последовательностей [1, с.130].

Свойство 1. Если в $X_\to$ имеются совпадения на расстоянии τ с начальным номером μ, то t делит τ и $\nu = \mu$.

Свойство 2. Пусть $X_\to$ — периодическая последовательность, $Y_\to = f^*(X_\to) = \{f(x_i)\}$, где $f : X \to Y$. Тогда $Y_\to$ — также периодическая последовательность, при этом $\nu(Y_\to) \leqslant \nu(X_\to)$ и $t(Y_\to)$ делит $t(X_\to)$. В частности, если f — биекция, то $\nu(Y_\to) = \nu(X_\to)$ и $t(Y_\to) = t(X_\to)$.

Свойство 3. Множество периодических последовательностей над полем P, длины периодов которых делят натуральное t, образуют линейное пространство над P.

Рассмотрим функцию $h : X^* \to Y$, где Y — некоторое множество. Функцию h можно рассматривать как обобщение хеш-функции. Через h^s обозначим ограничение функции h на множество X^s, то есть $h^s : X^s \to Y$, $s \geqslant 1$.

При натуральном s и при $\mu \in \mathbb{N}_0$ последовательности $X_\to$ поставим в соответствие последовательность $X_\to^{\mu,s}$ её s-грамм: $X_\to^{\mu,s} = \{(x_{(\mu+ks,s)}), k = 0, 1, \ldots\}$, которой также соответствует последовательность $h(X_\to^{\mu,s})$ над Y: $h(X_\to^{\mu,s}) = \{h^s(x_{(\mu+ks,s)}) : k = 0, 1, \ldots\}$.

Последовательность $X_\to$ назовем h-периодической, если при некоторых $s \in \mathbb{N}$ и $\mu \in \mathbb{N}_0$

$$h(x_{(\mu+ks,s)}) = h(x_{(\mu+(k+1)s,s)}), \quad k = 0, 1, \ldots \tag{1}$$

Равенство (1) будем интерпретировать так: в $X_\to$ имеются h^s-совпадения с начальным номером μ. Наименьшее из таких s назовем длиной h-периода последовательности $X_\to$ (обозначается $t_h(X_\to)$, кратко t_h), и если t_h — длина h-периода, то наименьший

из начальных номеров совпадения μ назовем длиной h-предпериода последовательности $X_\rightarrow$ (обозначается $\nu_h(X_\rightarrow)$, кратко ν_h). Если (1) выполняется при $\mu = 0$, то последовательность $X_\rightarrow$ назовем чисто h-периодической. Для последовательности $X_\rightarrow$ назовем h-периодом слово $x_{(\mu+ks,s)}$ и h-предпериодом — слово $x_0, x_1, \ldots, x_{\mu-1}$, где $s = t_h$ и $\mu = \nu_h, k = 0, 1, \ldots$

Заметим, что если в периодической последовательности имеются h^s-совпадения с начальным номером μ, то не обязательно t_h делит s и ν_h равно μ, то есть аналогия со свойством 1 не имеет места. Это подтверждается примером чисто периодической последовательности $X_\rightarrow$ над $\mathbb{N}_0$ при $h^s(x_i, x_{i+1}, \ldots, x_{i+s-1}) = x_i + x_{i+1} + \ldots + x_{i+s-1}$, где $s > 0$, $i \in \mathbb{N}_0$, $a \in \mathbb{N}$:

$$X_\rightarrow = \{a, 2a, 0, 0, 2a, a, a, 2a, 0, 0, 2a, a, a, 2a, 0 \ldots\} \tag{2}$$

Длина периода последовательности $X_\rightarrow$ равна 6, в $X_\rightarrow$ имеются h^3-совпадения с начальным номером 0 и h^2-совпадения с начальным номером 1, но нет h^1-совпадений. Следовательно, $t_h(X_\rightarrow) = 2$, $\nu_h(X_\rightarrow) = 1$.

Отметим элементарные свойства h-периодических последовательностей.

Свойство 4. Периодическая последовательность $X_\rightarrow$ с длиной предпериода ν и длиной периода t является h-периодической для любой функции $h : X^* \to Y$; при этом $t_h \leqslant t$, $\nu_h \leqslant \nu$ (в $X_\rightarrow$ имеются h^t-совпадения с начальным номером ν).

Свойство 5. Подпоследовательность $\{x_i, x_{i+1}, \ldots\}$ h-периодической последовательности $X_\rightarrow$ с длиной h-периода t_h и длиной h-предпериода ν_h является чисто h-периодической, если и только если $(i - \nu_h)$ кратно t_h.

Свойство 6. Не всякая h-периодическая последовательность является периодической.

Примером является последовательность хаотически чередующихся s-грамм u и w в алфавите X, где $h^s(u) = h^s(w)$:

$$X_\rightarrow = \{u, w, w, u, u, w, u, w, w, u, u, w, u, w, u, w, w, u \ldots\}.$$

В чисто h-периодической последовательности $X_\rightarrow$ имеются h^s-совпадения, значит, $X_\rightarrow$ имеет длину h-периода не более s, но не является периодической, так как построена как апериодическая последовательность s-грамм u и w.

2. Связь длин периодов и h-периодов последовательностей при аддитивной функции

Пусть Y — аддитивная полугруппа. Функцию $h : X^* \to Y$ назовем аддитивной, если для любого слова w длины $s > 1$ из того, что $w = uu'$, где $u \in X^\ell$, $u' \in X^r$, $\ell + r = s$, следует, что

$$h^s(w) = h^\ell(u) + h^r(u').$$

Пример 1 (аддитивные функции).

1) Длина слова u, то есть функция $L : X^* \to \mathbb{N}$, определенная для $u = x_1 x_2 \ldots x_\ell \in \in X^\ell$ формулой
$$L(u) = \ell.$$
2) Частота символа a в слове u, где $a \in X$; обозначим эту функцию $m_a(u)$.
3) Пусть $X = \{a_1, a_2, \ldots, a_k\}$, $m_i(u)$ — частота символа a_i в слове u, $i = 1, \ldots, k$. Функцией маркировки слов назовем функцию $m : X^* \to \mathbb{N}_0^k$, определенную формулой
$$m(u) = (m_1(u), \ldots, m_k(u)).$$

4) Пусть $X = \mathbb{N}_0$ или $X = \mathrm{GF}(k)$, где k — простое, и $u = x_1 x_2 \dots x_\ell \in X^\ell$. Функцией веса слов из X^* назовем функцию $\mathrm{wt} : X^* \to \mathbb{N}_0$, определенную формулой

$$\mathrm{wt}(u) = \mathrm{wt}(x_1) + \dots + \mathrm{wt}(x_\ell),$$

где $\mathrm{wt}(x_i) = x_i$ для любого $x_i \in X$.

Теорема 1. Пусть $X_\to = \{x_0, x_1, \dots\}$ — чисто периодическая последовательность с длиной периода t и длиной h-периода t_h, где h — аддитивная функция. Тогда t_h делит t.

Доказательство. По свойству 4 $t_h \leqslant t$. Пусть t_h не делит t, тогда разделим t на t_h с остатком:

$$t = kt_h + r, \quad 0 < r < t_h. \tag{3}$$

Покажем, что в последовательности $X_\to$ имеются h^θ-совпадения, где $\theta = (t_h, r)$. Так как $\theta < t_h$, то это приводило бы к противоречию, состоящему в том, что длина h-периода меньше t_h.

В соответствии с определением числа θ имеем: $t_h = p\theta$, $r = \delta\theta$, где $(p, \delta) = 1$. Тогда из (3) следует: $t = q\theta$, где $q = (kp + \delta)$.

Представим последовательность $X_\to$ как конкатенацию t_h-грамм: $X_\to = \{u_0 u_1 \dots\}$, где $u_i = x_{ip\theta} x_{ip\theta+1} \dots x_{ip\theta+p\theta-1}$, и как конкатенацию θ-грамм: $X_\to = \{w_0 w_1 \dots\}$, где $w_i = x_{i\theta} x_{i\theta+1} \dots x_{i\theta+\theta-1}$, $i \in \mathbb{N}_0$. Так как $\mathrm{HOK}(t, t_h) = qp\theta$, то слово $x_0 x_1 \dots x_{qp\theta-1}$ в последовательности $X_\to$ есть, с одной стороны, конкатенация t_h-грамм: $u_0 u_1 \dots u_{q-1}$, и, с другой стороны, конкатенация θ-грамм: $w_0 w_1 \dots w_{qp-1}$. Длина h-периода последовательности $X_\to$ есть t_h, поэтому при $i = 0, 1, \dots, q-1$

$$h^{p\theta}(u_i) = h^{p\theta}(u_{i+1}). \tag{4}$$

Вместе с тем из равенства $(p, \delta) = 1$ следует, что $(p, q) = 1$, значит (в соответствии с леммой Шора), упорядоченный набор слов $(u_0, u_1, \dots, u_{q-1})$ есть перестановка упорядоченного набора слов $(z_0, z_1, \dots, z_{q-1})$, где $z_i = w_i w_{i+1} \dots w_{i+p-1}$, $i = 0, 1, \dots, q-1$. Тогда из (4) получаем при $i = 0, 1, \dots, q-1$

$$h^{p\theta}(z_i) = h^{p\theta}(z_{i+1}).$$

Отсюда в соответствии с аддитивностью функции h выполнена цепь равенств:

$$\begin{aligned} h^\theta(w_i) + h^\theta(w_{i+1}) + \dots + h^\theta(w_{i+p-1}) \;&=\; h^{p\theta}(z_i) = \\ &= h^{p\theta}(z_{i+1}) \;= h^\theta(w_{i+1}) + \dots + h^\theta(w_{i+p-1}) + h^\theta(w_{i+p}), \end{aligned}$$

из которой следует, что $h^\theta(w_i) = h^\theta(w_{i+p})$, $i = 0, 1, \dots, q-1$. Последняя система равенств равносильна при $(p, q) = 1$ системе равенств

$$h^\theta(w_i) = h^\theta(w_{i+1}),$$

которая выполнена не только для $i = 0, 1, \dots, q-1$, но в силу периодичности последовательности $X_\to$ и для всех $i \in \mathbb{N}_0$. Отсюда следует, что длина h-периода последовательности $X_\to$ не превышает θ; значит, она меньше t_h, т. е. имеем противоречие. ■

Следствие 1. Пусть t — простое, тогда $t_h = 1$, если $h(x_0) = \dots = h(x_{t-1})$, и $t_h = t$ в остальных случаях.

3. h-периодичность рекуррентных последовательностей

Обозначим через ЛРПmax-n линейную рекуррентную последовательность порядка n над произвольным полем P порядка k с максимальной длиной периода, то есть $t = k^n - 1$.

Теорема 2. Для ЛРПmax-n в каждом из следующих случаев $t_h = k^n - 1$:

а) $h = m_a(u)$, где a отлично от нуля поля P;

б) $h = m(u)$;

в) $h = \mathrm{wt}(u)$, где $P = \mathrm{GF}(2)$ или $P = \mathrm{GF}(3)$.

Если $P = \mathrm{GF}(k)$, где $k > 3$ — простое, то $t_{\mathrm{wt}} = (k^n - 1)/d$, где d делит $(k-1)/2$.

Доказательство. На периоде ЛРПmax-n содержится $k^{n-1} - 1$ нулей и по k^{n-1} остальных элементов поля P [1, с. 166]. Длина h-периода ЛРПmax-n делит длину периода в силу аддитивности функции $m_a(u)$, то есть $k^n - 1 = dt_h$, где $d \geqslant 1$. Тогда период ЛРПmax-n разделяется на такие слова $u_1, \ldots, u_d$ одинаковой длины, что $m_a(u_1) = \ldots = m_a(u_d)$. Следовательно, d делит $m_a(u)$, где $m_a(u) = k^{n-1}$, что возможно только при $d = 1$, так как $(k^n - 1, k^{n-1}) = 1$. Тем самым доказано и «б». Также доказано и «в» при $P = \mathrm{GF}(2)$, так как в этом случае функции $m_1(u)$ и $\mathrm{wt}(u)$ совпадают.

По теореме 1 $t_{\mathrm{wt}} = (k^n - 1)/d$, где d делит $k^n - 1$. Вес периода u ЛРПmax-n над простым полем $\mathrm{GF}(k)$, где $k \geqslant 3$, равен

$$\mathrm{wt}(u) = \sum_{i=1}^{k-1} i \cdot k^{n-1} = \frac{k^n(k-1)}{2}.$$

Тогда в силу аддитивности функции wt вес wt-периода ЛРПmax-n равен $k^n(k-1)/(2d)$. Следовательно, d делит $(k-1)/2$, так как $(k^n - 1, k^{n-1}) = 1$.

Отсюда имеем, в частности, что $d = 1$ при $k = 3$, т. е. $t_{\mathrm{wt}} = 3^n - 1$ при $P = \mathrm{GF}(3)$. ■

Чисто периодическую рекуррентную последовательность порядка n над множеством X, где $|X| = k$, называют нормальной рекуррентной последовательностью, если длина ее периода равна k^n, и обозначают НРП(k,n). Генерируются НРП(k,n) полноцикловыми регистрами сдвига длины n над множеством X. НРП(2,n) называют последовательностями де Брёйна. Обзор свойств НРП(k,n) дан в [4].

Теорема 3. В любой НРП(2,n) имеются h-совпадения на расстоянии 2^{n-1} при $n > 0$ и при всех функциях h из $\{m_0(u), m_1(u), m(u), \mathrm{wt}(u)\}$.

Доказательство. Пусть $X_{\rightarrow} = \{x_0, x_1, \ldots\}$ — последовательность де Брёйна, имеющая длину периода 2^n. На ее периоде имеется 2^{n-1} единиц и столько же нулей. Следовательно, имеется хотя бы одно разделение периода последовательности де Брёйна на два слова u_1, u_2 длины 2^{n-1}, таких, что $m_0(u_1) = m_0(u_2)$ и $m_1(u_1) = m_1(u_2)$. Следовательно, при указанном разделении $m(u_1) = m(u_2)$ и $\mathrm{wt}(u_1) = \mathrm{wt}(u_2)$. ■

Следствие 2. Длина h-периода последовательности де Брёйна порядка n равна 2^r, где $r < n$, при всех функциях h из $\{m_0(u), m_1(u), m(u), \mathrm{wt}(u)\}$.

4. К анализу генераторов гаммы с неравномерным движением

При анализе линейности уравнений гаммообразования, связанных с генераторами гаммы с внешним управлением неравномерным движением, важным свойством является h-периодичность управляющей гаммы.

Рассмотрим класс генераторов, включающий генераторы «δ-τ шагов» и генераторы с перемежающимся шагом. Пусть $X_{\rightarrow}$ — последовательность над простым по-

лем $X = \mathrm{GF}(k)$, управляющая движением информации в линейных регистрах сдвига ЛРС-0, ..., ЛРС-$(k-1)$ над полем P, которые реализуют линейные подстановки $g_0, \ldots, g_{k-1}$ векторных пространств определенных размерностей. В i-м такте подстановка $g(i)$ пространства P^n состояний набора ЛРС-0, ..., ЛРС-$(k-1)$ определяется знаком x_i управляющей гаммы, схемой движения регистров, задаваемой матрицей $\Delta = (\delta(i,j))$ над $\mathbb{N}_0$ размера $k \times k$ (строки матрицы различны), и набором подстановок $g = (g_0, \ldots, g_{k-1})$. Пусть в i-м такте состояние всех ЛРС генератора есть $y(i) = (y_0(i), \ldots, y_{k-1}(i))$, где $y_j(i)$ — состояние ЛРС-j, $j = 0, \ldots, k-1, i \geqslant 0$. Тогда

$$y_j(i+1) = g_j^{\delta(x_i,j)}(y_j(i)).$$

Знак γ_i выходной гаммы генератора есть сумма битов, записанных в i-м такте в крайних ячейках всех ЛРС (как в генераторе с перемежающимся шагом).

Пусть $m_j(i,\tau)$ — частота символа j в слове $x_{(i,\tau)}$ и $G(i,\tau) = g(i)\cdot g(i+1)\cdot\ldots\cdot g(i+\tau-1)$; тогда

$$G(i,\tau) = (g_0^{z_0(i,\tau)}, \ldots, g_{k-1}^{z_{k-1}(i,\tau)}),$$

где $z_j(i,\tau) = m_0(i,\tau)\cdot\delta(0,j) + \ldots + m_{k-1}(i,\tau)\cdot\delta(k-1,j)$ — суммарная продвижка ЛРС-j при управляющем слове $x_{(i,\tau)}, j = 0, \ldots, k-1$. Заметим, что $G(i,\tau)$ и $G(\ell,\tau)$ суть одинаковые линейные подстановки пространства P^n, если одинаковы наборы величин $(z_0(i,\tau), \ldots, z_{k-1}(i,\tau))$ и $(z_0(\ell,\tau), \ldots, z_{k-1}(\ell,\tau))$. Отсюда если m — функция маркировки слов и $x_{(i,\tau)}$ есть m-период управляющей гаммы, то наборы величин $(z_0(i+r\tau,\tau), \ldots, z_{k-1}(i+r\tau,\tau))$ одинаковы при любом $r = 0, 1, \ldots$ Следовательно, если длина m-периода неизвестной чисто m-периодической управляющей последовательности равна τ, то линейные подстановки $G(i+r\tau,\tau)$ однозначно определены при некотором $i \in \{0, \ldots, \tau-1\}$ и при $r = 0, 1, \ldots$, поэтому знаки $\gamma_{i+r\tau}$ гаммы линейно выражаются через знаки состояния $y(i)$ генератора.

Выводы

1) Для криптографических приложений важным свойством является h-периодичность последовательностей при различных функциях h.
2) Наилучшие криптографические свойства ряда генераторов с неравномерным движением, связанные с нелинейностью уравнений гаммообразования, достигаются в схемах с управляющей гаммой, имеющей большие длины периода и m-периода, где m — функция маркировки слов.

ЛИТЕРАТУРА

1. *Фомичев В. М.* Методы дискретной математики в криптологии. М.: ДИАЛОГ-МИФИ, 2010. 424 с.
2. *Фомичев В. М., Фомичев Н. В.* Исследование линейных подсистем нелинейных систем уравнений гаммообразования // Системы высокой доступности. М.: Радиотехника, 2009. № 4. Т. 5. С. 28–33.
3. *Горъков И. Д.* Свойства σ-периодических последовательностей // Системы высокой доступности. М.: Радиотехника, 2009. № 4. Т. 5. С. 34–37.
4. *Агибалов Г. П.* Нормальные рекуррентные последовательности // Вестник Томского госуниверситета. 2007. Приложение № 23. С. 4–11.

2010 Теоретические основы прикладной дискретной математики №2(8)

DOI 10.17223/20710410/8/4

УДК 519.719.325

АДДИТИВНЫЙ ПОДХОД К ОПРЕДЕЛЕНИЮ СТЕПЕНИ НЕЛИНЕЙНОСТИ ДИСКРЕТНОЙ ФУНКЦИИ[1]

А. В. Черемушкин

Институт криптографии, связи и информатики, г. Москва, Россия

E-mail: avc238@mail.ru

В работе предлагается подход к определению степени нелинейности дискретных функций, заданных на абелевых группах, инвариантный по отношению к введению мультипликативных операций. В качестве приложения введенного понятия описан алгоритм нахождения групп инерции в группе сдвигов для функций p^m-значной логики.

Ключевые слова: *дискретные функции, степень нелинейности, группа инерции.*

1. Конечные производные и степень нелинейности функций на абелевых группах

Рассмотрим функции $F : G \to H$, у которых на множествах G и H заданы структуры абелевых групп.

Определим *производные по направлению* $\Delta_a F$, $a \in G$, функции F равенствами

$$\Delta_a F(x) = F(x + a) - F(x),$$

где $x \in G$.

Следующие свойства очевидно вытекают из определения.

A1. *При всех* $a \in G$ *выполняется равенство*

$$\Delta_a(F_1 + F_2) = \Delta_a F_1 + \Delta_a F_2.$$

Для обозначения нейтрального элемента групп G и H будем использовать символ 0.

A2. *Если* $\Delta_a F(0) = 0$ *для всех* $a \in G$, *то и* $\Delta_a F(x) = 0$ *для всех* $a, x \in G$.

A3. *При всех* $a, b \in G$ *выполняется равенство*

$$\Delta_{a+b} F(x) = \Delta_a F(x + b) - \Delta_b F(x).$$

A4. *При всех* $a, b \in G$ *выполняется равенство*

$$\Delta_a \Delta_b F = \Delta_{a+b} F - \Delta_a F - \Delta_b F.$$

Степенью нелинейности функции $F : G \to H$ (обозначается $\mathrm{dl}\ F$) называется минимальное натуральное число m, такое, что

$$\Delta_{a_1} \dots \Delta_{a_{m+1}} F(x) = 0$$

при всех $a_1, \dots, a_{m+1}, x \in G$.

[1]Работа выполнена при поддержке гранта Президента РФ НШ № 4.2008.10.

Заметим, что степень нелинейности определена не для всех функций и абелевых групп. Например, для функции 6-значной логики F, задающей подстановку $(0,2,1,5)(3)(4)$, последовательность

$$\Delta_1 F,\ \ \Delta_1\Delta_1 F,\ \ \Delta_1\Delta_1\Delta_1 F,\dots$$

является периодической, причем ни одна из функций этой последовательности не равна тождественно нулю.

Ниже будет показано, что если на множестве аргументов и значений функции задана структура элементарной абелевой группы, то степень нелинейности определена всегда.

Для степени нелинейности функции $F : G \to H$ выполняются следующие очевидные свойства.

B1. *Функция F имеет первую степень нелинейности в том и только в том случае, когда она имеет вид $F(x) = \varphi(x) + a$, $\varphi \in \mathrm{Hom}(G,H)$ — гомоморфизм, $a \in G$.*

B2. *Если* $\mathrm{dl}\ F$ *определена, то* $\mathrm{dl}\ F$ *— это максимальное m, такое, что при некоторых $a_1,\dots,a_m \in G$*

$$\Delta_{a_1}\dots\Delta_{a_m}F(0) \neq 0.$$

Вытекает из свойства A2 производных.

B3. *Если* $\mathrm{dl}\ F$ *определена, то* $\mathrm{dl}\ \Delta_a F \leqslant \mathrm{dl}\ F - 1$ *при всех $0 \neq a \in \Omega$, причем всегда найдется такой элемент $0 \neq a \in \Omega$, что* $\mathrm{dl}\ \Delta_a F = \mathrm{dl}\ F - 1$.

B4. *Если* $\mathrm{dl}\ F$ *определена, то для любых функций $F_1(x) = \psi(x)+a$, $F_2(x) = \varphi(x)+b$, где $\psi \in \mathrm{End}(G)$, $\varphi \in \mathrm{End}(H)$ — эндоморфизмы, $a \in G$, $b \in H$, выполнено равенство*

$$\mathrm{dl}\ F = \mathrm{dl}\ (F_2 \circ F \circ F_1),$$

где $\circ$ — операция композиции отображений, $(F_2 \circ F \circ F_1)(x) = F_2(F(F_1(x)))$.

B5. *Если для функций $F_i : G \to H$, $i = 1,2$, определены* $\mathrm{dl}\ F_1$ *и* $\mathrm{dl}\ F_2$, *то*

$$\mathrm{dl}\ (F_1 + F_2) \leqslant \max\{\mathrm{dl}\ F_1, \mathrm{dl}\ F_2\}.$$

2. Конечные производные функций на элементарных абелевых группах

Пусть на множестве аргументов и значений произвольной функции $F : G \to H$ заданы структуры элементарных абелевых p-групп. Если естественным образом рассматривать элементарные абелевы группы как векторные пространства над полем $\mathrm{GF}(p) = \mathbf{Z}_p$, $G = \mathbf{Z}_p^n$ и $H = \mathbf{Z}_p^k$, то функция F задается набором координатных функций $(F_1,\dots,F_k)$. Поэтому помимо введенного выше «аддитивного» определения степени нелинейности, вычисляемого с помощью производных по направлению, можно применять обычное «мультипликативное» определение степени отображения как максимума степеней многочленов координатных функций (см. [1]).

Покажем, что эти два подхода к определению степени нелинейности в данном случае совпадают.

Заметим, что можно рассматривать и более общий случай функций p^m-значной логики $F : \Omega^n \to \Omega^k$, где на множестве Ω задана структура элементарной абелевой p-группы. Если на множестве Ω ввести дополнительно операцию умножения так, чтобы Ω приобрело структуру конечного поля $\mathrm{GF}(q)$, $q = p^m$, то также можно показать, что «аддитивное» определение степени нелинейности совпадает с «мультипликативным» (см. ниже). Отсюда, в частности, вытекает, что способ введения операции умножения не влияет на значение степени нелинейности функций.

Помимо перечисленных выше свойств А1–А4 производных из-за введения операции умножения оказываются справедливыми также следующие.

А5. *При всех* $a \in \Omega^n$ *выполняется равенство*

$$\Delta_a F_1 F_2(x) = \Delta_a F_1(x) F_2(x) + F_1(x+a)\Delta_a F_2(x).$$

Если функции F_1 *и* F_2 *существенно зависят от непересекающихся множеств аргументов*: $F_1(x) = f_1(x')$ и $F_2(x) = f_2(x'')$, $x = (x', x'')$, $a = (a', a'')$, *то*

$$\Delta_a F_1 F_2(x) = \Delta_a f_1 f_2(x) = \Delta_{a'} f_1(x') f_2(x'') + f_1(x')\Delta_{a''} f_2(x'').$$

А6. *Для функции* $F : \Omega \to \Omega$ *справедливо разложение в точке* $x = 0$:

$$F(x) = F(0) - \sum_{j=1}^{q-1} x^j \sum_{0 \neq a \in \mathrm{GF}(q)} \frac{\Delta_a F(0)}{a^j}.$$

Пятое свойство очевидно. Шестое свойство вытекает из формулы

$$h(i) = - \sum_{0 \neq a \in \mathrm{GF}(q)} \frac{f(a)}{a^i}, \quad i \neq 0,$$

для коэффициентов разложения функции

$$f(x) = \sum_{i=0}^{q-1} h(i) x^i.$$

Здесь использовано равенство

$$\sum_{j=1}^{q-1} \left(\frac{x}{a}\right)^j = \begin{cases} -1, & x = a; \\ 0, & x \neq a, \end{cases}$$

справедливое для элементов поля $\mathrm{GF}(q)$.

Особо отметим свойства производных при $a = 1$. Напомним, что факториальные степени определяются равенствами

$$(x)_i = \begin{cases} x(x-1)\ldots(x-i+1), & i \geqslant 0; \\ 1, & i = 0. \end{cases}$$

Факториальные степени связаны с обычными степенями соотношениями:

$$(x)_n = \sum_{j=0}^{n} s(n,j) x^j; \tag{1}$$

$$x^i = \sum_{j=0}^{n} \sigma(n,j)(x)_j, \tag{2}$$

где $s(n,j)$ и $\sigma(n,j)$ — коэффициенты, которые для числовых полей называются числами Стирлинга первого и второго рода соответственно. Кроме того, при $0 \leqslant i \leqslant n$ справедливы равенства (в поле $\mathrm{GF}(p^m)$):

А7. $\Delta_1^i (x)_n = (n)_i (x)_{n-i}$;

A8. $\Delta_1^i x^n = \sum\limits_{j=0}^{n} \sigma(n,j)(j)_i (x)_{j-i}$;

A9. $\Delta_1^n (x)_n = n!$.

Доказательство этих свойств осуществляется несложной проверкой.

Так как данные равенства рассматриваются как соотношения в поле $\mathrm{GF}(p^m)$, то для получения нетривиальных соотношений имеет смысл ограничиться случаем $0 \leqslant i \leqslant n \leqslant p-1$.

В случае, когда a — произвольный элемент поля, не обязательно равный единице, можно положить

$$(x)_{i,a} = \begin{cases} x(x-a)(x-2a)\dots(x-(i-1)a), & i \geqslant 0; \\ 1, & i = 0. \end{cases}$$

Тогда после замены x на ax получаем аналогичные равенства:

$$(x)_{n,a} = \sum_{j=0}^{n} s(n,j) a^{n-j} x^j; \tag{3}$$

$$x^i = \sum_{j=0}^{n} \sigma(n,j) a^{n-j} (x)_{j,a}. \tag{4}$$

Теперь последние три свойства можно сформулировать в более общем виде:

$A7'$. $\Delta_a^i (x)_{n,a} = (n)_i a^i (x)_{n-i,a}$;

$A8'$. $\Delta_a^i x^n = \sum\limits_{j=0}^{n} \sigma(n,j) a^{n-j+i} (j)_i (x)_{j-i,a}$;

$A9'$. $\Delta_a^n (x)_{n,a} = n! a^n$.

3. Степень нелинейности функции p^m-значной логики

Пусть, как и выше, на множестве Ω задана структура конечного поля $\mathrm{GF}(p^m)$. Согласно введенному выше определению, степенью нелинейности $\mathrm{dl}\, F$ функции p^m-значной логики F называется минимальное натуральное число m, такое, что

$$\Delta_{a_1} \dots \Delta_{a_{m+1}} F(x) = 0$$

при всех $a_1, \dots, a_{m+1} \in \Omega$.

Из свойства A5 производных вытекает следующее очевидное свойство степени нелинейности произведения функций.

B6. $\mathrm{dl}\,(F_1 \cdot F_2) \leqslant \mathrm{dl}\, F_1 + \mathrm{dl}\, F_2$. *Если функции* F_1 *и* F_2 *зависят от непересекающихся множеств переменных, то* $\mathrm{dl}(F_1 \cdot F_2) = \mathrm{dl}\, F_1 + \mathrm{dl}\, F_2$.

Стандартное «мультипликативное» значение степени нелинейности p^m-значной функции F, заданной над конечным полем, называемое также индексом нелинейности, определяется как максимальное значение величины

$$\|b_1\| + \dots + \|b_n\|$$

для всех входящих в многочлен функции одночленов $x_1^{b_1} \cdot \dots \cdot x_n^{b_n}$, $b_i \in \{0, 1, \dots, p^m - 1\}$, $1 \leqslant i \leqslant n$, где $\|b_i\|$ — сумма цифр в p-ичной записи числа b_i, $1 \leqslant i \leqslant n$.

Как показывают следующие три свойства, эти определения в данном случае равносильны. Сначала установим равносильность для случая $m = 1$.

B7. *Для функций над полем* $\mathrm{GF}(p)$ *степень нелинейности и степень функции совпадают.*

Доказательство. Пусть $f(x_1, \ldots, x_n)$, $n \geqslant 1$, — произвольная функция, многочлен которой имеет степень $\deg f$, причем все переменные входят в степенях не выше $p-1$. Из свойств B1–B3 степени нелинейности следует, что $\mathrm{dl}\, f \leqslant \deg f$. Докажем обратное неравенство.

Воспользуемся индукцией по числу n. Если $n = 1$ и $\deg f = k$, причем

$$f(x) = c_k x^k + c_{k-1} x^{k-1} + \ldots + c_1 x + c_0,$$

то, согласно равенству (2) и свойству A9, имеем $\Delta_1^k f(x) = c_k k! \neq 0$.

Предположим, что утверждение справедливо для всех функций от $n-1$ переменного. Пусть $\deg f = k \geqslant 2$ и в многочлен функции f входит одночлен $x_1^{b_1} \cdot \ldots \cdot x_s^{b_s}$, $\sum\limits_{i=1}^{s} b_i = k$, $s \geqslant 1$. Разложим многочлен функции f по первой переменной:

$$f(x_1, \ldots, x_n) = x_1^k f_k(x_2, \ldots, x_n) + x_1^{k-1} f_{k-1}(x_2, \ldots, x_n) + \ldots$$
$$\ldots + x_1 f_1(x_2, \ldots, x_n) + f_0(x_2, \ldots, x_n).$$

Тогда для вектора $a = (1, 0, \ldots, 0) \in \mathrm{GF}(p)^n$ выполнено

$$\Delta_1^{b_1} f(x_1, \ldots, x_n) = \Delta_1^{b_1} x_1^{b_1} \cdot f_k(x_2, \ldots, x_n) = b_1! \cdot f_k(x_2, \ldots, x_n).$$

По предположению индукции функция f_k имеет степень нелинейности $k - b_1$, поэтому найдется набор векторов $a_1, \ldots, a_{k-b_1} \in \mathrm{GF}(p)^n$, такой, что

$$\Delta_{a_1} \ldots \Delta_{a_{k-b_1}} f(x_2, \ldots, x_n) \neq 0.$$

Отсюда следует, что $\mathrm{dl}\, f \geqslant k$.

B8. *Если при фиксации базиса поля* $\mathrm{GF}(p^m)$, *рассматриваемого как пространство над* $\mathrm{GF}(p)$, *функция* $F : \mathrm{GF}(p^m)^n \to \mathrm{GF}(p^m)$, $n \geqslant 1$, *задается в координатном виде набором многочленов* $f_1, \ldots, f_m$ *над полем* $\mathrm{GF}(p)$ *от* mn *переменных, то*

$$\mathrm{dl}\, F = \max_{i=\overline{1,n}} \mathrm{dl}\, f_i.$$

Доказательство. Пусть $e_1, \ldots, e_m$ — стандартный базис и функция F задается в координатном виде выражением

$$F(x_1, \ldots, x_n) = \sum_{j=1}^{m} e_i f_i(x_{1,1}, \ldots, x_{1,n}, \ldots, x_{m,1}, \ldots, x_{m,n}),$$

где значения переменных в обеих частях равенства связаны соотношениями $x_i = \sum_{j=1}^{m} x_{i,j} e_i$, $j = \overline{1,n}$, $i = \overline{1,m}$.

Заметим, что каждая координата $x_{i,j}$, $j = \overline{1,n}$, выражается через x_i, $i = \overline{1,n}$, как линейная функция $\mathrm{tr}(a_{i,j} x_i)$ ($\mathrm{tr}(x) = \sum\limits_{t=0}^{m-1} x^{p^t}$ — функция след) при некотором $a_{i,j} \in \mathrm{GF}(p^m)$. Подставляя эти выражения в правую часть равенства и используя свойства B1–B3 степени нелинейности, получаем оценку

$$\mathrm{dl}\, F \leqslant \max_{i=\overline{1,n}} \mathrm{dl}\, f_i.$$

С другой стороны, если

$$\max_{i=\overline{1,n}} \mathrm{dl}\, f_i = \mathrm{dl}\, f_s = k,$$

то по определению найдутся элементы $a_1, \ldots, a_k \in \mathrm{GF}(p)^{mn}$, такие, что

$$\Delta_{a_1} \ldots \Delta_{a_k} f_s \neq 0.$$

Отсюда, с учетом того, что абелевы группы полей $\mathrm{GF}(p)^{mn}$ и $\mathrm{GF}(p^m)^n$ совпадают, получаем

$$\Delta_{a_1} \ldots \Delta_{a_k} f = \sum_{i=1}^{m} e_i \Delta_{a_1} \ldots \Delta_{a_k} f_i \neq 0,$$

то есть

$$\mathrm{dl}\ F \geqslant \max_{i=\overline{1,n}} \mathrm{dl}\ f_i.$$

B9. *Степень нелинейности одночлена*

$$x_1^{b_1} \cdot \ldots \cdot x_n^{b_n},$$

где $b_i \in \{0, 1, \ldots, p^m - 1\}$, $i = 1, \ldots, n$, *совпадает с*

$$\|b_1\| + \cdots + \|b_n\|,$$

где $\|b_i\|$ *— сумма цифр в* p*-ичной записи числа* b_i, $1 \leqslant i \leqslant n$. *Степень нелинейности многочлена над полем* $\mathrm{GF}(p^m)$ *совпадает с максимальной степенью нелинейности для входящих в него одночленов.*

Доказательство. Рассмотрим сначала случай $n = 1$. Так как при $m = 1$ это утверждение по сути уже доказано (см. свойство B7), то рассмотрим случай $m \geqslant 2$.

Пусть $0 \leqslant k \leqslant (p-1)m$.

Покажем, что множество $\mathcal{U}_k$ многочленов, в которых все входящие в них одночлены x^b, $b \in \{0, 1, \ldots, p^m - 1\}$, удовлетворяют неравенству $\|b\| \leqslant k$, совпадает с множеством $\mathcal{U}'_k$ многочленов, степень нелинейности которых не превосходит k.

Обозначим через $M(p, m, k)$ число разбиений числа k на m неотрицательных слагаемых, каждое из которых не превосходит $p - 1$. Тогда

$$|\ \mathcal{U}_k\ | = \prod_{t=0}^{k} 2^{M(p,m,t)}.$$

С другой стороны, согласно свойству B8, множество $\mathcal{U}'_k$ совпадает с множеством функций, которые задаются в координатном виде набором многочленов $f_1, \ldots, f_m$ над полем $\mathrm{GF}(p)$ от m переменных степени нелинейности k. Как нетрудно проверить, множество $\mathcal{U}'_k$ имеет в точности такую же мощность:

$$|\ \mathcal{U}'_k\ | = \prod_{t=0}^{k} 2^{M(p,m,t)}.$$

В силу очевидного неравенства $\mathrm{dl}\ x^b \leqslant \|b\|$, которое вытекает из представления

$$x^b = x^{b^{(0)}} \cdot x^{p^1 b^{(1)}} \cdot \ldots \cdot x^{p^{m-1} b^{(m-1)}},$$

$b = \sum\limits_{i=0}^{m-1} p^i b^{(i)}$, $\sum\limits_{i=1}^{s} b^{(i)} = \|b\|$, выполняется включение $\mathcal{U}_k \subseteq \mathcal{U}'_k$. Отсюда получаем $\mathcal{U}_k = \mathcal{U}'_k$.

При $n \geqslant 2$ рассуждения полностью аналогичны, за исключением того, что в данном случае

$$|\ \mathcal{U}_k\ | = |\ \mathcal{U}'_k\ | = \prod_{t=0}^{k} 2^{M(p,mn,t)}.$$

В10. *Степень нелинейности функции p^n-значной логики для случая элементарных абелевых групп определяется только свойствами операции сложения. Поэтому для заданной элементарной абелевой p-группы при любом способе задания операции умножения так, чтобы в результате получилось поле из p^n элементов, степень нелинейности функций всегда будет инвариантна по отношению к выбору операции умножения.*

В11. *Если G, H и R — элементарные абелевы p-группы, $F_1 : G \to H$, $F_2 : H \to R$ и $\circ$ — операция композиции отображений $(F_1 \circ F_2)(x) = F_1(F_2(x))$, $x \in G$, то*

$$\mathrm{dl}(F_1 \circ F_2) \leqslant \mathrm{dl}\ F_1 \cdot \mathrm{dl}\ F_2.$$

Пусть $|\ G\ | = p^n$, $|\ H\ | = p^m$ и $|\ R\ | = p^k$. В силу свойства В8 достаточно предполагать, что на множествах элементов групп G, H и R заданы структуры $\mathrm{GF}(p)^n$, $\mathrm{GF}(p)^m$ и $\mathrm{GF}(p)^k$. Тогда функции F_1 и F_2 можно задать системами из m и k уравнений от n и m переменных соответственно над полем $\mathrm{GF}(p)$. По свойству В8 степень нелинейности совпадает с максимумом степеней одночленов в многочленах, задающих эти уравнения. Остается подставить во вторую систему вместо аргументов многочлены уравнений первой системы и воспользоваться свойством В5.

В12. *Если G и H — элементарные абелевы p-группы и $R \leqslant G$ — подгруппа в G, то для степеней нелинейности функции $F : G \to H$ и ее ограничения $F|_R : R \to H$ на подгруппу R выполнено неравенство* $\mathrm{dl}(F|_R) \leqslant \mathrm{dl}\ F$.

Это свойство очевидно вытекает из определения степени нелинейности.

В заключение заметим, что вопрос о свойствах аддитивного определения степени нелинейности функций для случая задания на множествах их аргументов и значений других типов групп, например, примарных циклических, остается пока открытым.

4. Вычисление групп инерции функции p^m-значной логики в группе сдвигов

В качестве применения введенного выше понятия степени нелинейности рассмотрим метод нахождения групп инерции функций p^m-значной логики в группе сдвигов, основанный на группировке одночленов в многочленах функций p^m-значной логики по степеням нелинейности.

Пусть $F : \Omega^n \to \Omega$ — функция p^m-значной логики, причем считаем, что $\Omega = \mathrm{GF}(p^m)$. Группа инерции $(\mathrm{H}_n)_F$ этой функции в группе сдвигов H_n относительно операции сложения в поле $\mathrm{GF}(p^m)$ состоит из преобразований $\begin{pmatrix} x \\ x+a \end{pmatrix}$ сдвига на векторы $a = (a_1, ..., a_n)$, где $a_1, ..., a_n \in \mathrm{GF}(p^m)$, таких, что

$$F(x_1 + a_1, ..., x_n + a_n) = F(x_1, ..., x_n)$$

при всех $x_1, ..., x_n \in \mathrm{GF}(p^m)$.

Ниже будет описан метод, позволяющий вычислять группу инерции $(\mathrm{H}_n)_F$ по известному многочлену функции $F(x_1, ..., x_n)$ над полем $\mathrm{GF}(p^m)$. Обозначим этот многочлен $f(x_1, ..., x_n)$.

В основе метода лежит сведение исходной задачи к решению нескольких систем уравнений над полем $\mathrm{GF}(p^m)$, являющихся линейными над полем $\mathrm{GF}(p)$. Поэтому сначала, следуя [2], напомним способ решения таких систем.

4.1. Решение систем p-линейных уравнений

1. Под линейным (над $\mathrm{GF}(p)$) отображением $L : \mathrm{GF}(p^m) \to \mathrm{GF}(p^m)$ будем понимать произвольный эндоморфизм поля $\mathrm{GF}(p^m)$, рассматриваемого как линейное пространство $(\Omega, +)$ над полем $\mathrm{GF}(p)$. Многочлен $l(x)$, представляющий линейное отображение L над полем $\mathrm{GF}(p)$, называется *p-многочленом* (см. [2]).

Как известно, произвольный p-многочлен имеет вид

$$l(x) = \sum_{i=0}^{m-1} a_i x^{p^i},$$

где $a_i \in \mathrm{GF}(p^m), i \in \overline{1, m-1}$. Если зафиксировать какой-либо базис $e_1, ..., e_m$ поля $\mathrm{GF}(p^m)$ над полем $\mathrm{GF}(p)$, то L, как линейное отображение векторного пространства, однозначно задается некоторой матрицей размера $m \times m$ с элементами из поля $\mathrm{GF}(p)$.

Таким образом, решение одного уравнения $L(x) = 0$ над полем $\mathrm{GF}(p^m)$ сводится к решению системы из m линейных уравнений над полем $\mathrm{GF}(p)$. Чтобы выписать эту систему, выразим элементы $L(e_i)$ в базисе $e_1, ..., e_m$ векторного пространства $(\Omega, +)$:

$$L(e_i) = \sum_{i=0}^{m-1} b_{ik} e_k, \quad i \in \overline{1, m-1}.$$

Тогда при

$$x = \sum_{i=0}^{m-1} x^{(i)} e_i,$$

где $x^{(i)} \in \mathrm{GF}(p)$, $i \in \{\overline{1, m}\}$, получаем

$$L(x) = L(\sum_{i=0}^{m-1} x^{(i)} e_i) = \sum_{i=0}^{m-1} x^{(i)} L(e_i) = (x^{(1)}, ..., x^{(m)}) B_L$$

при некоторой матрице $B_L = (b_{ij})$. Теперь искомая система уравнений принимает вид

$$(x^{(1)}, ..., x^{(m)}) B_L = (0, ..., 0).$$

2. Рассмотрим теперь систему уравнений над полем $\mathrm{GF}(p^m)$ с линейными над $\mathrm{GF}(p)$ многочленами. Ее можно записать в виде

$$\begin{cases} L_{11}(x_1) + ... + L_{1n}(x_n) = 0, \\ \qquad\qquad ... \\ L_{k1}(x_1) + ... + L_{kn}(x_n) = 0, \end{cases} \tag{5}$$

где L_{ij} — p-многочлены, $i \in \overline{1, k}$, $j \in \overline{1, n}$. Как и выше, зафиксируем некоторый базис поля $\mathrm{GF}(p^m)$, рассматриваемого как линейное пространство над полем $\mathrm{GF}(p)$. Сопоставим каждому многочлену L_{ij} $n \times m$-матрицу B_{ij} аналогично тому, как это было сделано выше. Тогда система (5) может быть записана в виде системы линейных уравнений над полем $\mathrm{GF}(p)$

$$\left(x_1^{(1)}, ..., x_1^{(m)}, ..., x_n^{(1)}, ..., x_n^{(m)}\right) \begin{pmatrix} B_{11} & ... & B_{k1} \\ ... & ... & ... \\ B_{1n} & ... & B_{kn} \end{pmatrix} = (0, \ldots, 0). \tag{6}$$

Теперь всякому решению системы (6) соответствует решение

$$(x_1, ..., x_n) = \left(\sum_{i=1}^{m} x_1^{(i)} e_i, \ldots, \sum_{i=1}^{m} x_n^{(i)} e_i\right)$$

системы (5).

4.2. Расширения группы инерции

Определим цепочки расширений группы инерции, используя сравнения функций с точностью до многочленов степени нелинейности не выше s, $0 \leqslant s \leqslant (p-1)mn$.

Множество функций $F : \mathrm{GF}(p^m)^n \to \mathrm{GF}(p^m)$, степень нелинейности которых не превосходит s, обозначим через

$$\mathcal{U}_s = \{F(x_1, ..., x_n) \quad : \quad \mathrm{dl}\ F \leqslant s\}.$$

При $s = -1$ полагаем $\mathcal{U}_{-1} = \{0\}$.

Несложно проверить, что множество $\mathcal{U}_s$ является векторным пространством над полем $\mathrm{GF}(p^m)$, в частности, $\mathcal{U}_s$ замкнуто относительно операций сложения функций и умножения на элементы поля $\mathrm{GF}(p^m)$). Введем также множества

$$(\mathrm{H}_n)_F^{(s)} = \left\{ \begin{pmatrix} x \\ x+a \end{pmatrix} \in \mathrm{H}_n \quad : \quad F(x+a) - F(x) \in \mathcal{U}_s \right\}.$$

Несложно проверить, что при любом $s \geqslant -1$ множество $(\mathrm{H}_n)_F^{(s)}$ является подгруппой группы H_n. В частности, при $s = -1$ выполнено равенство

$$(\mathrm{H}_n)_F^{(-1)} = \left\{ \begin{pmatrix} x \\ x+a \end{pmatrix} \in \mathrm{H}_n \quad : \quad F(x+a) = F(x) \right\} = (\mathrm{H}_n)_F.$$

С другой стороны, по свойству B1 степени нелинейности $\Delta_a F(x) \in \mathcal{U}_{\mathrm{dl}\, F-1}$, следовательно,

$$(\mathrm{H}_n)_F^{(\mathrm{dl}\, F-1)} = \mathrm{H}_n.$$

Поэтому первой нетривиальной группой может быть только группа $(\mathrm{H}_n)_F^{(\mathrm{dl}F-2)}$.

Каждой группе $(\mathrm{H}_n)_F^{(s)}$ можно однозначно поставить в соответствие подпространство

$$W_s = \left\{ a = (a_1, ..., a_n) \in \mathrm{GF}(p^m)^n \quad : \quad \begin{pmatrix} x \\ x+a \end{pmatrix} \in (\mathrm{H}_n)_F^{(s)} \right\}.$$

Очевидны следующие цепочки включений:

$$\{0\} = \mathcal{U}_{-1} \subseteq \mathcal{U}_0 \subseteq ... \subseteq \mathcal{U}_{(p-1)nm},$$
$$(\mathrm{H}_n)_F = (\mathrm{H}_n)_F^{(-1)} \subseteq ... \subseteq (\mathrm{H}_n)_F^{(s)} \subseteq ... \subseteq (\mathrm{H}_n)_F^{(\mathrm{dl}\, F-1)} = \mathrm{H}_n,$$
$$W_{-1} \subseteq W_0 \subseteq ... \subseteq W_{\mathrm{dl}\, F-1} = \mathrm{GF}(p^m)^n.$$

4.3. Метод нахождения групп инерции

Рассмотрим теперь сам метод нахождения группы инерции.

Пусть функция $F(x_1, ..., x_n) \in \mathcal{U}_k$ представима многочленом $f(x_1, ..., x_n)$ степени нелинейности $\mathrm{dl}\ F = k$, $1 \leqslant k \leqslant (p-1)nm$.

Рассмотрим многочлен

$$\Delta_a f = f(x+a) - f(x),$$

где $x = (x_1, ..., x_n)$, $a = (a_1, ..., a_n) \in \mathrm{GF}(p^m)^n$. Сгруппируем одночлены, входящие в $\Delta_a f$, по степеням нелинейности:

$$\Delta_a f = \sum_{i=0}^{k-1} \sum_{b \in I_i} t_b(a) \widetilde{X}_b,$$

где через

$$\widetilde{X}_b = x_1^{b_1} \cdot \dots \cdot x_n^{b_n}$$

обозначен произвольный одночлен многочлена $\Delta_a f$, $t_b(a)$ — коэффициент при этом одночлене, а I_i обозначает множество

$$I_i = \left\{ b = (b_1, \dots, b_n) \in \{0, 1, \dots, p^m - 1\}^n \ : \ \mathrm{dl}\, \widetilde{X}_b = i \right\}.$$

Как было показано выше, группы $(\mathrm{H}_n)_F^{(s)}$ могут быть нетривиальными только при $-1 \leqslant s \leqslant \mathrm{dl}\, F - 2$.

Первым шагом предлагаемого метода является нахождение группы $(\mathrm{H}_n)_F^{(\mathrm{dl}\, F-2)}$ (или, что то же самое, подпространства W_{k-2}). Ее нахождение сводится к решению системы уравнений

$$\{t_b(a) = 0, \quad b \in I_{k-1},$$

в левой части которой стоят p-линейные по переменной a многочлены $t_b(a)$ (см. утверждение 1 ниже). Используя описанный выше метод решения системы линейных уравнений, находим множество решений W_{k-2} и, следовательно, группу $(\mathrm{H}_n)_F^{(k-2)}$.

Далее процесс нахождения группы инерции осуществляется индуктивно по мере убывания значения s, $-1 \leqslant s \leqslant k-2$.

Предположим, что уже найдена группа $(\mathrm{H}_n)_F^{(s)}$ и соответственно подпространство W_s для $s \leqslant k-2$. Теперь для нахождения множества W_{s-1} надо решить систему уравнений (вообще говоря, нелинейных)

$$\{t_b(a) = 0, \quad b \in I_s. \tag{7}$$

Покажем, что на самом деле эта система является линейной на подпространстве W_s.

Утверждение 1. Система уравнений

$$\{t_b(z) = 0, \quad b \in I_s \tag{8}$$

является системой линейных уравнений относительно $z \in W_s$.

Доказательство. Заметим, что для $z \in W_s$ выполняется включение $\Delta_z f \in \mathcal{U}_s$ по определению группы $(\mathrm{H}_n)_F^{(s)}$. Поэтому для всех $z_1, z_2 \in W_s$ по свойствам A3 и A4 производных и свойству B1 степени нелинейности выполняется условие

$$\Delta_{z_1+z_2} f - \Delta_{z_1} f - \Delta_{z_2} f = \Delta_{z_1} f \Delta_{z_2} f \in \mathcal{U}_{s-1},$$

откуда получаем

$$t_b(z_1 + z_2) = t_b(z_1) + t_b(z_2)$$

для всех $z_1, z_2 \in W_s$, $b \in I_s$. Отсюда следует, что

$$t_b(c_1 z_1 + c_2 z_2) = c_1 t_b(z_1) + c_2 t_b(z_2)$$

для всех $z_1, z_2 \in W_s$, $c_1, c_2 \in \mathrm{GF}(p)$, $b \in I_s$, что и означает, что данная система является системой линейных уравнений на пространстве W_s. ■

Таким образом, система (7) при наложении ограничения $a \in W_s$ становится линейной, а множество ее решений образует пространство W_{s-1}.

В результате, последовательно находя подпространства

$$W_{k-2}, W_{k-3}, ..., W_1, W_0, W_{-1},$$

и решая в них системы линейных уравнений, тем самым находим группу

$$(\mathrm{H}_n)_F = \left\{ \begin{pmatrix} x \\ x+a \end{pmatrix} \ : \ a \in W_{-1} \right\}.$$

Понятно, что сложность данного алгоритма определяется степенью нелинейности функции (она определяет число шагов алгоритма) и сложностью решения систем линейных уравнений над полем $\mathrm{GF}(p)$ от не более чем mn неизвестных. Поэтому в худшем случае трудоемкость можно оценить величиной $O((\mathrm{dl}\ F) \cdot (nm)^3)$.

Заметим, что трудоемкость линейно зависит от параметра $\mathrm{dl}\ F$. Для более точной оценки следует воспользоваться одним из быстрых алгоритмов решения систем уравнений, а также учесть, что число неизвестных в получаемых системах в процессе решения должно монотонно уменьшаться (меняется число n). Следует также учитывать справедливость асимптотической оценки шенноновского типа о тривиальности расширений групп инерции почти всех функций в группах сдвигов ([3], теорема 5).

Пример.

Пусть функция от трех переменных над полем

$$\mathrm{GF}(2^5) = \mathrm{GF}(2)[x]/x^5 + x^2 + 1$$

задана многочленом

$$f(x_1, ..., x_n) = x_1^4 x_2^5 + x_1^4 x_2^4 x_3^1 + x_2^4 x_3^3 + x_1^4 x_2 x_3^2 + x_1^4 x_3^3 + x_1^4 x_2^4 x_3^3 + x_2^5 x_3^2.$$

Все одночлены этого многочлена имеют степень нелинейности 3.

Сначала найдем группу $(\mathrm{H}_3)_f^{(1)}$. Подпространство W_1 состоит из векторов $x = (x_1, x_2, x_3) \in \mathrm{GF}(2^5)^3$, являющихся решениями системы уравнений

$$\begin{cases} a_2 + a_3 + a_3^2 = 0, \\ a_1^4 + a_3^2 = 0, \\ a_2^4 + a_3^2 = 0, \\ a_1^4 + a_2 + a_3 = 0, \\ a_1^4 + a_2^4 = 0, \\ a_2 + a_2^4 + a_3 = 0, \end{cases}$$

которая получается приравниванием к нулю коэффициентов многочлена Жегалкина левой части уравнения $\Delta_a f(x) = 0$ при одночленах степени нелинейности 2. Поскольку последние три уравнения являются следствиями трех первых, то достаточно решить систему

$$\begin{cases} a_2 + a_3 + a_3^2 = 0, \\ a_1^4 + a_3^2 = 0, \\ a_2^4 + a_3^2 = 0. \end{cases} \tag{9}$$

Воспользуемся изложенным выше методом. Выберем базис

$$\{1, \theta, \theta^2, \theta^3, \theta^4\},$$

где θ — корень неприводимого многочлена x^5+x^2+1 над полем $\mathrm{GF}(2)$. Пусть линейному многочлену $l_i(x) = x^{2^i}$ соответствует матрица C_i, где $i \in \{0, 1, 2\}$.

Непосредственной проверкой убеждаемся, что

$$C_0 = E, \quad C_1 = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 \end{pmatrix}, \quad C_2 = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{pmatrix}.$$

Тогда система (9) имеет вид

$$\left(x_1^{(1)}, ..., x_1^{(5)}, x_2^{(1)}, ..., x_2^{(5)}, x_3^{(1)}, ..., x_3^{(5)}\right) \begin{pmatrix} 0 & C_2 & 0 \\ E & 0 & C_2 \\ C_1 + C_2 & C_1 & C_1 \end{pmatrix} = (0, ..., 0)\,.$$

Решая ее, находим единственное ненулевое решение

$$\left(x_1^{(1)}, ..., x_1^{(5)}, x_2^{(1)}, ..., x_2^{(5)}, x_3^{(1)}, ..., x_3^{(5)}\right) = (00101, 00101, 01001)\,.$$

Таким образом, $W_1 = \{0, a\}$, где $a = (\theta^2 + \theta^4, \theta^2 + \theta^4, \theta + \theta^4)$.
Подставляя найденный вектор в уравнение $\Delta_a f(x) = 0$, убеждаемся, что

$$W_0 = W_{-1} = \{0\}.$$

Окончательно получаем

$$\left|(\mathrm{H}_3)_f^{(1)}\right| = 2,$$

$$\left|(\mathrm{H}_3)_f^{(0)}\right| = |(\mathrm{H}_3)_f| = 1.$$

ЛИТЕРАТУРА

1. *Черемушкин А. В.* Аффинная эквивалентность и ее применение при изучении свойств дискретных функций (обзор результатов) // Материалы Междунар. научн. конф. по проблемам безопасности и противодействия терроризму. Интеллектуальный центр МГУ. (2–3 ноября 2005 г.) М.: МЦМНО, 2006. С. 103–130.
2. *Лиддл Р., Нидеррайтер Г.* Конечные поля. Т. 1, 2. М.: Мир, 1988. 818 с.
3. *Черемушкин А. В.* Некоторые асимптотические оценки для класса сильно зависимых функций // Вестник Томского госуниверситета. Приложение. 2006. № 17. С. 87–94.

МАТЕМАТИЧЕСКИЕ МЕТОДЫ КРИПТОГРАФИИ

DOI 10.17223/20710410/8/5

УДК 004.421.5

ВЫСОКОСКОРОСТНЫЕ ГЕНЕРАТОРЫ ПСЕВДОСЛУЧАЙНЫХ ПОСЛЕДОВАТЕЛЬНОСТЕЙ НА ОСНОВЕ КЛЕТОЧНЫХ АВТОМАТОВ

Б. М. Сухинин

Московский государственный технический университет им. Н. Э. Баумана, г. Москва, Россия

E-mail: b.sukhinin@gmail.com

Псевдослучайные последовательности (ПСП) широко используются в различных областях науки и техники — от моделирования физических процессов и методов Монте-Карло до криптографии. В работе исследуется ряд свойств однородных двумерных булевых клеточных автоматов, а также разрабатывается новый генератор псевдослучайных последовательностей, основанный на использовании этих автоматов. Выходные последовательности таких генераторов имеют хорошие статистические свойства, а аппаратная реализация предложенных алгоритмов на типовых ПЛИС обладает очень высоким быстродействием — до 25 Гбит/с на частоте 100 МГц.

Ключевые слова: *генераторы псевдослучайных последовательностей, клеточные автоматы, лавинный эффект.*

Введение

Клеточным автоматом (КлА) называется модель с дискретным временем, состоящая из множества ячеек памяти, упорядоченных в периодическую n-мерную решетку [1]. Заполнения ячеек являются элементами некоторого конечного множества. Для каждой ячейки выбирается ее окрестность, которая используется для определения заполнения ячейки на следующем такте работы по некоторому заранее заданному правилу.

Для классических клеточных автоматов выполняются два свойства: однородности и локальности [2]. Однородность означает, что все ячейки КлА являются неразличимыми по своим свойствам; кроме того, для решения проблемы краевых клеток противоположные края решетки отождествляются, то есть двумерная решетка закручивается в тор. В соответствии со свойством локальности в окрестность каждой ячейки входят только ячейки, удаленные от нее на расстояние не более заданного.

Обширные исследования одномерных клеточных автоматов были проведены Стефаном Вольфрамом (см., например, [3 – 5]; исследование же КлА размерности 3 и более обладает ограниченным практическим интересом в силу сложности их реализации. Поэтому в настоящей работе рассматриваются двумерные булевы клеточные автоматы с прямоугольными ячейками. В таких автоматах заполнения ячеек памяти содержат двоичные значения; использование двоичного множества $\{0, 1\}$ и прямоугольная форма ячеек облегчают реализацию КлА и их применение в вычислительной технике. В качестве правила, определяющего новые заполнения ячеек на следующем

такте работы, используется булева функция, аргументами которой являются заполнения ячеек, входящих в окрестность данной; такую функцию будем называть локальной функцией связи. В качестве окрестности ячейки используем подмножество ячеек, непосредственно смежных с данной, а также, возможно, ее саму; использование более широкого множества увеличивает число аргументов локальной функции связи и делает ее реализацию непрактичной.

1. Свойства клеточных автоматов

1.1. Зависимость числа единичных заполнений ячеек от веса локальной функции связи

Локальная функция связи является основным параметром, определяющим особенности функционирования клеточного автомата. Важной задачей является изучение зависимости между характеристиками локальной функции связи и распределением заполнений ячеек решетки клеточного автомата.

Рассмотрим локальную функцию связи f, вектор значений которой имеет длину 2^s, где s — число аргументов функции (оно совпадает с мощностью окрестности ячейки). Пусть вес функции f, то есть число наборов аргументов, на которых функция принимает единичные значения, равен ω. Относительным весом функции назовем величину $\omega_0 = \omega/2^s$.

Предположим, что начальные заполнения ячеек решетки распределены случайно и равновероятно, то есть для произвольной ячейки m

$$\Pr[m = 0] = \Pr[m = 1] = \frac{1}{2}.$$

В таком случае все возможные двоичные наборы $\Psi(m)$ длины s, соответствующие заполнениям ячеек из окрестности m, также будут встречаться с равной вероятностью.

Поскольку вес функции f равен ω и все наборы аргументов равновероятны, вероятности того, что функция f принимает единичное или нулевое значение на наборе $\Psi(m)$, составляют

$$\Pr[f(\Psi(m)) = 1] = \omega/2^s = \omega_0,$$
$$\Pr[f(\Psi(m)) = 0] = 1 - \omega_0.$$

Напомним, что заполнение ячейки m на следующем такте работы клеточного автомата совпадает со значением локальной функции связи на наборе $\Psi(m)$. Из этого очевидным образом следует, что одинаковое количество единичных и нулевых заполнений ячеек достигается только при $\omega_0 = 1/2$, что соответствует равновесным локальным функциям связи.

На рис. 1 изображены графики временно́й зависимости отношения числа единичных заполнений к общему количеству ячеек КлА при различных весах локальной функции связи. Данные отражают усреднение для 1 000 различных клеточных автоматов с размерами решетки 37×11 ячеек и случайно выбранной локальной функцией связи от 9 аргументов (длина вектора значений 512). На графиках хорошо видно, что каждому значению относительного веса ω_0 соответствует быстрое приближение к некоторому стационарному значению числа единичных ячеек клеточного автомата.

1.2. Характеристики лавинного эффекта

Понятие лавинного эффекта было введено Хорстом Фейстелем [6] в 1973 г. для оценки свойств криптографических преобразований. Лавинный эффект показывает,

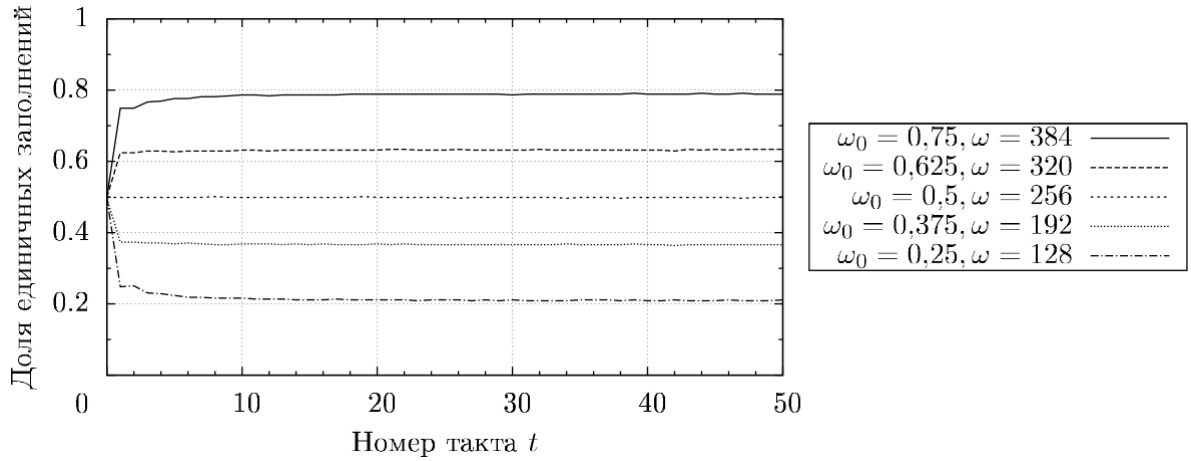

Рис. 1. Отношение числа единичных заполнений к общему количеству ячеек при различных весах локальной функции связи

насколько сильно изменяется выход некоторого преобразования при изменении одного бита входных данных.

Перед тем как перейти к описанию лавинного эффекта в двумерных клеточных автоматах, необходимо ввести несколько дополнительных понятий. Для этого рассмотрим клеточный автомат с решеткой размера $M_X \times M_Y$. Через $m_{(x,y)}$ обозначим ячейку с координатами (x, y). Поскольку решетка КлА представляет собой тор, вычисления координат осуществляются по модулю соответствующего размера решетки, то есть

$$m_{(x,y)} \equiv m_{((x \bmod M_X),(y \bmod M_Y))}.$$

В дальнейшем для упрощения записи операцию взятия остатка от деления будем опускать.

Введем понятие расстояния между ячейками клеточного автомата как максимальное абсолютное значение разности соответствующих координат. С учетом закручивания решетки КлА в тор расстояние $\Delta(m_{(x_1,y_1)}, m_{(x_2,y_2)})$ между ячейками $m_{(x_1,y_1)}$ и $m_{(x_2,y_2)}$ задается формулой

$$\Delta(m_{(x_1,y_1)}, m_{(x_2,y_2)}) = \max\big(\min(|x_1 - x_2|, M_X - |x_1 - x_2|), \min(|y_1 - y_2|, M_Y - |y_1 - y_2|)\big).$$

Очевидно, что максимально возможное расстояние между двумя ячейками КлА равно

$$\Delta_{\max} = \max\left(\left\lfloor \frac{M_X - 1}{2} \right\rfloor, \left\lfloor \frac{M_Y - 1}{2} \right\rfloor\right).$$

Рассмотрим два идентичных клеточных автомата, т. е. автоматы с одинаковыми размерами решетки $M_X \times M_Y$ (для определенности будем считать, что $M_X \geqslant M_Y$), одной и той же локальной функцией связи и одинаковыми заполнениями совпадающих по координатам ячеек. Обозначим через $m^t_{(x,y)}$ заполнение ячейки первого клеточного автомата с координатами (x, y) в момент времени t; для аналогичной ячейки второго клеточного автомата будем использовать обозначение $\widehat{m}^t_{(x,y)}$. В момент времени $t = 0$ изменим заполнение ячейки с координатами $(0, 0)$ второго клеточного автомата на противоположное:

$$\widehat{m}^0_{(0,0)} \leftarrow 1 - \widehat{m}^0_{(0,0)}$$

(поскольку в силу однородности все ячейки неразличимы по своим свойствам, выбор конкретной ячейки не ограничивает общности).

Лавинный эффект отражает распространение изменений, вызванных во втором клеточном автомате сменой заполнения одной ячейки памяти. Введём две числовые характеристики лавинного эффекта: интегральную и пространственную. Если изменения распространяются равномерно во всех направлениях с максимально возможной линейной скоростью (в данном случае составляющей одну ячейку за такт работы) и при этом изменяется заполнение половины всех ячеек, то такой лавинный эффект называется оптимальным.

Интегральной характеристикой лавинного эффекта $\eta(t)$ в клеточных автоматах назовем временну́ю зависимость отношения числа несовпадающих заполнений для ячеек с одинаковыми координатами к общему количеству ячеек в решетке:

$$\eta(t) = \sum_{\substack{0 \leqslant x < M_X \\ 0 \leqslant y < M_Y}} \frac{m^t_{(x,y)} \oplus \hat{m}^t_{(x,y)}}{M_X \cdot M_Y},$$

где сумма вычисляется как обычное арифметическое сложение, а операция $\oplus$ — как сложение по модулю 2. Интегральная характеристика оптимального лавинного эффекта имеет вид

$$\eta_{\text{opt}}(t) = \begin{cases} (2t+1)^2/(2 \cdot M_X \cdot M_Y), & 2t+1 \leqslant M_Y; \\ (2t+1)/(2 \cdot M_X), & M_Y < 2t+1 \leqslant M_X; \\ 1/2, & M_X < 2t+1. \end{cases}$$

Показателем, отражающим линейную скорость распространения изменений по решетке клеточного автомата, является пространственная характеристика лавинного эффекта $\mu(t)$, равная отношению максимального расстояния, на котором проявились изменения, к максимально возможному расстоянию:

$$\mu(t) = \frac{1}{\lfloor (M_X - 1)/2 \rfloor} \cdot \max_{\substack{0 \leqslant x < M_X \\ 0 \leqslant y < M_Y}} \left((m^t_{(x,y)} \oplus \hat{m}^t_{(x,y)}) \cdot \Delta(m_{(0,0)}, m_{(x,y)}) \right),$$

где $\Delta(m_{(0,0)}, m_{(x,y)})\big)$ — расстояние между ячейками с координатами $(0,0)$ и (x,y). Пространственная характеристика оптимального лавинного эффекта описывается формулой

$$\mu_{\text{opt}}(t) = \begin{cases} \dfrac{t}{\lfloor (M_X - 1)/2 \rfloor}, & t < \lfloor (M_X - 1)/2 \rfloor; \\ 1, & t \geqslant \lfloor (M_X - 1)/2 \rfloor. \end{cases}$$

Следует отметить, что лавинный эффект в конкретном клеточном автомате зависит от выбора начального заполнения ячеек решетки и локальной функции связи.

Таким образом, характеристики лавинного эффекта отражают свойства автомата в целом и должны рассматриваться как некоторый усредненный показатель. Кроме того, лавинный эффект существенно зависит от выбора окрестности, то есть от числа аргументов локальной функции связи. Из графиков характеристик лавинного эффекта, приведенных на рис. 2 и 3, видно, что характеристики приближаются к оптимальным по мере увеличения числа аргументов локальной функции связи; кроме того, графики для 8 и 9 аргументов являются практически идентичными, поэтому будем использовать локальные функции связи от 8 аргументов. Данные получены усреднением по 1 000 различным клеточным автоматам с размерами решетки 37×11 ячеек.

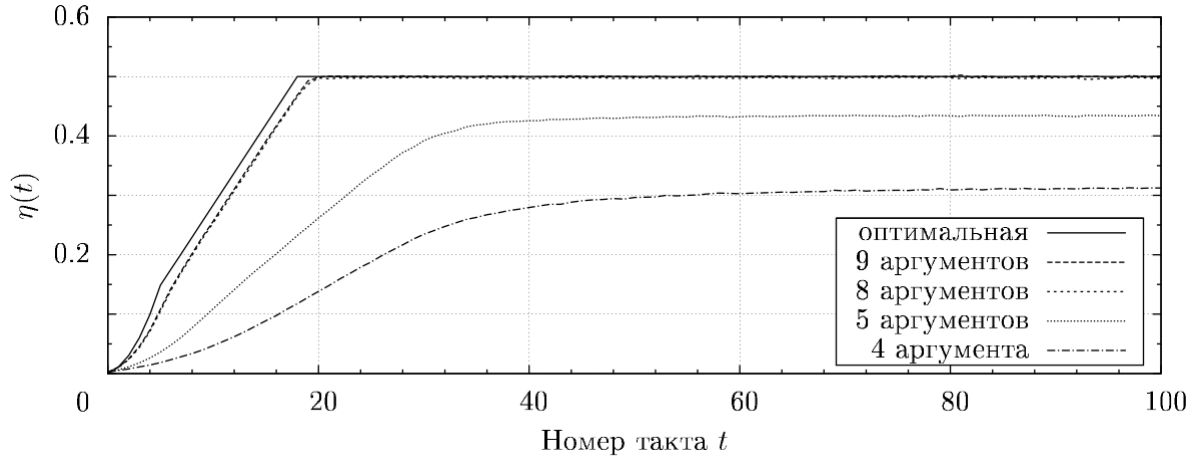

Рис. 2. Интегральная характеристика лавинного эффекта

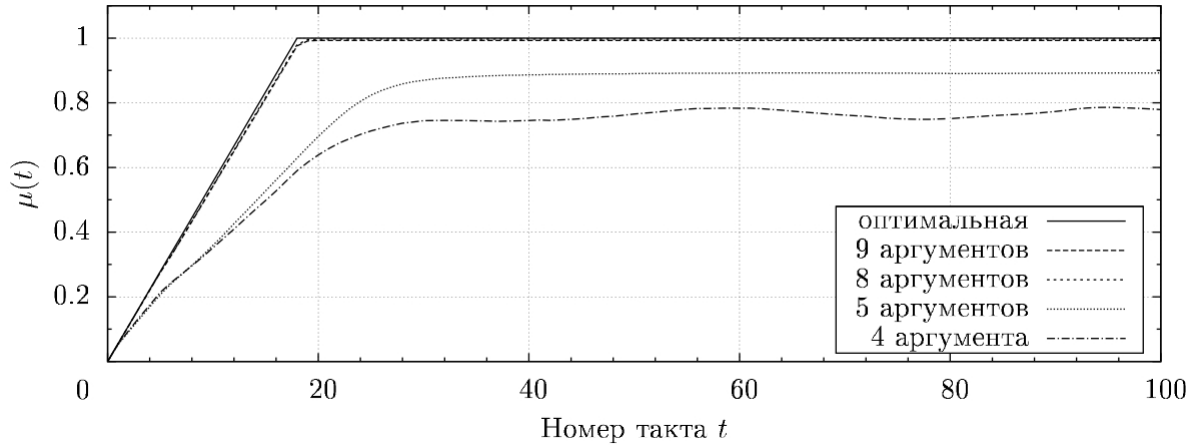

Рис. 3. Пространственная характеристика лавинного эффекта

1.3. Периодичность клеточных автоматов

Клеточные автоматы можно рассматривать как автономные конечные автоматы. Как и любой автономный конечный автомат, КлА обладают конечным периодом последовательности внутренних состояний, то есть заполнений ячеек решетки. В силу нелинейности локальной функции связи оценить период клеточного автомата не представляется возможным; тем не менее можно дать рекомендации по его увеличению.

Помимо классических временны́х периодов клеточных автоматов рассматриваем пространственные периоды, характеризующиеся следующим соотношением:

$$m_{(x,y)} = m_{((x+T_X),(y+T_Y))},$$

где $T_X \leqslant M_X$ и $T_Y \leqslant M_Y$ — величины пространственных периодов по горизонтали и вертикали соответственно. Очевидно, что для существования периода необходимо, чтобы его величина вдоль некоторой оси делила размер решетки вдоль той же оси. Для описания клеточных автоматов с установившимся пространственным периодом достаточно рассматривать подрешетку размера $T_X \times T_Y$; следствием этого является существенное снижение периода КлА.

На свойства периодичности также влияет структура связей клеточного автомата. На рис. 4 приведены графики пространственной характеристики лавинного эффекта, полученные в течение 300 тактов работы автомата. Колебания графика, соответствующего локальной функции связи с 4 аргументами, вызваны установившимся временным периодом. Кроме того, из рисунка видно, что с увеличением числа аргументов вероятность возникновения таких периодов резко сокращается.

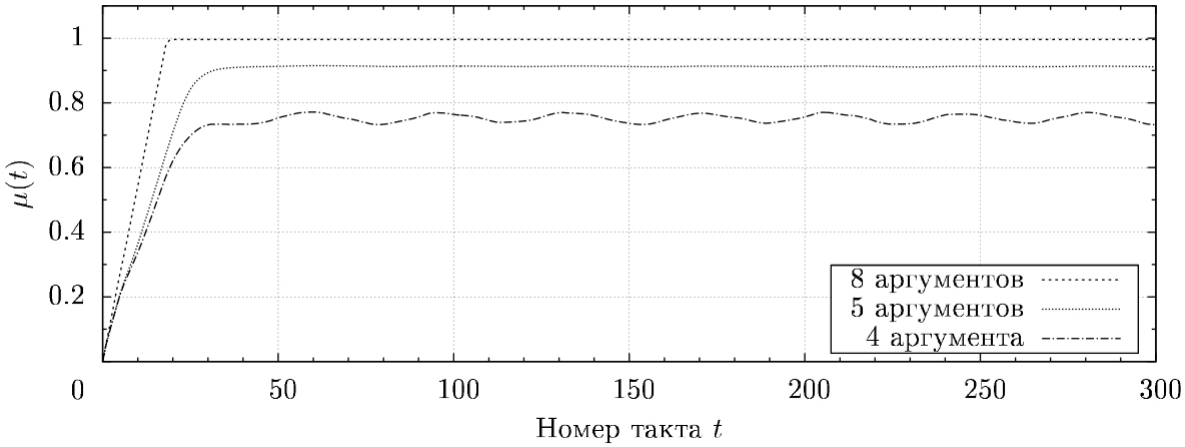

Рис. 4. Проявление периодичности на графике пространственной характеристики лавинного эффекта

2. Генераторы ПСП на основе клеточных автоматов

2.1. Структура генератора

Структура генератора ПСП, основанного на использовании однородных двумерных булевых клеточных автоматов, приведена на рис. 5. В состав генератора входят два клеточных автомата и регистр сдвига с линейной обратной связью (РСЛОС).

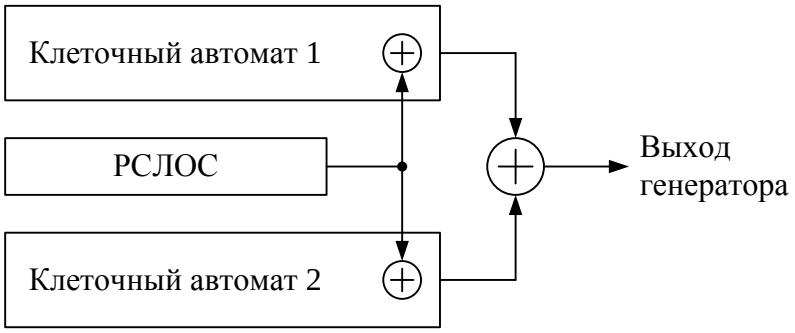

Рис. 5. Структура генератора ПСП на основе клеточных автоматов

Размеры решетки одинаковы для обоих автоматов и составляют 37×11 ячеек; выбор простых чисел позволяет избежать возникновения пространственных периодов. Окрестность каждой ячейки состоит из ячеек, непосредственно смежных с ней, что соответствует локальной функции связи от 8 аргументов. В качестве выхода клеточного автомата используются заполнения ячеек подрешетки размера 32×8, то есть ячеек $m_{(x,y)}$, где $0 \leqslant x < 32$ и $0 \leqslant y < 8$, что обеспечивает выработку каждым КлА 256 бит за один такт работы. Для каждого клеточного автомата используется своя собственная равновесная локальная функция связи.

Выход РСЛОС на каждом такте работы прибавляется по модулю 2 к заполнениям ячеек клеточных автоматов с координатами $(34, 9)$. Лавинный эффект позволяет гарантировать, что период внутренних состояний клеточных автоматов будет не меньше периода выходной последовательности РСЛОС. Считаем, что для практического применения генератора достаточно использовать РСЛОС длины 63, что обеспечивает период его выходной последовательности $2^{63} - 1 \approx 9{,}2 \cdot 10^{18}$ бит; период выходной последовательности КлА при этом составляет не менее $32 \cdot 8 \cdot (2^{63} - 1) \approx 2{,}4 \cdot 10^{21}$ бит. Тем не менее длина регистра может быть изменена при необходимости.

Выход генератора формируется посредством сложения по модулю 2 выходных последовательностей обоих клеточных автоматов, что позволяет существенно улучшить статистические свойства выходной последовательности генератора, увеличить ее период, а также затруднить восстановление внутреннего состояния генератора по выходным значениям.

2.2. Аппаратная реализация и статистические свойства генератора

Автором был разработан прототип аппаратной реализации предложенного генератора на ПЛИС Altera Cyclone II (EP2C35F672C6); структурная схема прототипа приведена на рис. 6. Выходная последовательность генератора подавалась напрямую на выводы микросхемы ПЛИС, а также записывалась для дальнейшего анализа во внутреннюю память. Параллельная структура клеточных автоматов позволила достичь вычисления нового состояния обоих КлА и формирования выхода генератора за один такт синхронизации схемы.

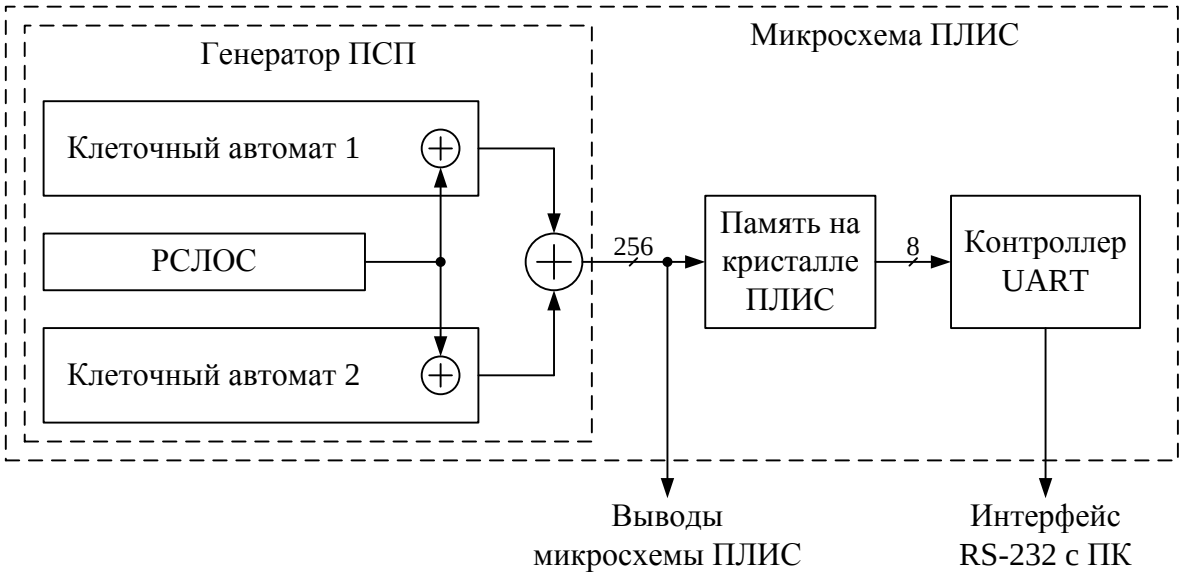

Рис. 6. Структурная схема прототипа аппаратной реализации генератора ПСП на ПЛИС

Рабочая частота схемы составила 100 МГц, причем статический анализ временных задержек показал, что она может быть повышена до 130 МГц без внесения каких-либо изменений. Учитывая, что на каждом такте работы генератор формирует 256 бит выходной последовательности, скорость ее выработки составила 23,8 Гбит/с.

Для исследования статистических свойств выходной последовательности использовался набор тестов NIST [7], включающий в себя 15 разновидностей проверок, направленных на выявление различных статистических отклонений исследуемой последовательности от истинно случайной. Следует отметить, что набор NIST предназначен для

тестирования криптографических генераторов псевдослучайных последовательностей, то есть в нем предъявляются наиболее жесткие требования. Тестирование генераторов с различными локальными функциями связи клеточных автоматов позволило обнаружить функции, при которых генератор успешно проходит все тесты из набора. Для сокращенной версии генератора, в которой один из двух клеточных автоматов отключен и не вырабатывает выходную последовательность, таких функций обнаружено не было. Тем не менее следует учитывать, что сокращенный генератор может использоваться в областях, где предъявляются менее жесткие требования к статистическим свойствам ПСП.

Заключение

В работе исследованы некоторые свойства однородных двумерных булевых клеточных автоматов и предложена структура генератора псевдослучайных последовательностей, основанного на использовании КлА. Выходные последовательности такого генератора успешно проходят набор статистических тестов NIST, предъявляющий наиболее жесткие требования к генераторам ПСП. Разработанный прототип аппаратной реализации генератора обеспечивает выработку выходной последовательности на сверхвысокой скорости до 25 Гбит/с.

Необходимо отметить, что использование клеточных автоматов предоставляет большое поле для исследований. В настоящее время ведется работа над программной реализацией генератора, использующей в качестве вычислительного устройства графический адаптер ПЭВМ, что позволит говорить о возможности массового применения подобных алгоритмов. Кроме того, одним из объектов исследований являются неоднородные клеточные автоматы, в которых окрестность каждой ячейки выбирается случайным образом, но при этом является фиксированной в течение работы КлА. Такие клеточные автоматы обладают существенно лучшими характеристиками по сравнению с рассмотренными выше, а также позволяют строить намного более эффективные реализации.

ЛИТЕРАТУРА

1. *Farmer D., Toffoli T., Wolfram S.* Preface to Cellular Automata // Proceedings of an Interdisciplinary Workshop. Los Alamos, New Mexico, 1984. P. vii–xii.
2. *Тоффоли Т., Марголус Н.* Машины клеточных автоматов. М.: Мир, 1991. 280 с.
3. *Wolfram S.* A New Kind of Science. Wolfram Media, 2002. 1192 p.
4. *Wolfram S.* Cellular Automata // Los Alamos Science. 1983. No. 9. P. 2–21.
5. *Wolfram S.* Cryptography with Cellular Automata // Proceedings of CRYPTO'85. 1986. P. 429–432.
6. *Feistel H.* Cryptography and Computer Privacy // Scientific American. 1973. V. 228. No. 5. P. 15–23.
7. `http://csrc.nist.gov/publications/nistpubs/800-22-rev1/SP800-22rev1.pdf` — NIST SP 800-22. A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications, revision 1.

МАТЕМАТИЧЕСКИЕ МЕТОДЫ СТЕГАНОГРАФИИ

DOI 10.17223/20710410/8/6

УДК 004.94

МОДИФИКАЦИЯ КАРТЫ ХОТЕЛЛИНГА, НИВЕЛИРУЮЩАЯ ВЛИЯНИЕ ТРЕНДА, И ЕЕ ПРИМЕНЕНИЕ ПРИ ОБНАРУЖЕНИИ ЦИФРОВЫХ ВОДЯНЫХ ЗНАКОВ

Б. Б. Борисенко

Институт проблем информационной безопасности МГУ, г. Москва, Россия

E-mail: fepem@yandex.ru

Предлагается использовать модифицированную статистику Хотеллинга в контрольной карте с целью исключения влияния тренда. Рассматривается вариант применения метода контрольных карт в задаче обнаружения наличия цифровых водяных знаков.

Ключевые слова: *компьютерная безопасность, цифровые водяные знаки, карты контроля качества.*

Для защиты авторских прав на электронные документы используются цифровые водяные знаки (ЦВЗ). Под ЦВЗ принято понимать специальную метку, встраиваемую в цифровой контент с целью аутентификации и защиты от копирования. Большинство современных методов обнаружения ЦВЗ используют в своей работе принцип обучения [1 – 7]. При этом обучение проводится как на выборке пустых (немаркированных) контейнеров, так и на выборке контейнеров с ЦВЗ. И те и другие генерируются аналитиком самостоятельно. Для создания пустых контейнеров целесообразно использовать фотоаппараты без функции внесения собственного ЦВЗ. Для классификации контейнеров используется, как правило, дискриминантный анализ (ДА) или метод опорных векторов (SVM — Support Vector Machine). Однако для построения выборки контейнеров с ЦВЗ *необходимо знать конкретный метод внедрения ЦВЗ* (или хотя бы класс методов), что не всегда является возможным.

Предлагаемый метод обнаружения ЦВЗ *не требует априорного знания метода внедрения цифровых водяных знаков*. Обучение производится только на выборке пустых контейнеров. При этом для каждого тестируемого контейнера на обучающую выборку накладываются определенные требования. Так, например, для изображений объекты обучающей выборки должны максимально походить на тестируемый контейнер, то есть принадлежать к одному типу изображений (пейзаж, средний план, поле), иметь примерно одинаковую гистограмму цветов и одинаковый коэффициент качества (процент сжатия). Поэтому особенностью предлагаемого метода является необходимость создания обучающей выборки в худшем случае для каждого тестируемого мультимедиа-контейнера (если тестируемые контейнеры одного типа, можно использовать одну и ту же обучающую выборку).

Метод основан на отслеживании поведения некоторых заранее определенных признаков контейнеров. От выбора конкретного набора признаков зависит результат работы метода (большая вероятность обнаружения ЦВЗ, меньшая вероятность ошибок

первого и второго рода), но не принцип работы самого метода. Набор признаков выбирается в зависимости от типа (графика, видео, звук) и формата контейнера.

Суть предлагаемого метода заключается в применении элементов метода контрольных карт [8, 9], используемого для выявления производственных сбоев, адаптированного под использование задач обнаружения цифровых водяных знаков. При этом:

— на основе обучающей выборки пустых контейнеров вычисляются контрольные границы для классификации. Пороги находятся из распределения Фишера или подбором по методу Монте-Карло;
— на основе признаков тестируемого контейнера и обучающей выборки пустых контейнеров вычисляется значение модифицированной статистики Хотеллинга для последующей классификации;
— значение статистики сравнивается с вычисленными порогами и выдается результат о наличии или отсутствии ЦВЗ.

Рассмотрим подробнее этапы предлагаемого метода, а также возможность проведения анализа без обучения на конкретном методе сокрытия. Для лучшего понимания приведем основы применения контрольных карт в производстве.

1. Методы статистического контроля технологического процесса

В связи с существенным ростом актуальности ряда практических задач, таких, как автоматическое обнаружение неисправностей, обслуживание оборудования на основе автоматического контроля его состояния, обеспечение безопасности сложных технических и информационных систем, автоматический контроль качества выпускаемой продукции, предсказание естественных катастрофических явлений, мониторинг в биомедицине и финансовой сфере, растет и необходимость выявления момента резкого изменения (*разладки*) некоторых вероятностных характеристик (признаков) у наблюдаемых процессов.

Статистический контроль технологического процесса представляет задачу последовательного обнаружения нарушений в независимой случайной последовательности при неизвестном априорном распределении момента времени, в который произошло нарушение. Вмешательство в технологический процесс для настройки требуется тогда, когда изделие еще удовлетворяет требованиям, но статистические показатели процесса свидетельствуют о наличии неслучайных воздействий. Практический инструмент для решения вопроса о необходимости такого вмешательства — *контрольная карта* — был предложен У. Шухартом [8, 9]: сигнал о разладке процесса подается при выходе контролируемого показателя за некоторую пороговую границу.

Задача последовательного обнаружения изменения среднего уровня технологического процесса для одномерной независимой гауссовской случайной последовательности рассмотрена в работах Е. Пейджа [10, 11]; на основе методов последовательного анализа была предложена контрольная карта кумулятивных сумм (CUSUM — Cumulative Sums). С. Робертсом [12, 13] предложено использование экспоненциального сглаживания для обнаружения нарушений в ходе процесса (метод экспоненциально взвешенных скользящих средних, EWMA — Exponentially Weighted Moving Average). Задача о статистическом контроле процесса при известных вероятностях перехода из налаженного состояния в разлаженное исследована А. Н. Ширяевым [14, 15].

Таким образом, используется три основных подхода к решению задачи статистического контроля процесса и различные их модификации:

— контрольная карта Шухарта (базируется на критерии Неймана — Пирсона);

— контрольные карты кумулятивных сумм (многократное применение последовательного анализа Вальда [16]);
— контрольные карты экспоненциально взвешенных скользящих средних (экспоненциальное сглаживание).

Обобщение контрольных карт Шухарта для независимой последовательности многомерных случайных векторов предложено Г. Хотеллингом [22]. Различные варианты обобщений для многомерного контроля алгоритмов CUSUM и EWMA предложены в [26 – 29].

Общий подход к контролю качества продукции достаточно прост. В процессе производства проводятся выборки изделий заданного объема. После этого строятся диаграммы (карты контроля качества) изменения выборочных значений параметра изделия (показателя качества). Ход процесса считается удовлетворительным, если найденные значения лежат в некоторых заранее заданных пределах. То есть в каждый момент времени проверяется гипотеза H_0 (процесс статистически управляем) при альтернативной гипотезе H_1 о статистической неуправляемости.

Если оказывается, что выборочные значения находятся на одной из контрольных границ или за ее пределами, то нулевая гипотеза отклоняется и считается, что процесс вышел из-под контроля (произошла разладка), предпринимаются необходимые действия для того, чтобы найти причину его разладки [17, 18]. Обычно контролируется как изменение среднего значения показателя качества, характеризующего уровень настройки процесса, так и изменение технологического рассеивания.

2. Карта Хотеллинга

Качество изделия обычно характеризуется несколькими показателями, которые могут быть коррелированы между собой. В этом случае независимый контроль по отдельным показателям может привести к значительным погрешностям вследствие различия доверительных областей и невозможности определения совместного уровня значимости [21]. В такой ситуации применяют многомерные контрольные карты. Впервые контроль качества с использованием нескольких характеристик был предложен Хотеллингом в [22]. Рассмотрим типы многомерных контрольных карт. Более подробный обзор многомерных карт можно найти, например, в [20, 28].

Предположим, что в технологическом процессе контролируются p показателей качества $\mathbf{X} = (X_1, X_2, \ldots, X_p)$, имеющих совместное нормальное распределение. Плотность распределения равна

$$f(\mathbf{X}) = (2\pi)^{-\frac{p}{2}} |\Sigma|^{-\frac{1}{2}} e^{\frac{-(\mathbf{X}-\mu)^{\mathrm{T}} \Sigma^{-1} (\mathbf{X}-\mu)}{2}}, \tag{1}$$

где μ — вектор средних значений; Σ — ковариационная матрица, элементы которой $\sigma_{ii} = {\sigma_i}^2$ — дисперсии случайных величин X_i, а $\sigma_{ij} = \sigma_{ji} = \rho_{ij}\sigma_i\sigma_j$ — ковариации (ρ_{ij} — коэффициент корреляции между величинами X_i и X_j).

Для проверки гипотезы $H_0\colon \mu = \mu_0$ в многомерном варианте применяется обобщенная статистика Хотеллинга [23, 24]:

$$T^2 = n(\bar{\mathbf{X}} - \mu_0)^{\mathrm{T}} S^{-1} (\bar{\mathbf{X}} - \mu_0), \tag{2}$$

где S — выборочная оценка ковариационной матрицы Σ. Применение контрольной карты Хотеллинга предполагает расчет для каждой t-й мгновенной выборки ($t = 1, \ldots, m$) статистики ${T_t}^2$ по формуле (2). При нормальном ходе процесса должно выполняться условие ${T_t}^2 < T^2_{\text{кр}}$, где $T^2_{\text{кр}}$ — граница критической области.

Многомерная контрольная карта Хотеллинга, по существу, есть та же карта Шухарта, в которой в качестве контролируемой величины используется обобщенная статистика Хотеллинга. Обратим внимание на то, что эта карта имеет только верхнюю контрольную границу.

Если ковариационная матрица Σ известна, статистика Хотеллинга имеет распределение χ^2. В этом случае положение контрольной границы на заданном уровне значимости α определяется по таблицам квантилей этого распределения $T^2_{\text{кр}} = \chi^2_{1-\alpha}(p)$ [25]. При неизвестной ковариационной матрице статистика

$$F = \frac{n-p}{p(n-1)} T^2 \tag{3}$$

имеет нецентральное F-распределение Фишера с p и $(n-p)$ степенями свободы и параметром нецентральности

$$\lambda^2 = n(\mu - \mu_0)^{\text{т}} \Sigma^{-1} (\mu - \mu_0). \tag{4}$$

При этом статистика T^2 имеет распределение Хотеллинга (центральное), плотность которого равна

$$f(x) = \frac{\Gamma\left(\frac{n+1}{2}\right) x^{\frac{p}{2}-1} \left(1 + \frac{x}{n}\right)^{-\frac{n+1}{2}}}{\Gamma\left(\frac{n-p+1}{2}\right) \Gamma\left(\frac{p}{2}\right) n^{\frac{p}{2}}}, x > 0. \tag{5}$$

В одномерном случае $p = 1$ и распределение Хотеллинга совпадает с квадратом распределения Стьюдента.

Карта Хотеллинга применима только в условиях совместной нормальности распределения контролируемых показателей. В многомерном случае практически полезным подходом при нарушении нормальности является преобразование (нормализация) данных таким образом, чтобы этот преобразованный набор имел хотя бы приблизительно нормальное распределение [21].

3. Анализ эффективности методов контроля качества в многомерном случае

Вопросы одномерного контроля качества изучены достаточно глубоко, в то время как исследования по многомерному статистическому контролю носят разрозненный характер и не позволяют обеспечить надежный контроль при проведении технологического процесса с коррелированными показателями качества. Чаще и в отечественной [19], и в зарубежной практике многомерный контроль подменяется независимым контролем нескольких показателей.

В случае некоррелированности показателей качества при независимом контроле с использованием соответствующего количества карт Шухарта в [21] было отмечено, что с увеличением числа показателей средняя длина серий несколько уменьшается. Если контролируемые показатели оказываются коррелированными, использование независимого контроля отдельных показателей может привести к серьезным ошибкам [30], связанным с двумя обстоятельствами.

Во-первых, различны доверительные области (acceptance region): при независимом контроле это прямоугольный параллелепипед, стороны которого определяются границами регулирования карт Шухарта. С учетом корреляционных связей в действительности доверительная область при многомерном нормальном распределении показателей представляет эллипсоид, главные оси которого повернуты относительно осей параллелепипеда. Точки, оказывающиеся внутри параллелепипеда, но вне эллипсоида,

свидетельствуют о нормальном ходе процесса, хотя на самом деле процесс статистически неуправляем.

Во-вторых, вычисление совместного уровня значимости невозможно при контроле по отдельным показателям, коррелированным между собой.

Один из возможных вариантов решения проблемы — переход от зависимых показателей к статистически независимым с использованием метода главных компонент [31]. Такое преобразование позволит, во-первых, обеспечить переход к некоррелированным показателям качества. Во-вторых, при определенных условиях после перехода к главным компонентам размерность задачи может снизиться, если у части последних главных компонент будут достаточно малые дисперсии.

Контрольная граница карты Хотеллинга в зависимости от конкретных условий определяется с использованием квантилей распределений χ^2 или Фишера с учетом количества контролируемых показателей, объема мгновенной выборки и количества выборок. При сравнении карт Шухарта на главных компонентах и карт Хотеллинга в [21] отмечается, что при некоррелированных показателях качества средняя длина серий в картах Шухарта меньше. Однако при коррелированных показателях, несмотря на уменьшение средней длины серий для обеих карт, степень уменьшения ARL с увеличением степени коррелированности для карт Хотеллинга гораздо выше. Среди недостатков карт Шухарта на главных компонентах отмечают также и невозможность во многих случаях правильной интерпретации данных, так как преобразование приводит к частичной потере первоначальной информации [32, 33].

Построение других многомерных карт — MCUSUM и MEWMA — является гораздо более трудоемким процессом. К тому же было установлено, что карты MCUSUM и MEWMA в случае больших сдвигов являются менее эффективными, чем карты Хотеллинга [34]. Накопление информации в таких картах происходит вследствие отслеживания истории процесса. Однако чаще всего производится анализ на наличие ЦВЗ абсолютно независимых друг от друга изображений, и, стало быть, для решения поставленных в работе задач необходимость в ведении истории процесса отпадает.

Таким образом, из анализа применения различных контрольных карт можно сделать вывод, что основным инструментом многомерного статистического контроля технологического процесса с коррелированными показателями качества является карта Хотеллинга.

Кроме выхода за границу регулирования к признакам нестабильности процесса также относятся и другие неслучайные структуры; в частности, в [19] упомянут *тренд процесса* — монотонное возрастание или убывание нескольких точек подряд. В одномерных картах Шухарта своевременно обнаружить неслучайные структуры позволяют предупреждающие линии (warning lines), впервые предложенные в [11]. Различные критерии серий с использованием таких линий были исследованы, в частности, в [35]. В картах Хотеллинга критерии нестабильности многомерного процесса вследствие появления тренда *отсутствуют* [21]. В связи с этим при наличии тренда карта Хотеллинга работает менее эффективно (возрастает значение ложных срабатываний, снижается вероятность объявления тревоги).

Поэтому для задач обнаружения ЦВЗ целесообразно разработать модификацию карты Хотеллинга, которая позволила бы повысить вероятность своевременного обнаружения разладки при наличии тренда.

4. Разработка модификации карты Хотеллинга, нивелирующей влияние тренда

На основе анализа применения многомерных контрольных карт был сделан вывод о том, что наиболее эффективным инструментом многомерного контроля качества является карта Хотеллинга. Однако возникает необходимость нивелировать влияние возможного тренда в одном или нескольких показателях (признаках), который может появиться, в частности, в результате незначительного изменения изображения (поворот на небольшой угол, небольшой сдвиг фотоаппарата при панорамной съемке, небольшое изменение яркости/цветности и т. п.). Также, если говорить об обнаружении ЦВЗ в изображениях, небольшое изменение признака, вызванное такой незначительной естественной модификацией изображения, может повлиять на решение анализатора (контрольной карты). В этом случае возрастает количество ложных тревог.

В соответствии с этим целесообразно выделить следующие направления исследования:

1) разработка модификации карты Хотеллинга, не чувствительной к тренду (в этом случае контрольная карта не считает тренд причиной нестабильности процесса);
2) разработка алгоритма применения модификации карты Хотеллинга к обнаружению ЦВЗ;
3) проведение экспериментов и сравнение результатов.

Статистика в карте Хотеллинга вычисляется по формуле (2), где ковариационная матрица обычно оценивается формулой

$$S = \frac{1}{(n-1)} \sum_{i=1}^{n} (\mathbf{x}_i - \bar{\mathbf{x}})(\mathbf{x}_i - \bar{\mathbf{x}})^{\mathrm{T}}. \tag{6}$$

При тренде такая оценка ковариационной матрицы будет неточной, так как тренд, изменяя значения векторов, будет изменять и значение среднего. Это может привести к ухудшению контроля качества. Следовательно, целесообразно применить другую оценку ковариационной матрицы. Кроме того, в случае анализа изображения на предмет наличия ЦВЗ использование мгновенных выборок для контроля зачастую невозможно (есть одно изображение, которое необходимо проанализировать). В этом случае приходится использовать результаты индивидуальных наблюдений, и для получения несмещенных оценок компонент ковариационной матрицы как при одномерном, так и при многомерном контроле применяются скользящие размахи [36, 37]. Также применение размахов в (9) позволяет дать более точную оценку ковариационной матрицы при наличии тренда в силу того, что в изображениях, содержащих некоторый локальный тренд и шумовую составляющую, ковариация между приращениями различных компонент остается постоянной (исключается влияние тренда) в предположении постоянства коэффициента корреляции между компонентами шумовой составляющей.

Для аппроксимации распределения статистики модифицированной карты одним из известных распределений потребуется нормирующий множитель.

Пусть $\mathbf{x}_1, \ldots, \mathbf{x}_n$ — выборка независимых в совокупности одинаково распределенных векторов, где $\mathbf{x}_i \in \mathbb{R}^d$; $\mathbf{x}_i \sim \mathcal{N}(\mu, \Sigma)$; $\mu \in \mathbb{R}^d$ — вектор математических ожиданий; $\Sigma \in \mathbb{R}^{d \times d}$ — ковариационная матрица.

Обозначим через $X \in \mathbb{R}^{n \times d}$ матрицу, составленную из векторов $\mathbf{x}_i^{\mathrm{T}}$, $i = 1, \ldots, n$, то есть $X^{\mathrm{T}} = (\mathbf{x}_1, \ldots, \mathbf{x}_n)$. Положим $\mathbf{z}_i = \mathbf{x}_{i+1} - \mathbf{x}_i$, где $i = 1, \ldots, n-1$.

Очевидно, что $Z = C \cdot X$, где

$$Z = \begin{pmatrix} \mathbf{z}_1^{\mathrm{T}} \\ \vdots \\ \mathbf{z}_{n-1}^{\mathrm{T}} \end{pmatrix} = \begin{pmatrix} \mathbf{x}_2^{\mathrm{T}} - \mathbf{x}_1^{\mathrm{T}} \\ \vdots \\ \mathbf{x}_n^{\mathrm{T}} - \mathbf{x}_{n-1}^{\mathrm{T}} \end{pmatrix}, \tag{7}$$

$$C = \begin{pmatrix} -1 & 1 & 0 & 0 & \cdots & 0 & 0 \\ 0 & -1 & 1 & 0 & \cdots & 0 & 0 \\ 0 & 0 & -1 & 1 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & -1 & 1 \end{pmatrix} \in \mathbb{R}^{(n-1)\times n}. \tag{8}$$

В дальнейшем, не ограничивая общности, будем предполагать, что $\mu = \mathbf{0}$.
Введем следующие обозначения:

$$S_n = \frac{1}{2(n-1)} \sum_{i=1}^{n-1} \mathbf{z}_i \cdot \mathbf{z}_i^{\mathrm{T}}; \tag{9}$$

$$\bar{x}_n = \frac{1}{n} \sum_{i=1}^{n} \mathbf{x}_i; \tag{10}$$

$$f(n) = \frac{2(n-1)^2}{3n-4}; \tag{11}$$

$$T_n = \frac{f(n) - d + 1}{f(n) \cdot d} \cdot \frac{n}{n+1} (\mathbf{x} - \bar{x}_n)^{\mathrm{T}} S_n^{-1} (\mathbf{x} - \bar{x}_n), \tag{12}$$

где $x \in \mathbb{R}^d$; $\mathbf{x} \sim \mathcal{N}(\mu, \Sigma)$ — вектор, отличный от векторов выборки $\mathbf{x}_1, \ldots, \mathbf{x}_n$.

Замечание 1. Если объем выборки n невелик по сравнению с d, то матрица S_n может оказаться вырожденной. В таком случае для подсчета статистики T_n можно поступить следующим образом:

1) снизить размерность векторов выборки $\mathbf{x}_1, \ldots, \mathbf{x}_n$ при помощи метода главных компонент, оставив компоненты с большей дисперсией [38]. Получим новую выборку $\tilde{\mathbf{x}}_1, \ldots, \tilde{\mathbf{x}}_n$;
2) подсчитать статистику T_n по выборке $\tilde{\mathbf{x}}_1, \ldots, \tilde{\mathbf{x}}_n$.

Лемма 1. При сделанных выше предположениях

$$S_n = \frac{1}{n-1} \cdot X^{\mathrm{T}} A X, \tag{13}$$

где $A = \frac{C^{\mathrm{T}} \cdot C}{2}$.

Доказательство. Имеем

$$S_n = \frac{1}{2(n-1)} \sum_{i=1}^{n-1} \mathbf{z}_i \cdot \mathbf{z}_i^{\mathrm{T}} = \frac{1}{2(n-1)} Z^{\mathrm{T}} Z.$$

Пусть $A = (C^{\mathrm{T}} \cdot C)/2$. Тогда из равенства $Z = C \cdot X$ следует, что
$S_n = \frac{1}{2(n-1)} (CX)^{\mathrm{T}} CX = \frac{1}{2(n-1)} X^{\mathrm{T}} (C^{\mathrm{T}} C) X = \frac{1}{n-1} X^{\mathrm{T}} A X$. ■

Лемма 2. S_n — несмещенная и состоятельная оценка матрицы Σ.

Доказательство.

1) Верна цепочка равенств

$$\mathsf{E}S_n = \frac{1}{2(n-1)}\sum_{i=1}^{n-1}\mathsf{E}(z)_i\mathbf{z}_i^{\mathrm{T}} = \frac{1}{2(n-1)}\sum_{i=1}^{n-1}\mathsf{E}(\mathbf{x}_{i+1}-\mathbf{x}_i)(\mathbf{x}_{i+1}-\mathbf{x}_i)^{\mathrm{T}} =$$

$$= \frac{1}{2(n-1)}\sum_{i=1}^{n-1}\mathsf{E}\mathbf{x}_{i+1}\mathbf{x}_{i+1}^{\mathrm{T}} - \mathbf{x}_i\mathbf{x}_{i+1}^{\mathrm{T}} + \mathbf{x}_i\mathbf{x}_i^{\mathrm{T}} - \mathbf{x}_{i+1}\mathbf{x}_i^{\mathrm{T}} =$$

$$= \frac{1}{2(n-1)}\sum_{i=1}^{n-1}(\mathsf{E}\mathbf{x}_{i+1}\mathbf{x}_{i+1}^{\mathrm{T}} - \mathsf{E}(\mathbf{x}_i\mathbf{x}_{i+1}^{\mathrm{T}}) + \mathsf{E}(\mathbf{x}_i x_i^{\mathrm{T}}) - \mathsf{E}(\mathbf{x}_{i+1}\mathbf{x}_i^{\mathrm{T}})) = \frac{1}{2(n-1)}\sum_{i=1}^{n-1}(\Sigma+\Sigma) = \Sigma,$$

где $\mathsf{E}(\mathbf{x}_i x_{i+1}^{\mathrm{T}}) = \mathsf{E}(\mathbf{x}_{i+1}\mathbf{x}_i^{\mathrm{T}}) = 0$, так как $\mathbf{x}_i$ и $\mathbf{x}_{i+1}$ — независимые случайные векторы.

2) Рассмотрим произвольный элемент s_{ij}^n, $i = 1,\ldots,d$, $j = 1,\ldots,d$, матрицы S_n. Здесь z_p^i и x_p^i — i-е координаты векторов $\mathbf{z}_p$ и $\mathbf{x}_p$ соответственно:

$$s_{ij}^n = \frac{1}{2(n-1)}\sum_{p=1}^{n-1} z_p^i z_p^j \ ;$$

$$\mathsf{D}(s_{ij}^n) = \frac{1}{4(n-1)^2}\mathsf{D}\left(\sum_{p=1}^{n-1} z_p^i z_p^j\right) = \frac{1}{4(n-1)^2}\left(\sum_{p=1}^{n-1}\mathsf{D}z_p^i z_p^j + 2\cdot\sum_{1\leqslant p<q\leqslant n-1}\operatorname{cov}(z_p^i z_p^j, z_q^i z_q^j)\right) =$$

$$= \frac{1}{4(n-1)^2}\sum_{p=1}^{n-1}\mathsf{D}z_p^i z_p^j + \frac{1}{4(n-1)^2}\cdot 2\cdot\sum_{1\leqslant p<q\leqslant n-1}\operatorname{cov}(z_p^i z_p^j, z_q^i z_q^j) = I + II \ ;$$

$$I = \frac{1}{4(n-1)^2}\left(\sum_{p=1}^{n-1}\mathsf{D}(x_{p+1}^i - x_p^i)(x_{p+1}^j - x_p^j)\right) =$$

$$= \frac{1}{4(n-1)^2}\left(\sum_{p=1}^{n-1}\mathsf{D}(x_{p+1}^i x_{p+1}^j - x_{p+1}^i x_p^j - x_p^i x_{p+1}^j + x_p^i x_p^j)\right) = O\left(\frac{1}{n}\right),$$

так как $\mathsf{D}(x_p^i x_p^j) < \infty$ и $\mathsf{D}(x_{p+1}^i x_p^j) < \infty$.

В связи с тем, что $z_p^i z_p^j = (x_{p+1}^i - x_p^i)(x_{p+1}^j - x_p^j)$, имеем

$$\operatorname{cov}(z_p^i z_p^j, z_q^i z_q^j) = \begin{cases} 0 & \text{при } p < q-1; \\ \neq 0 & \text{при } p = q-1. \end{cases}$$

Тогда

$$II = \frac{1}{4(n-1)^2}\cdot 2\sum_{p=1}^{n-2}\operatorname{cov}(z_p^i z_p^j, z_{p+1}^i z_{p+1}^j) = O\left(\frac{1}{n}\right),$$

так как $\operatorname{cov}(z_p^i z_p^j, z_{p+1}^i z_{p+1}^j) < \infty$.

Поскольку $\mathsf{D}(s_{ij}^n) \to 0$ при $n \to \infty$ и $\mathsf{E}S_n = \Sigma$, то S_n — состоятельная оценка ковариационной матрицы. ■

В [39] предлагается способ оценки компонент ковариационной матрицы через скользящие размахи.

В многомерном случае основная идея аппроксимации Welch — Satterthwaite [40, 41] может быть использована для того, чтобы показать, что распределение матрицы S_n может быть аппроксимировано с приемлемой точностью распределением Уишарта $W_d(f(n),\Sigma)$ [42, 43]. Возможность того, что распределение S_n может быть аппроксимировано распределением Уишарта $W_d(f(n),\Sigma)$ в многомерном случае, вытекает из следующего результата.

Теорема 1 [42]. Пусть $(\mathbf{y}_1, \ldots, \mathbf{y}_n)$ — выборка независимых в совокупности одинаково распределенных векторов $\mathbf{y}_i \in \mathbb{R}^d$, $\mathbf{y}_i \sim \mathcal{N}(\mathbf{0}, \Sigma_{\text{y}})$. Обозначим $Y = \begin{pmatrix} \mathbf{y}_1^{\text{т}} \\ \vdots \\ \mathbf{y}_n^{\text{т}} \end{pmatrix} \in \mathbb{R}^{n \times d}$ и положим $\xi = Y \cdot \eta$, где $\eta \in \mathbb{R}^d$ — произвольный вектор. Пусть также $B \in \mathbb{R}^{n \times n}$ — произвольная симметричная матрица ранга r. В таком случае

$$Y^{\text{т}} B Y \sim W_d(r, \Sigma_y) \tag{14}$$

тогда и только тогда, когда

$$\xi^{\text{т}} B \xi \sim \sigma_\eta^2 \chi_r^2 \tag{15}$$

для любого $\eta \in \mathbb{R}^d$, где $\sigma_\eta^2 = \eta^{\text{т}} \Sigma_y \eta$, а χ_r^2 — распределение хи-квадрат с r степенями свободы.

Теорема 2. При сделанных выше предположениях квантили распределения статистики T_n и квантили случайной величины с распределением Фишера с $(d, f(n)-d+1)$ степенями свободы отличаются не более чем на $O\left(\dfrac{1}{f(n)}\right)$, где $f(n) = O(n)$ при $n \gg 1$.

Доказательство. Обозначим $\xi_i = \mathbf{x}_i^{\text{т}} \eta$, где $\eta \in \mathbb{R}^d$ — произвольный постоянный вектор, а $\mathbf{x}_i$ — вектор из исходной выборки.

Очевидно, что $\xi_i \sim \mathcal{N}(0, \sigma_\eta^2)$, где $\sigma_\eta^2 = \eta^{\text{т}} \Sigma \eta$. Пусть

$$\xi = \begin{pmatrix} \xi_1 \\ \vdots \\ \xi_n \end{pmatrix}; \quad \xi = X \cdot \eta.$$

Подберем такие константы $\alpha > 0$ и $f > 0$, что первые два момента случайных величин $\dfrac{\xi^{\text{т}} A \xi}{\alpha}$ и $\sigma_\eta^2 \chi_f^2$ совпадают [40]. Первые моменты случайной величины $\dfrac{\xi^{\text{т}} A \xi}{\alpha}$ равны

$$\mathsf{E}\left(\frac{\xi^{\text{т}} A \xi}{\alpha}\right) = \mathsf{E}\left(\frac{\sum\limits_{i=1}^{n-1} (\xi_{i+1} - \xi_i)^2}{2\alpha}\right) = \frac{n-1}{\alpha} \sigma_\eta^2,$$

$$\mathsf{D}\left(\frac{\xi^{\text{т}} A \xi}{\alpha}\right) = \frac{\sum\limits_{i=1}^{n-1} \mathsf{D}((\xi_{i+1} - \xi_i)^2)}{4\alpha^2} + \frac{\sum\limits_{1 \leqslant i < j \leqslant n-1} 2\mathrm{cov}\left((\xi_{i+1} - \xi_i)^2, (\xi_{j+1} - \xi_j)^2\right)}{4\alpha^2} = \frac{\sigma_\eta^4}{\alpha^2}(3n-4).$$

В силу того, что χ_f^2 — случайная величина, имеющая χ_f^2-распределение с f степенями свободы, после приравнивания получим

$$\begin{cases} \dfrac{n-1}{\alpha} \sigma_\eta^2 = f \cdot \sigma_\eta^2; \\ \dfrac{\sigma_\eta^4}{\alpha^2}(3n-4) = 2f \cdot \sigma_\eta^4, \end{cases} \tag{16}$$

откуда

$$\begin{cases} \alpha = \alpha(n) = \dfrac{3n-4}{2(n-1)}; \\ f = f(n) = \dfrac{2(n-1)^2}{3n-4} \approx \dfrac{2}{3} n \text{ при } n \gg 1. \end{cases} \tag{17}$$

При подобранных значениях $\alpha = \alpha(n)$ и $f = f(n)$ из [41] следует, что квантили распределения случайных величин $\dfrac{\xi^{\mathrm{T}} A \xi}{\alpha}$ и $\sigma_\eta^2 \chi_f^2$ будут отличаться не более чем на $O\left(\dfrac{1}{f(n)}\right)$. В силу этого и из теоремы 1 следует, что квантили распределения случайных величин $\dfrac{(n-1)S_n}{\alpha(n)} = f(n)S_n$ и $W_d(f(n), \Sigma)$ будут отличаться не более чем на $O\left(\dfrac{1}{f(n)}\right)$.

Далее, так как по предположению $\mathbf{x}_i \sim \mathcal{N}(\mu, \Sigma)$, то $\bar{x}_n = \dfrac{1}{n}\sum\limits_{i=1}^{n} \mathbf{x}_i \sim \mathcal{N}\left(\mu, \dfrac{\Sigma}{n}\right)$, при этом $\bar{x}_n$ и S_n — независимые величины [42, 43]. Так как $\mathbf{x} \sim \mathcal{N}(\mu, \Sigma)$, то $\mathbf{x} - \bar{x}_n \sim \mathcal{N}\left(\mathbf{0}, \Sigma\left(1 + \dfrac{1}{n}\right)\right)$ и, аналогично, $\mathbf{x} - \bar{x}_n$ не зависит от S_n.

Из свойств распределения Уишарта [43], из того, что квантили распределения случайных величин $\dfrac{(n-1)S_n}{\alpha(n)} = f(n)S_n$ и $W_d(f(n), \Sigma)$ отличаются не более чем на $O\left(\dfrac{1}{f(n)}\right)$, и из [41, 42] следует, что квантили распределения статистики T_n из (12) будут отличаться от квантилей распределения Фишера с $(d, f(n) - d + 1)$ степенями свободы не более чем на $O\left(\dfrac{1}{f(n)}\right) = O\left(\dfrac{1}{n}\right)$, где $f(n) = O(n)$. ■

Таким образом, было показано, что, во-первых, оценивать ковариационную матрицу при помощи скользящих размахов можно достаточно точно (данная оценка является несмещенной и состоятельной) и, во-вторых, контрольная граница в модифицированной карте Хотеллинга может быть вычислена из распределения Фишера.

5. Предлагаемый метод обнаружения ЦВЗ

При решении задачи обнаружения ЦВЗ указанные выше методы контроля качества можно использовать для обнаружения разладки (появления ЦВЗ) после наблюдения нескольких пустых контейнеров (чем больше пустых контейнеров, тем выше вероятность правильного обнаружения). Применение таких подходов в классическом виде в стегоанализе не совсем корректно, так как в теории контроля качества подразумевается, что перед разладкой было достаточно много наблюдений при нормальном ходе производства, а после разладки производятся только бракованные изделия. В задаче стегоанализа такая ситуация происходит очень редко. Маркированные и немаркированные контейнеры обычно появляются в перемешанном порядке (если предположить, что, например, абонент маркирует только часть контейнеров, либо заранее неизвестно, какой из нескольких абонентов использует ЦВЗ), при этом контейнер с ЦВЗ может передаваться в самом начале работы канала связи. Поэтому предлагается перед первым наблюдением (изображением, полученным по каналу связи) провести обучение на выборке созданных самостоятельно заведомо пустых контейнеров.

Если детектор признает очередной наблюдаемый контейнер пустым (разладки не произошло), то контейнер добавляется в обучающую выборку. Если же контейнер признан содержащим ЦВЗ, тогда объявляется тревога и обучающая выборка остается прежней до следующего наблюдения.

Каждому изображению поставим в соответствие вектор из d признаков $\mathbf{x}_i \in \mathbb{R}^d$, каждая компонента которого есть некоторый признак контейнера. Кроме того, можно

выбирать из набора признаков наиболее информативные по отношению к конкретному методу внедрения ЦВЗ (изменения которых существеннее остальных при переходе от пустого контейнера к маркированному).

Постановка задачи. Будем считать, что все векторы признаков изображений, в которых нет ЦВЗ, порождены вероятностным распределением, принадлежащим одному классу. Соответственно изображения с ЦВЗ имеют признаки, порожденные другим распределением. При внедрении ЦВЗ в изображение распределение признаков изменяется существенно (гораздо существеннее, чем при любой другой модификации изображения, не влияющей на восприятие системой человеческого зрения). Модификация изображения, вносящая столь же значительные изменения в распределение признаков, например размытие изображения, как правило, визуально заметна. Задача состоит в том, чтобы по наблюдениям $\mathbf{x}_i$, $i = 1, 2, \ldots$, выявить, какие из изображений в потоке содержат ЦВЗ.

Поставленную задачу можно решить посредством карт контроля качества. В этом случае технологический процесс представляет собой передачу изображений по каналу связи. Процесс считается стабильным (изображение *не* содержит ЦВЗ), если признаки изображения принадлежат распределению с целевыми параметрами, полученными для обучающей выборки. И наоборот, процесс становится нестабильным (изображение содержит ЦВЗ), если распределение признаков отличается от целевого (верна альтернативная гипотеза).

В связи с тем, что для обнаружения ЦВЗ необходим анализ нескольких признаков, следует использовать многомерные контрольные карты, наиболее эффективной из которых, как было показано выше, является карта Хотеллинга.

Эффективность работы контрольной карты возрастает, если на этапе обучения использовались изображения, максимально похожие на тестовое. Также эффективность работы карты будет тем выше, чем больше объем обучающей выборки. Таким образом, при достаточно больших объемах обучающей выборки изображения, включенные в данную выборку, будут минимально отличаться друг от друга, и, как следствие, будут существовать такие подвыборки обучающих изображений, изменение ряда признаков (например, яркость изображения, цветовые гистограммы, блочность изображения, дисперсия цветового слоя, дисперсия ненулевых коэффициентов ДКП) которых будет иметь монотонный характер.

Присутствие тренда в некоторых признаках изображений (в том числе и из обучающей выборки), вызванного не внедрением ЦВЗ (при внедрении ЦВЗ тренд, как таковой, не появляется, а происходит скачок за пределы контрольной границы), должно быть учтено при работе контрольной карты. Следовательно, влияние тренда на объявление тревоги должно быть минимизировано.

Для этого предлагается использовать карту контроля качества с применением модифицированной статистики Хотеллинга, а именно критерий, основанный на статистике T_n (см. (12)). Пусть n — количество изображений, которые уже были предъявлены до текущего момента времени (допустим, что в этих изображениях ЦВЗ еще не было), а $\mathbf{x}$ — вектор признаков нового изображения, которое было предъявлено для анализа в текущий момент времени. Необходимо решить, содержит ли предъявленное изображение ЦВЗ.

Зададим уровень значимости $\alpha \in (0, 1)$. Вычислим значение контрольной границы $g(n)$:

$$\int_{g(n)}^{\infty} p_{d,f(n)-d+1}(x)dx = \alpha, \tag{18}$$

где $p_{d,f(n)-d+1}(x)$ — плотность распределения Фишера с d и $f(n)-d+1$ степенями свободы.

К обучающей выборке предъявляются следующие требования:

1) объем выборки n должен быть таким, что $f(n)-d+1>0$;
2) желательно, чтобы изображения обучающей выборки максимально походили на тестируемое (принадлежали к одному типу изображений (пейзаж, средний план, поле), имели примерно одинаковую гистограмму цветов, одинаковое разрешение, одинаковый коэффициент качества, были созданы однотипным способом (сфотографированы одним фотоаппаратом) и т. п.).

Критерий на основе предложенной статистики можно описать следующим образом:

1) накапливаем выборку данных $V_N=\{x_i : i=1,\dots,N\}$ — векторы признаков пустых изображений. N должно быть таким, что $f(N)-d+1>0$;
2) вычисляем $\bar{x}_N$ и S_N;
3) для тестируемого изображения задаем уровень значимости $\alpha\in(0,1)$ и выбираем $g(N)$, как описано выше (18). Проверяем условие: если $T_N<g(N)$, то принимаем гипотезу, что изображение не содержит ЦВЗ, в противном случае принимаем гипотезу, что содержит ЦВЗ.

Кроме предложенного способа для вычисления порога $g(n)$ можно также применять метод Монте-Карло, однако данный метод требует большого объема обучающей выборки и является достаточно трудоемким по сравнению с предложенным.

Вариант 1. Допустим, требуется оценить величину $g(n)$ для конкретного значения n. Рассматриваем выборку пустых изображений в количестве $5n$. Положим $N\gg 1$. Выделяем N случайных подвыборок объема $n+1$ из общей выборки.

Для каждой подвыборки, рассматривая в качестве базы n изображений, для оставшегося одного изображения вычисляем значения статистики T_n. Таким образом, получаем N значений статистики T_n.

Строим по полученным значениям гистограмму. Выбираем такое значение $g(n)$, чтобы α значений статистики T_n были больше $g(n)$.

Вариант 2 (bootstrap). Рассматриваем выборку объема $n+1$. Используя первые n изображений (с номерами $1,\dots,n$), считаем статистику T_n для последнего изображения, далее циклически сдвигаем выборку. То есть теперь используем изображения с индексами $n+1,1,2,\dots,n-1$ в качестве базы и считаем статистику T_n для изображения с индексом n.

Всего таких циклических сдвигов будет $n+1$ штук, то есть получим $n+1$ значений статистики T_n. Далее для вычисления порогов поступаем аналогично предыдущему варианту.

Обнаружение ЦВЗ в графических контейнерах проиллюстрировано при помощи модифицированных контрольных карт Хотеллинга (см. рис. 1–4). Результаты работы метода обнаружения ЦВЗ, внедренных методом Digimarc, показаны на рис. 1. При помощи статистики T_n оценивалось изменение выборочного среднего (в качестве набора признаков использовался характеристический вектор из [1]). Обучение проводилось на 200 фотореалистичных изображениях, индексы которых откладываются по оси абсцисс. На карте отображены значения статистики T_n для пустых контейнеров (обозначены точками) и для их маркированных аналогов (обозначены крестиками). Контрольная граница вычисляется из распределения Фишера (по формуле (18)). Ошибки 1-го и 2-го рода достаточно низкие, несмотря на небольшой объем обучающей выборки. Это обуславливается хорошим подбором изображений в обучающей выборке (обучающая

выборка была одной и той же для всех изображений тестовой выборки; обе выборки состоят из изображений, представляющих собой пейзажи летнего леса) и вставкой большого количества копий ЦВЗ в контейнер (зависит от алгоритма внедрения ЦВЗ).

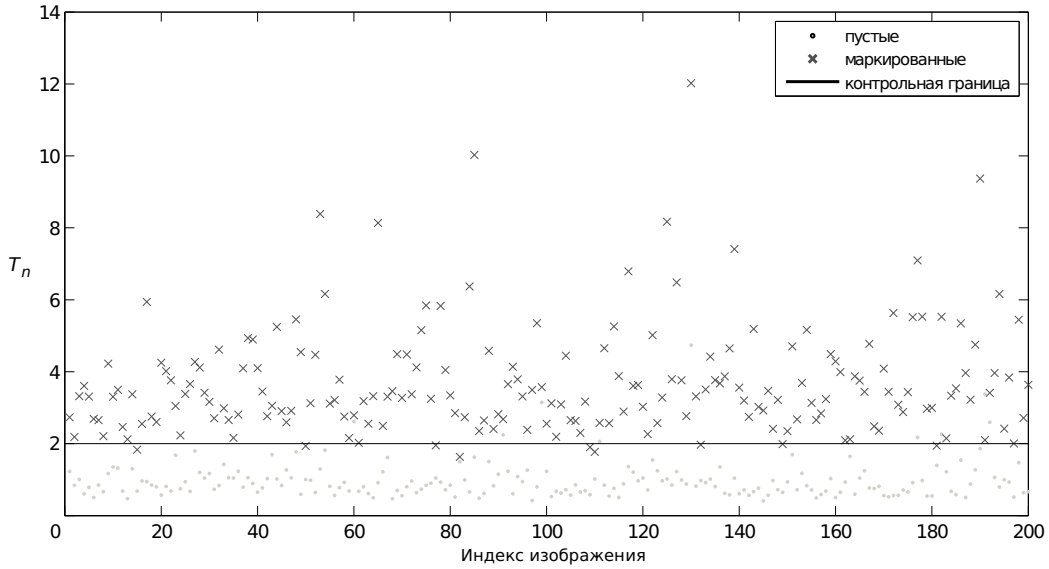

Рис. 1. Обнаружение ЦВЗ, внедренных Digimarc, при помощи модификации контрольной карты. Контрольная граница вычисляется из распределения Фишера

Качество классификации пустых контейнеров и контейнеров, содержащих ЦВЗ, внедренных различными методами, отображено на рис. 2. Тестовая выборка состоит из 300 изображений: 100 пустых контейнеров и 200 маркированных. При этом половина маркированных получена из пустых внедрением одним методом внедрения ЦВЗ, вторая половина получена также из пустых контейнеров другим методом внедрения ЦВЗ. Из рис. 2 видно, что пустые контейнеры (обозначены точками) классифицированы отдельно от их маркированных аналогов (метод внедрения ЦВЗ 1 — кружки, метод внедрения ЦВЗ 2 — крестики). Контрольная граница вычисляется из распределения Фишера (как и прежде, на основе только обучающей выборки пустых контейнеров). В зависимости от параметров и самой сути метода внедрения ЦВЗ соответствующие маркированные контейнеры находятся ближе или дальше от контрольной границы.

Таким образом, можно сделать вывод о том, что данная контрольная карта не привязана к конкретному методу внедрения ЦВЗ и позволяет обнаруживать ЦВЗ, внедренные различными методами.

Схожесть изображений обучающей выборки и тестового изображения имеет важное значение. На рис. 3 и 4 показаны результаты классификации для одной и той же тестовой выборки. При этом на рис. 3 показана классификация с обучением на однотипных с тестовыми контейнерах. На рис. 4 обучение производилось на смешанной выборке (состоящей из контейнеров как похожих на тестовые, так и отличающихся от них). В качестве контейнеров рассматривались изображения двух типов: рассвет и облака. Тестовая выборка содержала 10000 изображений. Обучающие выборки составляли по 3500 изображений каждого типа.

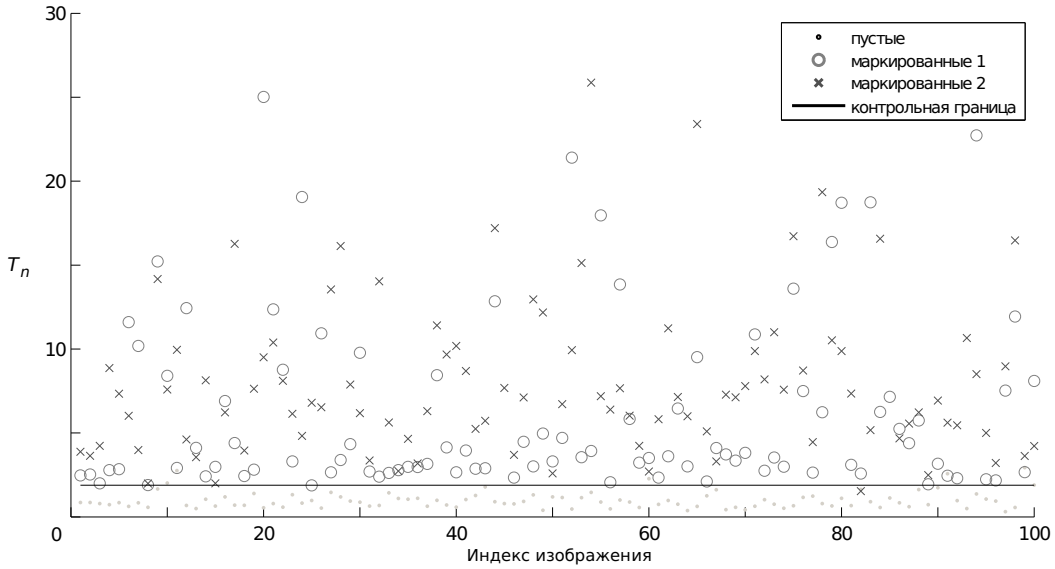

Рис. 2. Результат работы модифицированной контрольной карты при обнаружении ЦВЗ, внедренных различными методами

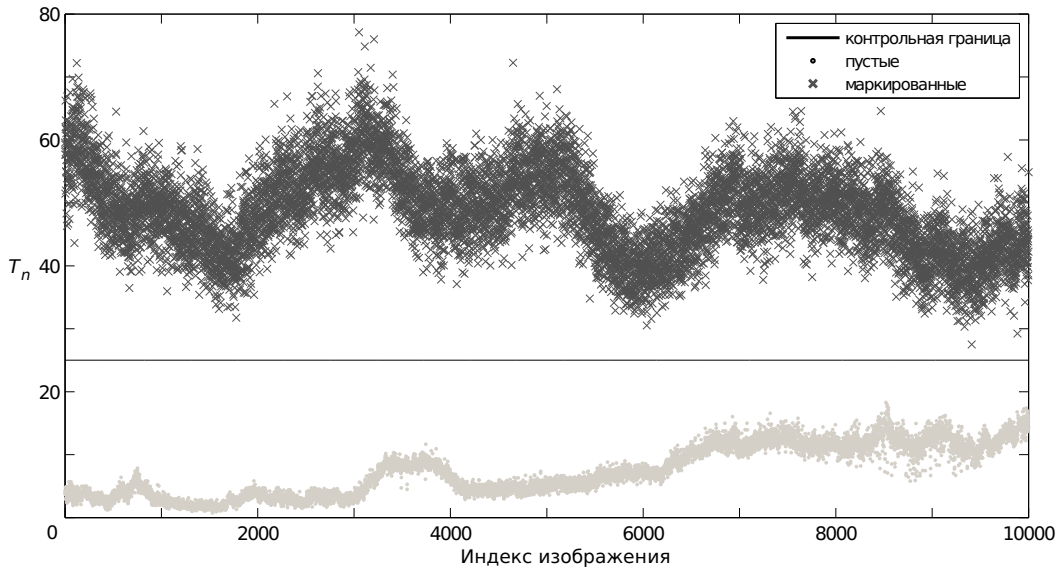

Рис. 3. Обнаружение ЦВЗ, внедренных Digimarc, при помощи модифицированной контрольной карты. Контрольная граница получена из распределения Фишера. Тип изображений обучающей и тестовой выборок совпадает

При обучении на достаточно большой (несколько тысяч) выборке контейнеров, не похожих на тестовые, наблюдается неплохое разделение на классы (ложная тревога и пропуск цели около 0,1), однако вычисленная контрольная граница не соответствует истинной.

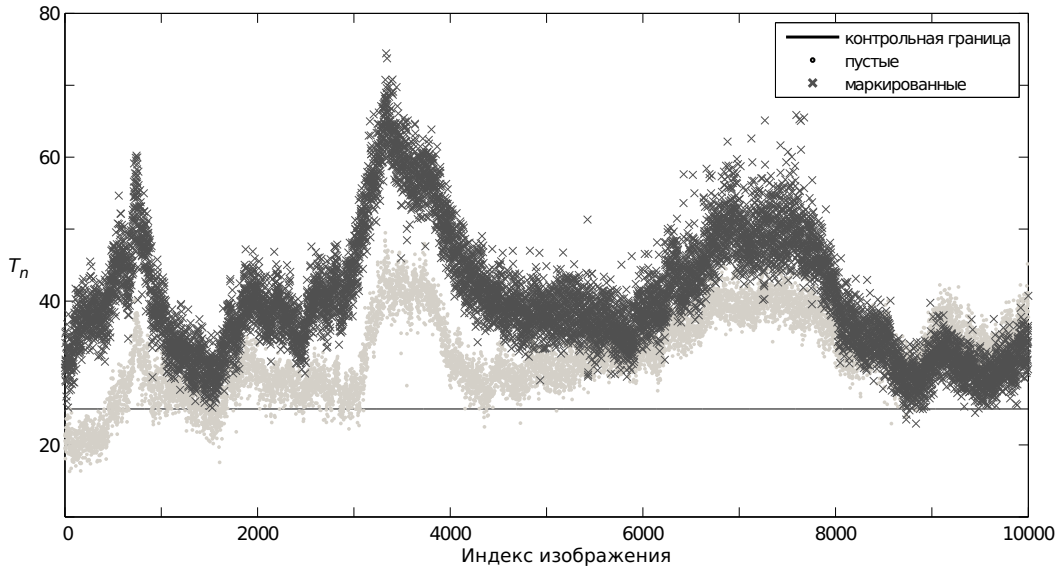

Рис. 4. Обнаружение ЦВЗ, внедренных Digimarc, при помощи модифицированной контрольной карты. Контрольная граница получена из распределения Фишера. Обучающая выборка состоит из изображений двух типов

ЛИТЕРАТУРА

1. *Fridrich J.* Feature-based steganalysis for JPEG images and its implications for future design of steganographic schemes // Proc. 6th Information Hiding International Workshop. LNCS. 2005. V. 3200. P. 67–81.
2. *Lyu S., Farid H.* Steganalysis using higher-order image statistics // IEEE Trans. Inform. Forens. Secur. 2006. V. 1. No. 1. P. 111–119.
3. *Holotyak T., Fridrich J., Voloshynovskiy S.* Blind Statistical Steganalysis of Additive Steganography Using Wavelet Higher Order Statistics // 9th IFIP TC-6 TC-11 Conf. on Communications and Multimedia Security. LNCS. 2005. V. 3677. P. 273–274.
4. *Pevny T., Fridrich J.* Detection of double-compression in JPEG images for applications in steganography // International Workshop on Digital Watermarking. Springer Verlag, 2007.
5. *Pevny T., Fridrich J.* Merging Markov and DCT Features for Multi-Class JPEG Steganalysis // Proc. SPIE Electronic Imaging. V. 6505. Photonics West, 2007. P. 03–04.
6. *Cancelli G., Doerr G., Cox I. J., Barni M.* Detection of LSB steganography based on the amplitude of histogram local extrema // Proc. International Conf. on Image Processing. 2008. P. 1288–1291.
7. `www.outguess.org` — Программный продукт StegDetect 0.6.
8. *Shewhart W. A.* The application of statistics as an aid in maintaining quality of manufactured product // J. Amer. Statist. Assoc. 1925. P. 546–548.
9. *Shewhart W. A.* Economic control of manufactured product. New York: Von Nostrand Reinhold, 1931.
10. *Page E. S.* Continuous inspection schemes // Biometrika. 1954. V. 41. P. 100–114.
11. *Page E. S.* Control charts with warning lines // Ibid. 1955. V. 42. P. 243–254.
12. *Roberts S. W.* Control charts based on geometric moving average // Technometrics. 1959. V. 1. P. 239–250.
13. *Roberts S. W.* A comparison of some control charts procedures // Ibid. 1966. V. 8. P. 411–430.

14. *Ширяев А. Н.* Обнаружение спонтанно возникающих эффектов // Докл. АН СССР. 1961. Т. 138. № 4. С. 799–801.
15. *Колмогоров А. Н., Прохоров Ю. В., Ширяев А. Н.* Вероятностно-статистические методы обнаружения спонтанно возникающих эффектов // Труды МИАН. 1988. Т. 182. С. 4–23.
16. *Вальд А.* Последовательный анализ. М.: Физматгиз, 1960. 328 с.
17. *Juran J. M.* Juran's quality handbook. McGraw-Hill, 5th ed., 1998. P. 109–117.
18. *Окрепилов В. В.* Управление качеством. СПб.: Наука, 2000. 912 с.
19. ГОСТ Р 50779.42–99 «Статистические методы. Контрольные карты Шухарта».
20. *Bersimis S., Psarakis S., Panaretos J.* Multivariate statistical process control charts: an overview // Qual. Reliab. Engng. Int. 2007. V. 23. P. 517–543.
21. *Клячкин В. Н.* Модели и методы многомерного статистического контроля технологического процесса: дис. ... докт. техн. наук. Ульяновск, 2003. 285 с.
22. *Hotelling H.* Multivariate Quality Control Illustrated by Air Testing of Sample Bombsights // Techniques of Statistical Analysis. New York: MacGraw-Hill, 1947. P. 111–184.
23. *Hotelling H.* The generalization of Student's ratio // Ann. Math. Statist. 1931. V. 2. P. 360–378.
24. *Williams J. D., Woodall W. H., Birch J. B., Sullivan J. H.* Distribution of Hotelling's T^2-statistic based on the successive differences estimator // J. Qual. Techn. 2006. V. 38. No. 3. P. 217–229.
25. *Крамер Г.* Математические методы статистики. М.: Мир, 1975. 648 с.
26. *Woodall W. H., Ncube M. M.* Multivariate CUSUM quality control procedures // Technometrics. 1985. V. 27. P. 285–292.
27. *Crosier R. B.* Multivariate generalizations of cumulative sum quality-control schemes // Ibid. 1988. V. 30. P. 291–303.
28. *Клячкин В. Н.*Многомерный статистический контроль технологического процесса. М.: Финансы и статистика, 2003. 192 с.
29. *Lowry C. A., Woodall W. H., Champ C. W., Rigdon S. E.* A multivariate EWMA control chart // Technometrics. 1992. V. 34. P. 46–53.
30. *Montgomery D. C.* Introduction to Statistical Quality Control. 3rd Edition. New York: John Wiley & Sons, 1996.
31. *Jackson J. E., Mudholkar G. S.* Control procedures for residuals associated with principal components analysis // Technometrics. 1979. V. 21. P. 341–349.
32. *Hawkins D. M.* Multivariate quality control based on regression-adjusted variables // Ibid. 1991. V. 33. P. 61–75.
33. *Lowry C. A., Montgomery D. C.* A review of multivariate control charts // IEEE Trans. 1995. V. 27. P. 800–810.
34. *Lucas J. M.* A modified V-mask control scheme // Technometrics. 1973. V. 15. P. 833–847.
35. *Champ C. W., Woodall W. H.* Exact results for Shewhart control charts with supplementary runs rules // Ibid. 1987. V. 29. P. 393–399.
36. *Митаг Х., Ринне Х.* Статистические методы обеспечения качества. М.: Машиностроение, 1995. 616 с.
37. *Ryan T. P.* Statistical methods for quality improvement. New York: John Wiley & Sons, 1989. 420 p.
38. Прикладная статистика: Классификация и снижение размерности: справочное издание / под ред. С. А. Айвазяна. М.: Финансы и статистика, 1989. 607 с.
39. *Клячкин В. Н.* Об оценке ковариационной матрицы при многомерном статистическом контроле технологического процесса // ОПиПМ. 2007. Т. 14. № 4. С. 174.

40. *Satterthwaite F. E.* An approximate distribution of estimates of variance components // Biometrics Bulletin. 1946. V. 2. P. 110–114.
41. *Welch B. L.* The generalization of "student's" problem when several different population variances are involved // Biometrika. 1947. V. 34. P. 28–35.
42. *Seber G. A. F.* Multivariate observations. New York: John Wiley & Sons, 1984.
43. *Bilodeau M., Brenner D.* Theory of multivariate statistics. Springer, 1999.

2010 Математические основы надёжности вычислительных и управляющих систем №2(8)

МАТЕМАТИЧЕСКИЕ ОСНОВЫ НАДЁЖНОСТИ ВЫЧИСЛИТЕЛЬНЫХ И УПРАВЛЯЮЩИХ СИСТЕМ

DOI 10.17223/20710410/8/7

УДК 681.3.012+681.3-192

ПРИМЕНЕНИЕ НЕНАДЁЖНЫХ ТЕСТОВ ДЛЯ САМОДИАГНОСТИКИ МОДУЛЬНЫХ ВЫЧИСЛИТЕЛЬНЫХ СИСТЕМ ПРИ КРАТНЫХ ОТКАЗАХ

Ю. К. Димитриев*, А. Ф. Задорожный**

*Институт физики полупроводников СО РАН, г. Новосибирск, Россия;
**Новосибирский архитектурно-строительный университет, г. Новосибирск, Россия

E-mail: dimi@isp.nsc.ru, 3af@ngs.ru

Исследуется самодиагностирование на системном уровне для модульных вычислительных систем (ВС) при кратных неисправностях и при использовании ненадёжных тестов. Конкретизировано понятие, введена численная характеристика ненадёжного теста. Предложена классификация ненадёжных тестов по введённому критерию. Выделен класс ненадёжных тестов, перспективный в отношении его практической реализации. Методом имитационно-статистического моделирования проанализирована зависимость эффективности самодиагностирования от свойств ненадёжного теста. Анализ проведён на примере описанного авторами децентрализованного алгоритма самодиагностирования. Выполнено сравнение введённых показателей эффективности самодиагностирования при использовании ненадёжных тестов, соответствующих известной модели ПМЧ (Препараты, Метца и Чжена), и в случае применения предложенного авторами класса ненадёжных тестов. Определены условия, при которых использование класса предложенных авторами ненадёжных тестов позволяет улучшать показатели эффективности средств самодиагностирования ВС.

Ключевые слова: *модели самодиагностирования, модульные системы, кратные отказы, ненадёжные тесты.*

Введение

Исследуется возможность децентрализованной диагностики модульных вычислительных систем в условиях кратных отказов. Модули с точки зрения диагностики являются неделимыми единицами системы. Их функциональные характеристики позволяют им в одиночку осуществить полную проверку других (соседних) модулей и оценить их исправность. При этом состояние ВС определяется сопоставлением результатов тестирования, полученных разными модулями (самодиагностика ВС). Кратность неисправностей есть предельно допустимое число неисправных модулей, при котором ещё существует возможность определения состояния ВС.

При децентрализованном диагностировании каждый модуль ВС определяет своё состояние независимо от других модулей, сопоставляя исходы тестирования, полученные от некоторого подмножества других модулей.

В качестве диагностической модели системы используется известная теоретико-графовая модель, предложенная в работе [1]: диагностической моделью ВС служит

орграф $G = (V, E)$, в котором множество вершин $V = \{1, 2, \ldots, N\}$ представляет модули ВС, а множество дуг $E = \{(i, j) : i, j \in V\}$ — тесты над ними. Для $i, j \in V$ дуга (i, j) существует, если и только если модуль i может проверить модуль j. Вершины графа имеют отметки x, значения которых определяют состояние модулей: $x(i) = 0$ при исправности и $x(i) = 1$ при неисправности модуля i. Исходы тестов над модулями (тест прошёл / тест не прошёл) соответствуют двоичным весам дуг графа. Совокупность вершин F_k, $F_k \subseteq V$, имеющих единичные значения меток x, составляет образ неисправностей системы, а совокупность значений весов $a(i, j)$, $(i, j) \in E$, дуг графа образует его синдром S_k. При заданной кратности неисправностей t допустимое множество $F(t)$ состояний системы составляют все возможные сочетания из r неисправных модулей, где $r = 1, 2, \ldots, t$. Каждый образ неисправностей мощности r может порождать 2^r синдромов.

В терминах описанной модели определение состояния системы заключается в идентификации вершин её диагностического графа, образующих множество F_k, по заданному синдрому S_k.

В каждом конкретном случае результат тестирования зависит от состояния тестирующего и тестируемого модулей. Поэтому тест будем описывать четверкой булевых переменных $(a_{gg}, a_{gb}, a_{bg}, a_{bb})$, каждая из которых представляет возможное значение исхода тестирования в зависимости от технического состояния модулей: a_{gg} — оба модуля исправны, a_{gb} — тестирующий модуль исправен, а тестируемый неисправен, и т. д. При этом исход $a(i, j)$ теста принимаем равным нулю, если модуль i считает исправным модуль j, и единице — в противном случае. Для рассматриваемой модели принято, что диагностический тест, выполняемый исправным модулем, является полным (т. е. обнаруживает все допустимые неисправности тестируемого модуля) и корректным (т. е. правильно идентифицирует состояние тестируемого модуля). Когда тест выполняется неисправным модулем, он утрачивает полноту или корректность, вследствие чего получаемая тестирующим модулем оценка может не соответствовать фактическому состоянию тестируемого модуля. Говорят, что такой тест является ненадёжным.

Таким образом, исходы тестирования, соответствующие модели ПМЧ, можно описать четвёркой $(0, 1, \alpha, \alpha)$. Полнота и корректность теста, выполняемого исправным модулем, определяют детерминированность двух первых элементов четвёрки. Ненадёжность теста, выполняемого неисправным модулем, выражается использованием знака α для третьего и четвертого элементов четвёрки.

Исходы ненадёжного теста могут быть детерминированными или случайными (недетерминированными). Если принять во внимание сущность теста, то можно сделать определённые упрощающие предположения об условиях непрохождения теста. Например, если для тестирования используется сравнение результатов одного и того же пользовательского задания, назначенного на соседние модули, и если эти задания достаточно сложны, то маловероятно, что два неисправных модуля выполнят задание с одинаковым результатом. Тесты такого рода, описываемые четвёркой $(0, 1, \alpha, 1)$, использованы в модели Барси, Грандони и Маестрини (БГМ) [2]. Детерминированность элемента a_{bb} означает, что переход неисправных модулей в одно и то же состояние (в смысле теории автоматов) относится к категории недопустимых состояний системы. В литературе [3] рассматриваются и другие модели, отличающиеся условиями детерминированности исхода ненадёжного теста.

Существующие методы *системного диагностирования* делят на вероятностные и детерминистические. Обзор методов системного диагностирования можно найти в работе [4]. Цель алгоритмов вероятностного диагностирования состоит в попытке обес-

печить корректное диагностирование с высокой вероятностью для произвольных отказовых состояний (образов неисправностей) системы $F_k \subseteq V$ и любых порождаемых ими исходов тестирования (синдромов). Это значит, что создаваемый диагностический образ может оказаться либо некорректным (исправные модули идентифицируются как неисправные и наоборот), либо неполным (состояние некоторых модулей остаётся неопределенным). Преимущества вероятностного подхода — простота и скорость алгоритма диагностирования, отсутствие ограничивающих допущений на структуру системы и класс допустимых неисправностей.

Детерминистические методы, базирующиеся на теоретико-графовых моделях, не принимают во внимание случайность образов неисправностей и порождаемых ими синдромов. Они гарантируют корректную и полную диагностику для любого образа неисправностей F_k из ограниченного класса допустимых неисправностей $F(t)$ и для каждого синдрома S_k, совместного с ним, при условии соблюдения определенных априорных требований к структуре тестовых связей и к поведению неисправных модулей. Возможность корректной идентификации состояния системы достигается ценой усложнения структуры системы и алгоритмов диагностирования.

Общее предположение для работ, основанных на теоретико-графовых моделях, состоит в том, что все модули системы имеют одинаковую вероятность быть неисправными. Впервые представление случайных событий введено в теоретико-графовую модель в работе [5]. Авторы рассмотрели случай, когда модули системы имеют разную вероятность оказаться неисправными. Они ввели вероятностную меру p и вывели необходимые и достаточные условия p—t-диагностируемости системы. В такой системе корректно идентифицируется любое подмножество (неисправных) модулей, имеющее мощность не более t и априорную вероятность неисправности не менее p. Однако если в системе есть подмножества, для которых априорная вероятность незначительно меньше p, то вероятность некорректной идентификации состояния соответствующих модулей окажется лишь немногим меньше, чем вероятность их корректной идентификации.

В работе [6] показано, что если вероятность неисправности одинакова для всех модулей системы, то необходимые и достаточные условия p—t-диагностируемости совпадают с условиями t-диагностируемости, выведенными для ПМЧ- и БГМ-модели.

Вопрос о свойствах случайного исхода ненадёжного теста в литературе не рассматривается. Цель данной работы состоит в том, чтобы восполнить этот пробел и проанализировать свойства недетерминированного ненадёжного теста и их влияние на возможность диагностирования системы.

1. Свойства ненадёжного теста

Надёжность теста, выполняемого неисправным модулем, обеспечивается введением дополнительных программно-аппаратурных средств, которые удорожают систему. Характер этих средств в значительной мере зависит от свойств самого теста. Например, предположим, что тестирование состоит в выполнении тестируемым модулем специально подобранной тестовой программы и последующем сравнении тестирующим модулем результатов тестирования, полученных от тестируемого модуля, с эталонными значениями, хранящимися в памяти тестирующего модуля. Тогда, чтобы обеспечить надёжность децентрализованного диагностирования, потребуется хранить эталонные значения в устойчивой к отказам постоянной памяти каждого модуля, а для сравнения исходов тестирования с эталонными применять аппаратуру повышенной отказоустойчивости.

Исход недетерминированного ненадёжного теста в литературе описывается как величина, не имеющая числового значения. Введём в качестве численной меры полноты и корректности теста, выполняемого неисправным модулем, значения $p(0)$ и $p(1)$ — вероятности получения исхода теста $a(i,j) = 0$ и $a(i,j) = 1$ соответственно; $p(0) + p(1) = 1$. Учитывая последнее равенство, в дальнейшем для характеристики надёжности теста используем значение $p(1)$. В литературе подробно исследованы две граничные (в смысле достигаемой кратности неисправностей) модели. Они отличаются предположением о детерминированности исхода ненадёжного теста. Модель ПМЧ [1] описывается четвёркой $(0, 1, \alpha, \alpha)$. Недетерминированное значение элементов четвёрки, относящихся к описанию свойств ненадёжного теста, молчаливо предполагает, что исходы ненадёжного теста имеют произвольное и равновероятное значение, то есть $p(0) = p(1) = 0{,}5$. При таком предположении кратность допустимых неисправностей вычислительной системы не превышает значения $t \leqslant \lfloor (N-1)/2 \rfloor$, где N — число модулей в системе, а $\lfloor x \rfloor$ — наибольшее целое, не превышающее величины x. Модель БГМ [2] описывается четвёркой $(0, 1, \alpha, 1)$. Здесь предполагается, что для исправного тестируемого модуля $p(0) = p(1) = 0{,}5$, а для неисправного тестируемого модуля $p(1) = 1$. При таком предположении кратность допустимых неисправностей системы увеличивается до значения $t = N - 2$.

Приведённые результаты иллюстрируют зависимость диагностических свойств системы от вероятностных свойств ненадёжного теста.

Исход ненадёжного теста может зависеть или не зависеть от технического состояния проверяемого модуля. В модели ПМЧ эти значения не зависят от состояния тестируемого модуля, а в модели БГМ — зависят. По соотношению между $p(0)$ и $p(1)$ ненадёжные тесты делим на девять классов, перечисленных в табл. 1. В каждом классе выделяем подклассы в соответствии с численными значениями вероятностей $p(0)$ и $p(1)$ в диапазоне $1 \geqslant p(0) \geqslant 0$, $1 \geqslant p(1) \geqslant 0$ при нормирующем условии $p(0) + p(1) = 1$. Численные значения $p(0)$ и $p(1)$ определяются при проектировании средств тестирования ВС.

Таблица 1

Классы ненадёжных тестов

Номер класса	Проверяемый модуль исправен	Проверяемый модуль неисправен
1	$p(0) = p(1)$	$p(0) = p(1)$
2	$p(0) > p(1)$	$p(0) = p(1)$
3	$p(0) < p(1)$	$p(0) = p(1)$
4	$p(0) = p(1)$	$p(0) > p(1)$
5	$p(0) > p(1)$	$p(0) > p(1)$
6	$p(0) < p(1)$	$p(0) > p(1)$
7	$p(0) = p(1)$	$p(0) < p(1)$
8	$p(0) > p(1)$	$p(0) < p(1)$
9	$p(0) < p(1)$	$p(0) < p(1)$

В приведённой классификации модель БГМ соответствует классу 7, а модель ПМЧ — классу 1.

Очевидно, что наилучшим является класс 8, для которого вероятность исхода, соответствующего фактическому состоянию проверяемого модуля, больше вероятности исхода, не соответствующего фактическому состоянию проверяемого модуля. Аналогично, наихудшим классом является класс 6. Особый случай составляют классы 1,

5 и 9. Для класса 1, описывающего модель ПМЧ, исход ненадёжного теста не зависит от фактического состояния проверяемой вершины. Для классов 5 и 9 существуют подклассы с исходами ненадёжного теста, которые как зависят, так и не зависят от фактического состояния проверяемой вершины.

Достижение зависимости исхода ненадёжного теста от состояния проверяемого модуля требует усложнения теста и системы диагностирования в целом. Независимость исхода ненадёжного теста достигается более простыми средствами.

Исходя из необходимости обеспечивать высокие надёжность и полезность ВС, для ненадёжного теста с независимыми исходами имеет смысл добиваться значения $p(1) > 0{,}5$. В самом деле, для смежных модулей i и j исход $a(i,j) = 1$ означает, что $\{i,j\} \cap F_k \neq \varnothing$. Поэтому для поддержания высокого уровня полезности системы предпочтительнее приостановить всякое взаимодействие между указанными модулями (даже ценой приостановки работы исправного модуля j, попавшего под подозрение), нежели подвергнуть систему риску разрушительного воздействия со стороны неидентифицированного неисправного модуля i, как могло бы быть, если принять $p(1) < 0{,}5$. Кроме того, если иметь в виду сложность тестового воздействия и многоэлементность ответной реакции со стороны тестируемого модуля, то становится очевидным, что единичный исход ненадёжного теста более естественен и более вероятен, чем нулевой. В этом отношении достижение для ненадёжного теста условия $p(0) \geqslant 0{,}5$ потребовало бы применения специальных средств. Следовательно, наиболее простым в реализации является класс 9 с независимыми исходами ненадёжного теста. Эта модель занимает промежуточное положение между моделью $(0,1,\alpha,\alpha)$ и моделями $(0,1,1,1)$, $(0,1,1,\alpha)$, $(0,1,\alpha,1)$.

Существующие модели исходов ненадёжного тестирования молчаливо предполагают, что для каждого допустимого образа неисправностей $F_k \in F(t)$ все порождаемые им синдромы S_k равновероятны. Введение численной меры для исхода ненадёжного теста позволяет построить более адекватную реальности функцию распределения для множества синдромов, порождаемых заданным образом неисправностей. Введение численной меры для исхода ненадёжного теста позволяет также поставить новую задачу — задачу синтеза теста с заданной надёжностью.

Указанные соображения послужили основанием для сравнения эффективности систем диагностирования с ненадёжными тестами классов 1 и 9. Для класса 9 выбрано $p(1) = 0{,}75$ независимо от фактического состояния тестируемого модуля. Эффективность использования ненадёжных тестов классов 1 и 9 изучается методом имитационно-статистического моделирования на примере децентрализованного алгоритма самодиагностирования «СЧЁТ», описанного в работе [7]. Этот алгоритм соответствует свойству живучести вычислительной системы, при котором она сохраняет работоспособность в условиях действующего на неё потока отказов отдельных модулей и их восстановления.

2. Алгоритм диагностирования

В качестве диагностического графа ВС используется её рабочий граф, заданный использованием системы по назначению. Он представляет собой диаграмму Кэли некоторой конечной абелевой группы Г. Дуги в диаграмме Кэли имеют отметки $g_0, g_1, \ldots, g_{n-1}$ образующих группы. Описанные графы названы КАИС-графами (графы для представления конечных абелевых изотропных и симметричных структур) [8]. Благодаря свойству симметричности представляемой структуры каждая вершина диагностического графа инцидентна $2n$ дугам с метками образующих и обратными им.

Рассматриваемые ВС являются t-диагностируемыми с ремонтом, т. е. для них существует последовательность выполнения множества всех возможных тестов и ремонт выявленных неисправных модулей, которая позволяет идентифицировать все неисправности, первоначально присутствовавшие в системе, при условии, что число их не превышает t.

Правильное определение состояния ВС возможно, если оно осуществляется заведомо исправными модулями, составляющими (диагностическое) ядро системы. Алгоритм, описанный в работе [7], реализует известный метод раскрутки, при котором на каждом шаге исправность одних модулей определяется другими, исправность которых установлена на предыдущих шагах. Функциональные характеристики модулей ВС таковы, что любой из них можно использовать в качестве ядра. Если $t < N$, то задачу определения состояния ВС можно уточнить: по результатам взаимной проверки идентифицировать хотя бы один заведомо исправный модуль. Выявленные им исправные модули включаются в состав наращиваемого ядра ВС.

В работе [7] установлено, что для диагностического графа, обладающего свойством симметричности, преобразование

$$\forall i, j \in V \ [((i,j) \in E) \wedge (a(i,j) = 1) \to E = E - \{(i,j),(j,i)\}]$$

порождает граф $G_0 = (V_0,\ E_0)$, в котором отметки x вершин каждой его компоненты связности совпадают. Предложено выбирать в качестве t такое наибольшее t^*, при котором для любого F_k, $|F_k| \leqslant t^*$, и любого порождаемого F_k синдрома в G_0 ещё существует хотя бы одна компонента связности порядка не менее $(t^* + 1)$. С учетом предположения о допустимой кратности неисправностей в такую компоненту входят вершины, представляющие исправные модули, которые, следовательно, могут использоваться как ядро ВС. Для некоторых из исследованных групп-графов диагностической структуры ВС значения t^*/N приведены в табл. 2.

Таблица 2

Кратность неисправностей

N	20	25	30	37	40	50	60	70	80	90
t^*	4	8	8	8	12	12	16	16	16	20
t^*/N	0,200	0,320	0,2667	0,217	0,300	0,240	0,267	0,229	0,200	0,222
$k_{_N}$	0,047	0,038	0,031	0,031	0,024	0,019	0,015	0,013	0,012	0,010

Для упрощения изложения терминология ВС распространена на представляющий её граф. Поэтому в дальнейшем допускаем выражения «состояние графа» вместо «состояние системы», «исправная (неисправная) вершина» вместо «исправный (неисправный) модуль», «исход теста (i,j)» вместо «вес дуги (i,j)» и т. п.

Алгоритм самодиагностирования состоит из двух этапов. На первом этапе осуществляется построение G_0. При этом последовательно строятся компоненты связности и определяются их порядки до тех пор, пока не будет получена компонента связности порядка $(t^* + 1)$. Каждая компонента связности G_{0k} получает номер k вершины, инициировавшей её формирование (корневая вершина). Обозначим G'_0 часть графа G_0, построенную к текущему шагу алгоритма. Если вершина i включена в компоненту связности G_{0k}, то очередная включаемая в неё вершина ищется среди вершин, смежных с нею в графе $G - G'_0$. Вершина j, смежная с вершиной i, включается в компоненту связности G_{0k}, если $a(i,j) = 0$. Формирование компоненты продолжается до тех пор,

пока порядок её не станет равным (t^*+1) или пока не будут испытаны все вершины графа $G-G'_0$, смежные с нею. Если порядок сформированной компоненты менее (t^*+1), то процесс формирования компонент связности повторяется для вершин из $G-G'_0$. Для выбора очередной корневой вершины используется описанный в [7] алгоритм обхода КАИС-графа по остовному контуру.

На втором этапе, после выделения диагностического ядра, осуществляется определение состояния всех вершин, не вошедших в диагностическое ядро. При этом сначала с помощью найденных исправных вершин устанавливается состояние других вершин из компоненты связности, которая имеет порядок (t^*+1), а затем — состояние вершин из других компонент связности.

Рассмотрим процесс формирования компоненты связности более подробно. Пусть $G_{0k}=(V_{0k},E_{0k})$, $G_{0k}\in G'_0$, — формируемая компонента связности, а i — очередная тестирующая вершина этой компоненты. Множество $n(i)$ вершин, смежных с i в графе G, разделяем на непересекающиеся подмножества $V_D(i)$ и $V_B(i)$ прямого и обратного направлений тестирования соответственно. К множеству $V_D(i)$, $0\leqslant|V_D(i)|\leqslant 2n$, вершин прямого направления тестирования относим смежные с i вершины из $G-G'_0$. Множество $V_B(i)$ вершин обратного направления тестирования составляет одна вершина, а именно та принадлежащая G_{0k} вершина j, из которой вершина i включена в эту компоненту. Соответственно сказанному, тесты над вершинами из $V_D(i)$ называем прямыми, а над вершинами из $V_B(i)$ — обратными. Принято, что обратный тест для вершины i осуществляется после выполнения всех прямых тестов.

Пусть в компоненте связности $G_{0k}=(V_{0k},E_{0k})$ вершина i — очередная тестирующая вершина и $V_R=V_{0k}\setminus\{i\}$. В работе [7] установлены следующие свойства структуры компоненты связности:

1) в формируемой компоненте связности для всякого $k\in V_R$ существует простой путь из вершины k в вершину i;

2) в компоненте связности, закончившей своё формирование (порядок компоненты не более t^*), все дуги являются симметрическими.

Рассмотрим вершины $i,j\in V_{0k}$, смежные в компоненте связности G_{0k}. Выделим три правила, использование которых позволяет по исходу одного теста определять состояние целой группы вершин (так называемые групповые тесты).

1. Если выполняется $a(i,j)=a(j,i)=0$, то обе вершины исправны или обе неисправны.

2. Если вершина j неисправна и $a(i,j)=0$, то вершина i также неисправна.

3. Если выполняется $a(i,j)=0$ и $a(j,i)=1$, то вершина i неисправна независимо от состояния вершины j.

Нетрудно видеть, что правила 1 и 2 являются транзитивными.

Из транзитивности правила 1 следует, что в компоненте связности G_{0k}, закончившей своё формирование, все вершины имеют одинаковое состояние. Это позволяет определять состояние всех её вершин по исходу единственного теста, выполняемого из любой вершины диагностического ядра над смежной с ней вершиной из G_{0k}.

Рассмотрим формируемую компоненту связности G_{0k}, если её порядок не более t^*. Пусть неисправность вершины j этой компоненты установлена в процессе её обратного тестирования из вершины $i\in V_{0k}$. В таком случае для вершины j выполняется свойство (1) структуры, и вследствие транзитивности правила 2 тест (i,j) идентифицирует неисправность всех вершин из $\{j\}\cup V_B(j)$. Таким образом достигается возможность «раннего», т. е. до выявления диагностического ядра, обнаружения неисправных вершин.

Аналогично осуществляется «ранняя» идентификация неисправных вершин из $\{j\} \cup V_B(j)$ для компоненты связности порядка $(t^* + 1)$, когда выполняется обратный тест (i, j) из очередной идентифицированной исправной вершины i этой компоненты. Заметим, что если для выполняемого обратного теста получено $a(i, j) = 0$, то состояние исправности устанавливается для вершины j и для вершин, связанных с нею цепями из симметричных дуг согласно правилу 1.

Подводя итог сказанному, финальные состояния (состояние исправности или неисправности) для вершин графа могут быть установлены в любой из последовательно выполняемых фаз диагностирования: фаза 1 — от начала выполнения алгоритма до идентификации первой исправной вершины; фаза 2 — от идентификации первой исправной вершины i до завершения идентификации финального состояния всех вершин компоненты связности, в которую входит вершина i; фаза 3 — от завершения фазы 2 до завершения идентификации финального состояния для всех вершин графа.

Эффективность диагностирования ВС изучается методом имитационно-статистического моделирования функционирования системы при использовании децентрализованной реализации алгоритма «СЧЁТ». Корректность, полнота и конечность этого алгоритма показаны в работе [7].

Алгоритм «СЧЁТ» является полным в том смысле, что он обеспечивает определение финального состояния для любого допустимого образа неисправностей и любого порождаемого им синдрома. Образ неисправностей и порождаемый им синдром — случайные события. Законы распределения этих событий задают среду для выполнения алгоритма самодиагностирования. Изучение поведения алгоритма самодиагностирования в такой среде позволяет достичь двух целей:

1. Определить условия и границы практического применения рассматриваемого подхода к самодиагностированию.

2. Установить целесообразность реализации отдельных элементов алгоритма самодиагностирования (практическую полноту алгоритма), так как некоторые его шаги могут выполняться редко из-за малой вероятности осуществления событий, вызывающих их выполнение.

Параметрами, задающими условия выполнения алгоритма самодиагностирования, являются структура диагностического графа $G = (V, E)$ и мощность r образа неисправностей. Для исследуемого диагностического графа $G = (V, E)$ порядка $N = |V|$ параметр r пробегает значения от 1 до t^*.

Для каждой пары G и r проводится B опытов, в ходе которых собираются необходимые статистические данные. В каждом опыте неисправные вершины, составляющие образ неисправностей $F_k \subset V$, порождаемый этим образом синдром S_k и вершина, которая соответствует модулю, инициирующему диагностирование системы, выбираются по жребию.

В ходе моделирования обобщаются данные для определения интегральных потенциальных характеристик эффективности алгоритма, а также для оценки его практической полноты. Изучается зависимость введенных показателей от вероятности, с которой неисправные модули определяют действительное состояние тестируемых ими модулей.

В качестве основных интегральных характеристик эффективности диагностики используются следующие показатели.

1. Вектор-коэффициент $\mathbf{K}_1 = \{K_{1r}\}$, $r \in \{1, 2, \ldots, t^*\}$, относительных расходов на определение состояния ВС (служит мерой полноты использования тестов над системой). Его координатами являются величины $K_{1r} = M_r(T_1)/|E|$, где $M_r(T_1)$ — среднее

число тестов, необходимых для определения состояния ВС, когда $|F_k| = r$. Для рассматриваемых диагностических графов $|E| = 2nN$ (в рассматриваемом случае $n = 2$). Значение $|E|$ соответствует общему числу тестов над модулями системы.

2. Вектор-коэффициент $\mathbf{K}_2 = \{K_{2r}\}$ (служит для оценки относительных накладных расходов на выделение диагностического ядра). Его координатами являются величины $K_{2r} = (M_r(T_2) - t^*)/t^*$. Здесь $M_r(T_2)$ — среднее число тестов, необходимых для идентификации первой исправной вершины, когда $|F_k| = r$, а кратность неисправностей t^* совпадает с минимальным числом тестов, выполнение которых необходимо для идентификации первой исправной вершины.

Очевидно, что чем меньше значения K_{1r} и K_{2r}, тем эффективнее алгоритм самодиагностирования.

Для оценки условий практической полноты алгоритма применяются вектор-коэффициенты, координатами которых являются значения характеристик событий, связанных с формированием компонент связности и возможностью идентификации состояния их вершин. Эти события характеризуются средними значениями, которые рассчитываются по множеству всех B опытов, а также *условными* средними значениями, которые рассчитываются по множеству $B(q) \subseteq B$ опытов, в которых анализируемое событие q случается. Завершение формирования компонент связности характеризуют показатели 3 – 6.

3. Вектор-коэффициент $\mathbf{K}_3 = \{K_{3r}\}$ относительного покрытия образа неисправностей мощности r. Координатами $\mathbf{K}_3$ являются величины $K_{3r} = M_r(A_F)/r$, где $M_r(A_F)$ — среднее число событий A_F, т. е. среднее число неисправных вершин $i \in F_k$, которые были включены во все сформированные в фазе 1 компоненты связности. Средние значения координат $\mathbf{K}_3$ рассчитываются на множестве всех B опытов.

4. Вектор-коэффициент $\mathbf{K}_4 = \{K_{4r}\}$ эффективности обратного тестирования с координатами $K_{4r} = (M_r(A_1) + M_r(A_2))/r$, где $M_{rH}(A_1)$ и $M_{rH}(A_2)$ — среднее число событий A_1 и A_2 соответственно. События A_1 и A_2 состоят в идентификации неисправных вершин по результатам обратного тестирования соответственно в ходе выполнения фазы 1 и 2 диагностирования.

5. Вектор-коэффициент $\mathbf{K}_5$, координатами которого служат значения условного среднего числа $K_{5r} = M_{5r}(A_1)$ событий A_1. Значения $M_{5r}(A_1)$ рассчитываются на множестве $B(A_1)$ опытов, в которых событие A_1 имеет место.

6. Вектор-коэффициент $\mathbf{K}_6$, координатами которого служат значения условного среднего числа $K_{6r} = M_{6r}(T_6)$ тестов до первого события A_1. Значения $M_{6r}(T_6)$ рассчитываются на множестве $B(A_1)$ опытов, в которых событие A_1 происходит.

Кроме того, в число показателей эффективности алгоритма самодиагностирования входят показатели для оценки числа и порядка формируемых в фазе 1 компонент связности, эффективности обратного тестирования в фазах 1 и 2, а также эффективности группового тестирования в фазах 1, 2 и 3.

Эффективность использования ненадёжных тестов класса 9 изучается на примере групп-графа с двумя образующими g_0 и g_1, заданного определяющими отношениями $g_0^{30} = I$, $g_1^2 = g_0^6$, где I — единичный элемент группы; порядок группы $N = 30$.

В ходе моделирования выявлена монотонность убывания значений координат анализируемых вектор-коэффициентов при росте N. Поэтому численные значения, полученные для $N = 30$, для масштабируемых ВС могут рассматриваться как оценки снизу для эффективности использования тестов класса 9.

3. Обсуждение результатов

Введённые показатели эффективности алгоритма «СЧЁТ» представляют собой монотонно возрастающие функции от r и монотонно убывающие функции от N.

Говоря в общем, полученные результаты моделирования подтверждают ожидаемое изменение характера самодиагностирования, связанное с изменением вероятности исхода ненадёжного теста. Для образа неисправностей F_k заданной мощности r переход от тестов класса 1 к тестам класса 9 вызывает увеличение числа компонент связности, образуемых в первой фазе самодиагностирования, и уменьшение их порядков, а также уменьшение числа неисправных вершин, включённых в компоненту связности порядка $(t^* + 1)$. Следствием этого является ожидаемое падение эффективности обратного и группового тестирования.

Относительно зависимости численных значений показателей алгоритма от используемого класса ненадёжных тестов можно отметить следующее.

1. Результаты имитационно-статистического моделирования показали, что предлагаемый алгоритм самодиагностирования обладает высокой эффективностью, которая увеличивается с ростом N. Число тестов, необходимых для идентификации финального состояния вершин графа в исследованном диапазоне изменения N, не превосходит величины $0{,}4\,|E|$.

Значения координат вектор-коэффициента $\mathbf{K}_1$ для классов 1 и 9 при $N = 30$ приведены в табл. 3. Наблюдаемый незначительный рост относительных расходов на определение состояния ВС для класса 9 связан с уменьшением вероятности использования группового тестирования.

Таблица 3

Координаты вектор-коэффициента $\mathbf{K}_1$

r	1	2	3	4	5	6	7	8
Класс 1	0,311	0,314	0,317	0,320	0,324	0,329	0,333	0,339
Класс 9	0,312	0,314	0,318	0,321	0,326	0,331	0,336	0,343

2. В исследованном диапазоне значений $90 \geqslant N \geqslant 13$ и $t^* \geqslant r \geqslant 1$ координаты вектор-коэффициента $\mathbf{K}_2$ для тестов класса 1 находятся в диапазоне от 0,1 до 0,5 при $N = 13$ и от 0,01 до 0,3 при $N = 90$.

Увеличение вероятности исхода $a(i, j) = 1$ для ненадежного теста класса 9 при $r \leqslant t^*$ практически не сказывается на величине относительных накладных расходов на выделение ядра. Значения координат вектор-коэффициента $\mathbf{K}_2$ для классов 1 и 9 при $N = 30$ приведены в табл. 4.

Таблица 4

Координаты вектор-коэффициента $\mathbf{K}_2$

r	1	2	3	4	5	6	7	8
Класс 1	0,035	0,08	0,145	0,18	0,24	0,30	0,39	0,48
Класс 9	0,041	0,09	0,148	0,20	0,27	0,34	0,42	0,52

3. Для тестов класса 1 координаты $\mathbf{K}_3$ находятся в диапазоне от 0,077 до 0,092 для $N = 13$ и в диапазоне от 0,011 до 0,013 для $N = 90$. При этом значения координат вектор-коэффициента $\mathbf{K}_4$ меньше соответствующих значений $\mathbf{K}_3$ менее чем на 7 процентов для $N = 13$ и менее чем на 4 процента — для $N = 90$. Разность значений $\mathbf{K}_3$ и

$\mathbf{K}_4$ указывает на долю тех неисправных вершин в компонентах связности, сформированных в фазах 1 и 2, состояние которых определяется в фазе 3 с помощью групповых тестов из исправных вершин. Вклад в значение координат вектор-коэффициента $\mathbf{K}_3$ определяется отношением $M_r(A_1) : M_r(A_2) : M_r(A_3) = 10^{-3} : 1 : 1^{-1}$, то есть основной вклад в значение координат этого вектор-коэффициента вносит вхождение неисправных вершин в компоненту связности порядка (t^*+1). В фазе 3 алгоритма определяется не менее $0{,}9r$ неисправных вершин.

Для тестов класса 9 значения координат вектор-коэффициента $\mathbf{K}_3$ относительного покрытия образа неисправностей остаются на уровне тестов класса 1, а значения координат вектор-коэффициента $\mathbf{K}_4$ эффективности использования обратного тестирования уменьшаются. В табл. 5 приведены значения координат вектор-коэффициентов $\mathbf{K}_3$ и $\mathbf{K}_4$ для $N = 30$.

Таблица 5

Координаты вектор-коэффициентов $\mathbf{K}_3$ и $\mathbf{K}_4$

r	1	2	3	4	5	6	7	8
$\mathbf{K}_3$, класс 1	0,0338	0,0341	0,0362	0,0373	0,0383	0,0400	0,0415	0,0439
$\mathbf{K}_3$, класс 9	0,0338	0,0348	0,0348	0,0369	0,0378	0,0900	0,0400	0,0421
$\mathbf{K}_4$, класс 1	0,0311	0,0321	0,0335	0,0348	0,0355	0,0380	0,0391	0,0408
$\mathbf{K}_4$, класс 9	0,0228	0,0233	0,0233	0,0250	0,0255	0,0265	0,02720	0,0290

4. Событие A_1 характеризуется вектор-коэффициентами, координатами которых являются вероятность, с которой оно случается, среднее число неисправных вершин, идентифицируемых в фазе 1 самодиагностирования, и среднее число тестов до первого события A_1. Указанные средние значения рассчитываются как по исходам всех B опытов, так и по исходам $B(A_1)$ тех опытов, в которых событие A_1 случается. Эти показатели позволяют уточнить обстоятельства, при которых событие A_1 происходит.

4.1. Анализ значений координат $\mathbf{K}_3$ показывает, что для тестов класса 1 событие A_1 является достаточно редким. В рассмотренном диапазоне значений N наибольшая величина вероятности события A_1, равная 0,0139, достигается при $N = 22$ и $r = 7$.

4.2. Установлено, что в классе 1 ненадёжных тестов при любом N для $r = t^*/2$ среднее число событий A_1 составляет приблизительно 10^{-4}. Наибольшее среднее число неисправных вершин, идентифицируемых в фазе 1, рассчитанное на множестве всех B опытов, составило величину 0,018. Это значение найдено для $N = 40$ и $r = 12$.

4.3. Вектор-коэффициент $\mathbf{K}_5$ характеризует эффективность обратного тестирования в фазе 1. Для класса 1 ненадёжных тестов в исследованном диапазоне параметров координаты $\mathbf{K}_5$ находятся в диапазоне между 1 и 1,34 вершинами. По итогам моделирования можно принять, что в исследованном диапазоне параметров для любого N при $r = t^*/2$ условное среднее число неисправных вершин, идентифицируемых в фазе 1 алгоритма, равно 1,2 (заметим, что условное среднее число выполненных для этого тестов составляет величину 6,125).

4.4. Структуру формируемой компоненты связности, в которой по исходу обратного тестирования идентифицируются вошедшие в неё неисправные вершины, характеризует вектор-коэффициент $\mathbf{K}_6$. Анализ показал, что в классе 1 ненадёжных тестов координаты этого вектор-коэффициента находятся в диапазоне от 5,5 до 7,0; при этом наименьшее возможное количество тестов до события A_1 при $r \geqslant 2$ и $n = 2$ равно 5 (заметим, что при $r = 1$ событие A_1 не происходит). Указанные значения среднего числа тестов до события A_1 означают, что в фазе 1 идентификация неисправной вершины

происходит чаще всего для компоненты связности порядка 2 по исходу обратного теста, выполняемого второй вершиной, включенной в эту компоненту.

4.5. Для класса 9 ненадёжных тестов вероятность события A_1 идентификации неисправных вершин в фазе 1 самодиагностирования уменьшается по сравнению с тестами класса 1. Средний размер компоненты связности, финальное состояние вершин которой идентифицируется в фазе 1 алгоритма, также уменьшается, а само событие идентификации неисправных вершин случается позднее, чем для тестов класса 1.

В табл. 6 приведены значения координат вектор-коэффициентов $\mathbf{K}_5$ и $\mathbf{K}_6$ для $N = 30$.

Таблица 6

Координаты вектор-коэффициентов $\mathbf{K}_5$ и $\mathbf{K}_6$

r	2	3	4	5	6	7	8
$\mathbf{K}_5$, класс 1	1,005	1,047	1,080	1,098	1,171	1,181	1,273
$\mathbf{K}_5$, класс 9	1,000	1,008	1,041	1,064	1,108	1,130	1,175
$\mathbf{K}_6$, класс 1	5,320	5,750	5,890	6,050	6,050	6,290	6,512
$\mathbf{K}_6$, класс 9	6,120	6,290	6,290	6,580	6,030	6,720	6,830

5. Полученные при моделировании данные показывают, что значения координат $p(A_2)$ вектор-коэффициента вероятности идентификации неисправной вершины в фазе 2 самодиагностирования образуют функцию, близкую к линейной.

5.1 Для класса 1 ненадёжных тестов аппроксимирующая функция для зависимости $p(A_2)$ от значений r имеет вид $y = k_N r$, где k_N — величина $p(A_2)$, найденная для заданного N при $r = 1$. Максимальная относительная погрешность коэффициента k_N при аппроксимации методом наименьших квадратов составляет не более 1%. Значения k_N приведены в табл. 2.

При использовании ненадёжных тестов класса 9 вероятность идентификации неисправных вершин в фазе 2 сохраняет линейный характер, но значение k_N уменьшается. Так, для $N = 30$ получено $k_N = 0{,}023$, а для $N = 90$ имеем $k_N = 0{,}0084$.

5.2. В фазе 2 идентификация неисправных вершин осуществляется при выполнении обратного тестирования, имеющего групповой характер.

Моделирование показало, что для класса 1 ненадёжных тестов среднее число неисправных вершин, идентифицируемых в фазе 2 самодиагностирования, находится в диапазоне от 1,07 до 1,72 при $N = 13$ и от 1,005 до 1,2 при $N = 90$.

Для класса 9 ненадёжных тестов среднее число неисправных вершин, идентифицируемых в фазе 2, уменьшается. В табл. 7 приведены значения этого показателя для $N = 30$.

Таблица 7

Среднее число неисправных вершин, идентифицируемых в фазе 2

r	1	2	3	4	5	6	7	8
Класс 1	0,031	0,062	0,100	0,140	0,170	0,220	0,270	0,310
Класс 9	0,023	0,048	0,069	0,100	0,125	0,150	0,175	0,200

Анализ показателей, характеризующих структуру компоненты связности порядка $(t^* + 1)$, показал, что структура этой компоненты в большинстве случаев является цепью. Вероятность, что эта компонента связности имеет структуру дерева, равна 0,015.

Это случается, если в состав данной компоненты связности входят неисправные вершины. Эти неисправные вершины образуют ветви, исходящие из корневой вершины.

6. Событие, состоящее в том, что в фазе 1 самодиагностирования потребуется формирование нескольких компонент связности, является достаточно редким. Для исследованных диагностических графов при использовании тестов класса 1 наибольшее значение вероятности, что в фазе 1 самодиагностирования потребуется формирование нескольких компонент связности, составляет 0,0425 (при $N = 25$, $r = 8$), а среднее число компонент связности равно 1,0485. Для любого графа исследованной области значение 0,01 вероятности формирования в фазе 1 более одной компоненты связности достигается при $r = N/2$. (Заметим, что минимальное число формируемых компонент связности равно 1, что соответствует случаю, когда порядок первой компоненты равен $(t^* + 1)$.)

При переходе от класса 1 к классу 9 ненадёжных тестов событие перехода к формированию новой компоненты связности становится чаще, а среднее расстояние передачи метки «Начальная вершина компоненты» уменьшается. В табл. 8 для случая $N = 30$ приведены значения координат вектор-коэффициента $\mathbf{K}_7$, характеризующего среднее число событий, когда в фазе 1 формируется более одной компоненты связности, и вектор-коэффициента $\mathbf{K}_8$, характеризующего среднее число формируемых компонент связности.

Таблица 8

Координаты $\mathbf{K}_7$ и $\mathbf{K}_8$

r	1	2	3	4	5	6	7	8
$\mathbf{K}_7$, класс 1	0,002	0,004	0,007	0,011	0,015	0,018	0,023	0,033
$\mathbf{K}_7$, класс 9	0,002	0,004	0,007	0,012	0,016	0,018	0,022	0,034
$\mathbf{K}_8$, класс 1	0,010	0,021	0,035	0,050	0,063	0,080	0,100	0,180
$\mathbf{K}_8$, класс 9	0,010	0,021	0,035	0,051	0,0631	0,081	0,101	0,181

Типичный сценарий самодиагностирования выглядит следующим образом. В большинстве случаев первая же формируемая компонента связности имеет порядок (t^*+1) вершин и структуру простой цепи. В тех случаях, когда формирование компоненты начинается с неисправной вершины, компонента связности может приобретать структуру дерева, ветви которого исходят из начальной вершины и состоят из неисправных вершин. Таким образом, в фазе 2 алгоритма число выполняемых обратных тестов близко порядку компоненты связности. Если в ходе диагностирования образуется несколько компонент связности, то эти компоненты связности в типичном случае состоят из одной–двух вершин, а число образуемых компонент также исчисляется единицами. Следовательно, эффективность использования групповых свойств тестов в фазах 1 и 3 алгоритма невелика. Начальные вершины последовательно формируемых компонент связности в типичном случае оказываются смежными в диагностическом графе. Возможность идентификации неисправной вершины по исходу обратного тестирования в фазе 1 алгоритма имеет малую вероятность.

Итак, полученные при моделировании результаты показывают, что значительная часть сложности полного и корректного алгоритма самодиагностирования «СЧЁТ» связана с обработкой событий, имеющих малую вероятность. Это даёт возможность в случае использования ненадёжных тестов класса 1 упростить алгоритм, сохранив за ним свойство практически полного. Так, с небольшой потерей эффективности можно

отказаться от: 1) использования алгоритма формирования остовного цикла для диагностического графа; 2) хранения структуры компонент связности, если финальное состояние их вершин не идентифицировано в фазе 1 алгоритма.

Модели системы самодиагностирования с ненадёжными тестами классов 1 и 9 имеют одно и то же множество допустимых неисправностей и совместных с ними синдромов. Различие состоит только в значениях вероятности образования каждого конкретного синдрома. Естественно поэтому, что алгоритм «СЧЁТ» сохраняет работоспособность для ненадёжных тестов обоих этих классов. Переход от класса 1 к классу 9 сказывается лишь на получаемых численных значениях показателей эффективности алгоритма самодиагностирования.

Наиболее существенным в этом отношении является то, что при использовании тестов класса 9, во-первых, уменьшается вероятность образования компонент связности порядка 2 и более; во-вторых, значительно уменьшается вероятность того, что при $a(i,j)=0$ имеет место $j \in F_k$.

Первое обстоятельство позволяет отказаться от выполнения обратных тестов при формировании компонент связности в фазе 1. При этом структура компоненты связности, имеющей порядок более единицы и закончившей формирование, имеет структуру ориентированного дерева. Такая структура на этапе 3 позволяет в ряде случаев выиграть в общем числе тестов, необходимых для установления состояния вершин компоненты связности. Это происходит благодаря использованию групповых свойств теста, выполняемого из идентифицированной исправной вершины. Это свойство имеет следующий вид.

Пусть G_{0i} — компонента связности, имеющая структуру ориентированного восходящего дерева, в котором i — корневая вершина, а j — лист. Пусть k — вершина диагностического ядра, а вершины i и k смежны в диагностическом графе G. Если $a(k,i)=0$, то исход теста имеет групповой характер: все вершины в G_{0i} исправны. Если k — вершина диагностического ядра, а вершины j и k смежны в диагностическом графе G, то $a(k,j)=1$, и вершина j неисправна. К вершине j при этом применимо правило 2 группового тестирования, согласно которому все вершины в G_{0i}, входящие в путь из i в j, неисправны.

Второе обстоятельство позволяет использовать в качестве диагностического ядра последнюю вершину простой ориентированной цепи, состоящей из k, $k < t^*$, вершин, в которой все дуги имеют вес 0. Здесь решающее правило состоит в том, что в ориентированной цепи исправные вершины не могут предшествовать неисправным, и чем длиннее такая цепь, тем больше вероятность, что конечная её вершина исправна. Для заданной вероятности $p(0)$ в ориентированной цепи, начинающейся неисправной вершиной и содержащей k вершин, вероятность, что последняя вершина окажется неисправной, равна $p(0)^{k-1}$. К примеру, для выбранного значения $p(0) = 0{,}25$ при $k=5$ вероятность того, что пятая вершина цепи окажется неисправной, равна $p(0)^4 = 0{,}004$. Заметим, что полученное значение вероятности оказаться неисправной для $(k+1)$-й вершины цепи позволяет использовать в качестве ядра цепь из пяти вершин при любом значении N.

Заключение

Изучается графовая модель, представляющая систему самодиагностирования модульных вычислительных систем при кратных отказах и ненадёжных тестах. Уточнено понятие ненадёжного теста. В качестве численной характеристики ненадёжного теста предложено использовать вероятность его исхода. Дана классификация моделей

самодиагностируемых систем по значениям вероятности исходов ненадёжного теста. Методом имитационно-статистического моделирования проанализировано поведение описанного децентрализованного алгоритма самодиагностирования в среде, которую образуют заданное множество допустимых образов неисправностей и порождаемых ими синдромов. Показана зависимость эффективности алгоритма от вероятности исхода ненадёжного теста. Для выделенного класса ненадёжных тестов выведены условия упрощения алгоритма самодиагностирования.

Проведённые исследования позволяют также поставить новую задачу в области диагностирования систем с кратными отказами — задачу разработки теста с заданными характеристиками надёжности.

ЛИТЕРАТУРА

1. *Preparata F. P., Metze G., Chien R. J.* On connection assignement problem of diagnosable systems // IEEE Trans. El. Comput. 1967. V. EC-16. No. 12. P. 848–854.
2. *Barsi F., Grandoni F., Maestrini P.* A theory of diagnosability of digital systems // IEEE Trans. Comput. 1976. V. C-25. No. 6. P. 585–589.
3. *Радойчевски В. Ц., Шалаев А. Я.* Параллельная диагностируемость модульных систем при децентрализованной дешифрации синдрома // Электронное моделирование. 1992. Т. 14. № 1. С. 57–63.
4. *Lee S. G., Shin K. G.* Probabilistic diagnosis of multiprocessor systems // ACM Computing Surveys. March 1994. V. 26. P. 121–139.
5. *Maheshwari S. N., Hakimi S. L.* On model for diagnosable systems and probabilistic fault diagnosis // IEEE Trans. Comput. 1976. V. C-25. No. 3. P. 228–236.
6. *Fujiwara H., Kinoshita K.* Some extence theorems for probabilistically diagnosable systems // IEEE Trans. Comput. 1978. V. C-27. No. 4. P. 379–384.
7. *Димитриев Ю. К.* Самодиагностика модульных вычислительных систем. Новосибирск: ВО «Наука», 1993. 293 с. (Гл. 3, п. 3.6.)
8. *Корнеев В. В.* О макроструктуре однородных вычислительных систем // Вопр. теории и построения вычисл. систем. Новосибирск, 1974. Вып. 60. С. 17–34.

2010 Математические основы надёжности вычислительных и управляющих систем №2(8)

DOI 10.17223/20710410/8/8

УДК 519.17: 681.3

АНАЛИТИЧЕСКИЙ ПОДХОД К СИНТЕЗУ РЕГУЛЯРНЫХ ГРАФОВ С ЗАДАННЫМИ ЗНАЧЕНИЯМИ ПОРЯДКА, СТЕПЕНИ И ОБХВАТА

В. А. Мелентьев

Институт физики полупроводников СО РАН, г. Новосибирск, Россия

E-mail: melva@isp.nsc.ru

Предложен подход к аналитическому решению задачи синтеза структур системы с заданными свойствами. Подход основан на представлении графа системы его проекциями. Изложены понятия, основные положения и свойства проекций графа. Суть подхода состоит в построении базовой проекции остовного дерева синтезируемого графа и в его доопределении другими проекциями. Это аналогично решению системы уравнений, в качестве которой использовано множество проекций графа. Даны примеры и приведены результаты генерации графов.

Ключевые слова: *порядок, диаметр, обхват и проекция графа, синтез регулярного графа.*

Введение

Проблема синтеза структур компьютерных систем и сетей связи с заданными свойствами представлена в научной литературе достаточно широко. Наиболее распространенный подход к решению этой проблемы состоит в генерации случайных сетей с последующей режекцией не отвечающих заданным критериям вариантов. В качестве критериев при этом используют такие общеизвестные показатели, как диаметр, связность, коэффициент кластеризации и т. п. Известные из [1] исследования устойчивости вычислительных сетей и систем к случайному и/или преднамеренному удалению вершин из их структур свидетельствуют о бóльшей устойчивости регулярных структур: наиболее устойчивая топология случайных графов характеризуется распределением степени вершин с не более чем тремя несовпадениями. При этом ни в теории сетей и систем, ни в фундаментальной ее основе — теории графов проблематика генерации структуры (графа) с заданными коммуникативными свойствами систематическими методами, исключающими необходимость перебора, практически не исследована. Связано это было, прежде всего, с отсутствием аппарата описания графов, позволяющего максимально формализованно производить их анализ и преобразования.

В данной работе впервые представлен аналитический подход к решению проблемы синтеза регулярных графов заданного порядка n и степени s. Подход основан на предложенной в [2, 3] формализации описания графа $G(V, E)$ его проекциями $P(v_0)$, $v_0 \in V$, и состоит в построении базовой проекции остовного дерева генерируемого графа, в анализе этой проекции, в выявлении с ее помощью нижней границы диаметра $d(G)$ и верхней границы обхвата $g(G)$ и в доопределении неизвестных ребер графа другими его проекциями в соответствии с требуемыми значениями показателей. Таким образом, поиск недостающих в остовном подграфе ребер подобен решению системы уравнений, в качестве которой использовано множество проекций генерируемого графа.

1. Основные положения

Во избежание разночтений здесь приведены некоторые общеизвестные определения [4], а также основные сведения о проекциях графа, используемых в работе.

Регулярный граф — связный граф $G(V, E)$, у которого степени всех вершин $v_i \in V$ равны между собой; степень вершин при этом называется степенью $s(G)$ регулярного графа.

Эксцентриситет вершины — для вершины u величина $e(u) = \max\limits_{u \in V} \partial(u, v)$, где $\partial(u, v)$ — расстояние между вершинами u и v из V.

Диаметр — наибольший из эксцентриситетов вершин связного графа: $d(G) = \max\limits_{u \in V} e(u)$.

Обхват — длина минимального цикла в графе.

Проекция $P(v_i)$ графа $G(V, E)$ — описание графа в скобочной форме, отправной точкой (ракурсной вершиной) которого является вершина $v_i \in V$.

Технология построения скобочных описаний графа и их свойства достаточно подробно представлены в работах [2, 3]. Однако в связи с тем, что проекции графа являются базовым понятием предлагаемого в работе подхода и этот инструмент описания графов пока малоизвестен из-за его относительной новизны, приведем краткое изложение основ построения проекций неориентированного связного графа, продемонстрировав их для наглядности примером единичного куба, используемого затем для сопоставления с полученными в результате синтеза графами, обладающими теми же (как в единичном кубе) значениями порядка и степени.

Проекцию $P(w)$ графа $G(V, E)$ с ракурсной вершиной $w \in V$ назовем w-й проекцией этого графа, либо его w-м ракурсом. Для конкретизации числа k уровней в проекции в обозначение добавим соответствующий индекс — $P_k(w)$. Тогда $P_0(w) = w$. Продолжив описание до 1-го уровня, получим $P_1(w) = w^{\mathcal{N}(w)}$. Здесь порожденное вершиной w подмножество $\mathcal{N}(w)$ является окружением вершины w и состоит из $s(w)$ вершин, где $s(w) = \deg(w)$ — степень вершины w. Таким образом, j-я вершина $(i-1)$-го уровня проекции порождает на следующем i-м уровне подмножество $V_{ij} \subset V$ вершин; соответственно, число таких подмножеств равно числу вершин предшествующего уровня. Подмножеству $V_{ij} \subset V$ поставим в соответствие множество предшествующих ему вершин $V'_{ij} \subset V$, включенных в маршрут $M(v_0, v_{i-1,j})$ из ракурсной вершины v_0 в вершину $v_{i-1,j}$, порождающую подмножество V_{ij} и являющуюся его непосредственной предшественницей. Для подмножества вершин 1-го уровня V_{1w}, порожденного единственной вершиной w 0-го уровня, подмножество V'_{1w} состоит из одной вершины: $V'_{10} = \{w\}$. Подмножества $V'_{i+1,j}$ вышестоящих уровней для $i \geqslant 1$ получаем из соответствующих подмножеств V'_{ij} предшествующих уровней добавлением в них непосредственно предшествующей подмножеству $V_{i+1,j}$ вершины $v_{ij} \in V_{ij}$: $V'_{i+1,j} = V'_{ij} \cup \{v_{ij}\}$. Заметим, что в общем случае связных графов, в том числе содержащих циклы, на разных уровнях проекции $P_k(w)$ или на одном и том же ее уровне (кроме первого) может быть несколько экземпляров одной и той же вершины, и индексы этих экземпляров не должны совпадать. Отсутствие кратности (повторяемости) вершин на первом уровне объясняется тем, что рассматриваемые в данной работе объекты не являются мультиграфами; мы не касаемся здесь также особенностей описания графов с петлями, поэтому $v_{i-1,j} \neq v_{ij}$. Общее число C_i вершин i-го уровня проекции равно сумме мощностей подмножеств V_{ij} вершин этого уровня — $C_i = \sum\limits_j |V_{ij}|$, а множество M_i находящихся на этом уровне вершин представляет собой объединение подмножеств V_{ij}, т. е. $M_i = \bigcup\limits_j V_{ij}$

и $C_i \geqslant |M_i|$. Вершины, входящие в состав подмножества V_{ij}, определяются вычитанием из окружения вершины $v_{i-1,j}$, порождающей это подмножество, множества его вершин-предшественниц V'_{ij}: $V_{ij} = \mathcal{N}(v_{i-1,j})\backslash V'_{ij}$. Это исключает повторение вершин в маршрутах, определяемых проекцией. Тогда выражение для 3-уровневой проекции, содержимое каждого уровня которой раскрыто здесь на примере лишь одной вершины одного из подмножеств вершин этого уровня, имеет вид

$$P_3(w) = w^{\{u^{\{v^{\{t:t\in\mathcal{N}(v)\backslash V'_{3v},\ V'_{3v}=V'_{2u}\cup\{u\}=\{w,u\}\}}:v\in\mathcal{N}(u)\backslash V'_{2u},\ V'_{2u}=\{w\}\}}:u\in\mathcal{N}(w)\}}.$$

Здесь множество вершин 1-го уровня состоит из единственного подмножества — $V_{1w} = \mathcal{N}(w)$, мощность этого множества $|V_{1w}| = \deg(w)$. Множество вершин 2-го уровня включает в себя $\deg(w)$ подмножеств, каждое из которых имеет своими непосредственными предшественницами все вершины 1-го уровня; в частности, вершина v входит в подмножество вершин V_{2u}, непосредственной предшественницей которых является вершина u: $v \in V_{2u}$, $V_{2u} = \mathcal{N}(u)\backslash V'_{1u}$, $V'_{1u} = \{w\}$. Итак, множество вершин любого n-го ($n \leqslant k$) уровня проекции $P_k(w)$ объединяет в себе подмножества, число которых равно числу вершин $(n-1)$-го уровня, а их содержимое получено вычитанием из окружений этих вершин всех их предшественниц в данной проекции.

Продемонстрируем данное выше конструктивное описание проекций на простом примере единичного куба (рис. 1).

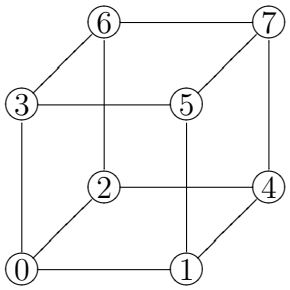

Рис. 1. Единичный куб

Выберем вершину $v_0 = 0$ в качестве ракурсной вершины проекции $P_k(0)$. Окружение этой вершины $\mathcal{N}(0) = \{1, 2, 3\}$ составляет множество вершин 1-го уровня: $M_1 = \{1, 2, 3\}$, $|M_1| = C_1 = 3$. Множество M_2 вершин 2-го уровня объединяет в себе $C_1 = 3$ подмножества, являющиеся окружениями (без вершины 0, непосредственно предшествующей этим подмножествам) трех вершин 1-го уровня: $M_2 = M_{21} \cup M_{22} \cup M_{23} = \{4, 5\} \cup \{4, 6\} \cup \{5, 6\} = \{4, 5, 6\}$, при этом $C_2 = 6$, а $|M_2| = 3$. Отметим, что $M_1 \cup M_2 \neq V$, поэтому построение проекции необходимо продолжить следующим уровнем. Множество M_3 вершин 3-го уровня состоит из 6 подмножеств, представляющих собой окружения соответствующих им 6 вершин 2-го уровня, каждое из которых модифицировано вычитанием множеств предшествующих этому окружению вершин: $M_3 = M^1_{34} \cup M^1_{35} \cup M^2_{34} \cup M^1_{36} \cup M^2_{35} \cup M^2_{36}$. Здесь первая цифра нижнего индекса указывает на принадлежность к соответствующему уровню проекции, вторая идентифицирует вершину графа, а верхний индекс служит для дополнительной индексации нескольких экземпляров одной и той же вершины на рассматриваемом уровне проекции. Из построенной таким образом проекции

$$P_3(0) = 0^{\{1^{\{4^{\{2,7\}},5^{\{3,7\}}\}},2^{\{4^{\{1,7\}},6^{\{3,7\}}\}},3^{\{5^{\{1,7\}},6^{\{2,7\}}\}}\}}$$

видно, что только на 3-м уровне появляется вершина 7, не включенная в состав ни одного из подмножеств предшествующих уровней: $M_3 = \{2,7\} \cup \{3,7\} \cup \{1,7\} \cup \{3,7\} \cup \cup\{1,7\} \cup \{2,7\} = \{1,2,3,7\}$, $C_3 = 12$, $|M_3| = 4$. Запись этой же проекции в однострочном варианте имеет вид

$$P_3(0) = 0\{1\{4\{2,7\},5\{3,7\}\},2\{4\{1,7\},6\{3,7\}\},3\{5\{1,7\},6\{2,7\}\}\}.$$

Открывающиеся скобки перед тем или иным подмножеством вершин указывают на его вложенность, а число скобок, «непогашенных» закрывающимися, определяет уровень вложенности этого подмножества в множества вершин-предшественниц. В работах [2, 3] показано, что уровень вложенности подмножества в множество потомков ракурсной вершины (номер уровня в проекции) определяет ее удаленность от вершин соответствующего подмножества, а k-й уровень, впервые доопределяющий множество вершин всех нижерасположенных уровней проекции графа $G(V,E)$ до V, определяет эксцентриситет ракурсной вершины: $e(v_0) = k$, для которого $\bigcup\limits_{i=0}^{k-1} M_i \neq V$, $\bigcup\limits_{i=0}^{k} M_i = V$. Проекция графа $P_k(v_0)$ считается полной, если ею определены все вершины и все ребра (отношения смежности) этого графа. Тогда необходимые условия полноты проекции могут быть записаны следующим образом: $\bigcup\limits_{i=0}^{k} M_i = V$ и $\bigcup\limits_{i=0}^{k} E_i = E$. Здесь $E_i = \{u,v : u \in M_{i-1}, v \in M_i\}$ — множество ребер, инцидентных парам вершин со смежных уровней проекции. Нетрудно заметить, что второе из этих условий полноты проекции (реберное) поглощает в себе первое (для вершин). Из приведенной выше проекции $P_3(0)$ видно, что и вершинное, и реберное условия полноты выполняются здесь только на 3-м уровне: $\bigcup\limits_{i=0}^{3} M_i = V$ и $|\bigcup\limits_{i=0}^{3} M_i| = |V| = 8$; $E_0 = \varnothing$, $E_1 = \{\{0,1\},\{0,2\},\{0,3\}\}$, $E_2 = \{\{1,4\},\{1,5\},\{2,4\},\{2,6\},\{3,5\},\{3,6\}\}$, $E_3 = \{\{4,2\},\{4,7\},\{5,3\},\{5,7\},\{4,1\},\{6,3\},\{6,7\},\{5,1\},\{6,2\}\}$ и $E = \bigcup\limits_{i=0}^{3} E_i = \{\{0,1\},\{0,2\},\{0,3\},\{1,4\},\{1,5\},\{2,4\},\{2,6\},\{3,5\},\{3,6\},\{4,7\},\{5,7\},\{6,7\}\}$, $|E| = 12$.

Дополним приведенные здесь описания проекций их свойствами, доказанными в [2, 3]. Выше указывалось, что не являющаяся ракурсной вершина $v_j \neq v_0$ проекции $P_k(v_0)$ может находиться на любом из $k > 0$ ее уровней с некоторой кратностью $0 \leqslant m_{ij} \leqslant C_k$. Для вершин из V_{ij}, расположенных на уровнях $0 < i \leqslant k$, существуют упорядоченные множества вершин $W(v_{ij}) = (v_0, v_{10}, \ldots, v_{ij})$, представляющие собой простые цепи из v_0 в v_{ij}, а длины этих цепей — $\partial(v_0, v_{ij}) = i$.

Отметим и другое свойство проекции графа, доказательство которого дано там же: число уровней $k_{\min}(v_0)$ в минимальной полной проекции $P_{k_{\min}}(v_0)$ связного простого графа $G(V,E)$ не меньше эксцентриситета $e(v_0)$ ракурсной вершины v_0 и не превышает увеличенного на единицу его значения:

$$k_{\min}(v_0) = \begin{cases} e(v_0), & \text{если } \nexists\{u,v\} \in M_k \ (\partial(v_0,u) = \partial(v_0,v) = e(v_0) \quad \& \ \partial(u,v) = 1); \\ e(v_0)+1, & \text{если } \exists\{u,v\} \in M_k \ (\partial(v_0,u) = \partial(v_0,v) = e(v_0) \quad \& \ \partial(u,v) = 1). \end{cases}$$

Следствием этого является то, что проекция графа, построенная из любой его вершины до уровня с номером, большим диаметра, всегда является полной.

В дополнение к приведенным выше свойствам, используемым предлагаемым в работе подходом, сформулируем следующее вспомогательное утверждение.

Лемма 1. Если в проекции $P(v_0)$ графа $G(V,E)$ вершина $u \in V$ принадлежит двум подмножествам одного и того же или двум подмножествам разных уровней, то обхват графа $g(G)$ не превышает суммы номеров этих уровней.

Доказательство этого утверждения следует из приведенного выше свойства проекции, согласно которому для любой вершины $u \in V$, расположенной на i-м уровне ($i > 0$) проекции $P(v_0)$ графа $G(V,E)$, существует простая цепь $W(v_0,u)$ из v_0 в u, длина $\partial(v_0,u)$ которой равна i. Изначально мы исключили наличие в синтезируемых здесь графах кратных ребер, поэтому вершина u не может входить в состав подмножества вершин, порожденных одной из вершин предшествующего уровня проекции. Проиндексируем ее принадлежность разным подмножествам, помня при этом, что это одна и та же конечная вершина ($u_1 = u_2 = u$) двух разных путей $W_1(v_0,u)$ и $W_2(v_0,u)$ из v_0 в u. Если пересечение множеств W_1 и W_2 входящих в эти пути вершин содержит всего две вершины ($W_1 \cap W_2 = \{v_0, u\}$), то длина простого цикла, образованного этими цепями, максимальна и равна сумме номеров уровней i_1 и i_2, на которых расположена вершина u. Если же $W_1 \cap W_2$ включает в себя большее число вершин ($|W_1 \cap W_2| > 2$), то очевидно, что существует несколько (именно $|W_1 \cap W_2| - 2 \geqslant 1$) не совпадающих с v_0 вершин, из которых также существуют простые цепи в u_1 и u_2, являющиеся участками цепей W_1 и W_2. При этом вершина v_x пересечения, расположенная на более высоком уровне, является начальной вершиной двух непересекающихся цепей из v_x в u_1 и u_2. Тогда длина цикла, образованного этими цепями, определяется выражением $(i_1 - i_x) + (i_2 - i_x) = i_1 + i_2 - 2i_x$, и $g(G) \leqslant i_1 + i_2$, что и требовалось доказать.

2. Описание подхода

Из приведенных в п. 1 свойств проекций следует, что вершина v_0 регулярного простого графа с невзвешенными ребрами обладает минимальным эксцентриситетом, если уровень k_e проекции $P_{k_e}(v_0)$, на котором впервые выполняется условие $\bigcup\limits_{i=0}^{k_e} M_i = V$, является минимально возможным для заданных значений порядка $n = |V|$ и степени s графа. Максимально возможное при степени графа s число вершин n, расположенных на i-м ($i > 0$) уровне проекции, равно

$$C_i(s) = s(s-1)^{i-1}. \tag{1}$$

Тогда минимальное число уровней в проекции синтезируемого графа с заданными парой $\{n, s\}$ значениями его порядка и степени может быть определено из соотношения

$$1 + s\sum_{i=1}^{k_e-1}(s-1)^{i-1} < n \leqslant 1 + s\sum_{i=1}^{k_e}(s-1)^{i-1}. \tag{2}$$

Из определения диаметра графа как наибольшего из эксцентриситетов всех его вершин следует, что синтезируемый граф будет обладать минимальным диаметром, если условие (2) выполняется для всех его проекций. Диаметр рассмотренного выше единичного куба с $n = 8$, $s = 3$ равен трем — $d(G) = 3$, что превышает полученное из (2) значение $k_e = 2$ для графа с $4 < n \leqslant 10$ и $s = 3$. Это означает принципиальную возможность синтеза графа с диаметром $d(G) = 2$ — меньшим, чем у обладающего теми же значениями порядка и степени единичного куба.

Следует обратить внимание на то, что процедура синтеза графа в предлагаемом подходе достаточно формализована, чтобы не использовать его геометрическое представление, поэтому из соображений общепринятости далее покажем полученный граф

в традиционном виде, но только как результат проведенных нами процедур формального синтеза.

Итак, для генерации 2-уровневой проекции $P_2(v_j)$ графа с $n = 8$, $s = 3$ и диаметром $d = 2$ все его $|V| = 8$ вершин необходимо разместить на двух уровнях проекции; пронумеруем вершины от 0 до 7. На 0-м уровне проекции $P_2(v_0)$ разместим вершину $v_0 = 0$, она же является корневой вершиной выстраиваемого для данной проекции остовного дерева. На 1-м уровне расположим 3 произвольно выбранные вершины (в данном случае выбраны вершины 1, 2 и 3). Так как рассматриваемые здесь графы не имеют кратных ребер, то и вершины 1-го уровня в любой проекции не могут быть кратными. На 2-м уровне проекции в соответствии с (1) $C_2(3) = 6$, т. е. здесь должны быть размещены 6 вершин, в то время как у нас остались нераспределенными всего 4 вершины: 4, 5, 6 и 7. Таким образом, на этом уровне мы имеем возможность разместить все вершины из $\{4, 5, 6, 7\}$, при том что некоторые из них могут быть кратными и/или их число может быть дополнено до $C_2(3) = 6$ вершинами предыдущего уровня. Так как в общем случае подмножества вершин 2-го уровня могут включать в себя и вершины 1-го, то проекцию $P_2(0)$ запишем следующим образом:

$$P_2(0) = 0^{\{1^{\{2,3,4,5,6,7\}_2},2^{\{1,3,4,5,6,7\}_2},3^{\{1,2,4,5,6,7\}_2}\}}. \tag{3}$$

Здесь множество $\{v_x, v_y, \ldots, v_z\}_m$ является потенциальным подмножеством окружения вершины, непосредственно предшествующей этому подмножеству в данной проекции. Индекс m определяет число искомых в данном подмножестве вершин. Изначально в такие подмножества включим все вершины, число известных элементов окружения которых меньше степени графа s. Таковыми здесь являются все вершины графа, кроме 0, т. е. исходным будет подмножество $\{1, 2, 3, 4, 5, 6, 7\}$. Так как вершина не содержит в своем окружении саму себя, то в проекции $P_2(0)$ подмножества соответствующим образом скорректированы. Выше уже отмечено, что множество M_2 вершин 2-го уровня обязательно должно содержать в себе подмножество $\{4, 5, 6, 7\}$, иначе эксцентриситет вершины 0 превысит полученное из (2) значение $e(v_0) = k_e = 2$, причем на оставшиеся два места могут претендовать любые две вершины из $\{1, 2, 3, 4, 5, 6, 7\}$. Отметим также, что при общем числе ребер в графе $|E| = ns/2 = 12$ проекцией (3) определены лишь три ребра. Остальные 9 неизвестных ребер предстоит определить, причем вторым уровнем проекции могут быть определены не более шести ребер. Из этого следуют недостаточность двух уровней проекции для ее полноты и необходимость последующего надстраивания ее 3-м уровнем в случае описания графа не системой, а лишь одной из его проекций.

Из проекции (3) и доказанной в п. 1 леммы видно, что если хотя бы одна из вершин 1-го уровня будет включена в состав 2-го уровня, то длина минимального цикла графа (его обхват) не превысит трех. Если же на 2-м уровне разместить только вершины из $\{4, 5, 6, 7\}$, обхват такого графа будет равен 4. Покажем это на синтезе соответствующих графов.

Итак, для синтеза графа с $g(G) = 3$ соединим вершину $v_1 = 1$ с вершиной $v_2 = 2$; выбор этих вершин может быть произвольным. Получим

$$P_2(0) = 0^{\{1^{\{2,\{4,5,6,7\}_1\}},2^{\{1,\{4,5,6,7\}_1\}},3^{\{4,5,6,7\}_2}\}}.$$

Заметим, что введенное отношение смежности для двух вершин 1-го уровня занимает две из шести позиций 2-го уровня и оставшихся четырёх позиций едва хватает для размещения четырёх вершин множества $\{4, 5, 6, 7\} = M_2 \backslash \{v_1, v_2\} = V \backslash (M_1 \cup M_0)$. Поэтому

в потенциальных списках подмножеств вершин 2-го уровня оставляем только эти вершины. Две из них должны быть смежны вершине 3 и по одной — вершинам 1 и 2, при этом выбор отношений смежности здесь может быть произволен, поскольку вершины подмножества $\{4,5,6,7\}$ пока что изолированы и в этом отношении абсолютно равноправны. Запишем известные и принятые на настоящий момент отношения смежности в виде списка окружений каждой из вершин синтезируемого графа:

$$\begin{array}{llll}\mathcal{N}(0)=\{1,2,3\}, & \mathcal{N}(2)=\{0,1,5\}, & \mathcal{N}(4)=\{1,\{5,6,7\}_2\}, & \mathcal{N}(6)=\{3,\{4,5,7\}_2\},\\ \mathcal{N}(1)=\{0,2,4\}, & \mathcal{N}(3)=\{0,6,7\}, & \mathcal{N}(5)=\{2,\{4,6,7\}_2\}, & \mathcal{N}(7)=\{3,\{4,5,6\}_2\}.\end{array}$$

Построенная в соответствии с этим списком 2-уровневая проекция графа

$$P_2(0)=0^{\{1^{\{2,4\}},2^{\{1,5\}},3^{\{6,7\}}\}}$$

содержит все 8 вершин графа, но не является полной, так как множество вершин 2-го уровня включает в себя вершины с не выявленными пока окружениями. Число известных ребер в синтезируемом графе увеличилось с 3-х до 8-и, но неизвестными при этом остались 4 ребра. Достроив проекцию 3-м уровнем, получим

$$P_3(0)=0^{\{1^{\{2^{\{0,5\}},4^{\{5,6,7\}_2}\}},2^{\{1^{\{0,4\}},5^{\{4,6,7\}_2}\}},3^{\{6^{\{4,5,7\}_2},7^{\{4,5,6\}_2}\}}\}}.$$

Используя приведенный выше список окружений, построим проекцию графа с ракурсной вершиной $v_1=1$:

$$P_3(1)=1^{\{0^{\{2^{\{5\}},3^{\{6,7\}}\}},2^{\{0^{\{3\}},5^{\{4,6,7\}_2}\}},4^{\{5,6,7\}_2}\}}.$$

Как уже отмечалось, равенство диаметра синтезируемого графа заданному значению будет обеспечено, если эксцентриситет любой из его вершин не превысит этого значения. Другими словами, если $d(G)=d$, то все вершины графа должны быть размещены не более чем на $k=d$ уровнях любой v_j-й проекции графа $P_k(v_j)$: $\bigcup\limits_{i=0}^{k} M_i = V$, $k \leqslant d$. Таким образом, анализируя проекцию $P_3(1)$, заметим, что выполнение этого условия в ней единственным образом возможно только при соединении вершины 4 с вершинами 6 и 7. Скорректировав список окружений соответствующим образом ($\mathcal{N}(4)=\{1,\{5,6,7\}_2\} \Rightarrow \mathcal{N}(5)=\{2,\{4,6,7\}_2\}$), получаем единственно возможное решение:

$$\begin{array}{llll}\mathcal{N}(0)=\{1,2,3\}, & \mathcal{N}(2)=\{0,1,5\}, & \mathcal{N}(4)=\{1,6,7\}, & \mathcal{N}(6)=\{3,4,5\},\\ \mathcal{N}(1)=\{0,2,4\}, & \mathcal{N}(3)=\{0,6,7\}, & \mathcal{N}(5)=\{2,6,7\}, & \mathcal{N}(7)=\{3,4,5\}.\end{array}$$

Чтобы истинность решения не вызывала сомнений, ниже даны построенные на его основе все минимальные полные проекции и геометрическое изображение (рис. 2) полученного графа:

$$P_3(0)=0^{\{1^{\{2^{\{5\}},4^{\{6,7\}}\}},2^{\{1^{\{4\}},5^{\{6,7\}}\}},3^{\{6^{\{4,5\}},7^{\{4,5\}}\}}\}},$$

$$P_3(1)=1^{\{0^{\{2^{\{5\}},3^{\{6,7\}}\}},2^{\{0^{\{3\}},5^{\{6,7\}}\}},4^{\{6^{\{3,5\}},7^{\{3,5\}}\}}\}},$$

$$P_3(2)=2^{\{0^{\{1^{\{4\}},3^{\{6,7\}}\}},1^{\{0^{\{3\}},4^{\{6,7\}}\}},5^{\{6^{\{3,4\}},7^{\{3,4\}}\}}\}},$$

$$P_3(3)=3^{\{0^{\{1^{\{2,4\}},2^{\{1,5\}}\}},6^{\{4^{\{1,7\}},5^{\{2,7\}}\}},7^{\{4^{\{1,6\}},5^{\{2,6\}}\}}\}},$$

$$P_3(4) = 4^{\{1^{\{0^{\{2,3\}},2^{\{0,5\}}\}},6^{\{3^{\{0,7\}},5^{\{2,7\}}\}},7^{\{3^{\{0,6\}},5^{\{2,6\}}\}}\}},$$
$$P_3(5) = 5^{\{2^{\{0^{\{1,3\}},1^{\{0,4\}}\}},6^{\{3^{\{0,7\}},4^{\{1,7\}}\}},7^{\{3^{\{0,6\}},4^{\{1,6\}}\}}\}},$$
$$P_3(6) = 6^{\{3^{\{0^{\{1,2\}},7^{\{4,5\}}\}},4^{\{1^{\{0,2\}},7^{\{3,5\}}\}},5^{\{2^{\{0,1\}},7^{\{3,4\}}\}}\}},$$
$$P_3(7) = 7^{\{3^{\{0^{\{1,2\}},6^{\{4,5\}}\}},4^{\{1^{\{0,2\}},6^{\{3,5\}}\}},5^{\{2^{\{0,1\}},6^{\{3,4\}}\}}\}}.$$

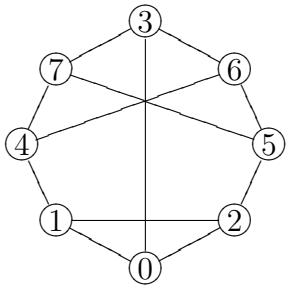

Рис. 2. Регулярный граф с $n = 8$, $s = 3$, $g = 3$

Рассмотрим теперь синтез графа с теми же значениями порядка n, степени s и диаметра d, но с бóльшим, чем в предыдущем случае, обхватом $g = 4$. Естественно, что при этом треугольные циклы будут исключены и проекция (3) преобразуется к виду

$$P_2(0) = 0^{\{1^{\{4,5,6,7\}},2^{\{4,5,6,7\}},3^{\{4,5,6,7\}}\}}.$$

Произвольным образом разместим вершины из $\{4, 5, 6, 7\}$ на 2-м уровне проекции, введя смежность вершины 1 с вершинами 4 и 5 и вершины 2 с вершинами 6 и 7, и отразим эти изменения в проекции и в списке окружений синтезируемого графа:

$$P_2(0) = 0^{\{1^{\{4,5\}},2^{\{6,7\}},3^{\{4,5,6,7\}_2}\}},$$

$$\begin{array}{ll} \mathcal{N}(0) = \{1, 2, 3\}, & \mathcal{N}(1) = \{0, 4, 5\}, \\ \mathcal{N}(2) = \{0, 6, 7\}, & \mathcal{N}(3) = \{0, \{4, 5, 6, 7\}_2\}, \\ \mathcal{N}(4) = \{1, \{3, 5, 6, 7\}_2\}, & \mathcal{N}(5) = \{1, \{3, 4, 6, 7\}_2\}, \\ \mathcal{N}(6) = \{2, \{3, 4, 5, 7\}_2\}, & \mathcal{N}(7) = \{2, \{3, 4, 5, 6\}_2\}. \end{array}$$

Число ребер $|E_k|$, задаваемых k-уровневой ($k \leqslant e(v_j)$) проекцией $P_k(v_j)$ регулярного графа $G(V, E)$ степени s, не превышает величины $s\sum\limits_{i=1}^{k}(s-1)^{i-1}$, поэтому 2-уровневой проекцией регулярного графа степени $s = 3$ в лучшем случае могут быть заданы не более 9 ребер из общего числа $|E| = 12$. Проекция $P_2(0)$ содержит в себе все 8 вершин графа, но определяет лишь 7 его ребер, поэтому для полноты проекцию следовало бы нарастить еще одним уровнем. Однако поскольку далее применяется система проекций, полнота описания графа которой обеспечивается несмотря на то, что отдельные ее проекции не обладают этим свойством и содержат неизвестные ребра, ограничимся здесь использованием проекций с двумя уровнями, достаточными для анализа эксцентриситетов ракурсных вершин и обхватов соответствующих проекциям подграфов:

$$\begin{array}{ll} P_2(0) = 0^{\{1^{\{4,5\}},2^{\{6,7\}},3^{\{4,5,6,7\}_2}\}}, & P_2(1) = 1^{\{0^{\{2,3\}},4^{\{3,5,6,7\}_2},5^{\{3,4,6,7\}_2}\}}, \\ P_2(2) = 2^{\{0^{\{1,3\}},6^{\{3,4,5,7\}_2},7^{\{3,4,5,6\}_2}\}}, & P_2(3) = 3^{\{0^{\{1,2\}},\{4,5,6,7\}_2\}}, \\ P_2(4) = 4^{\{1^{\{0,5\}},\{3,5,6,7\}_2\}}, & P_2(5) = 5^{\{1^{\{0,4\}},\{3,4,6,7\}_2\}}, \\ P_2(6) = 6^{\{2^{\{0,7\}},\{3,4,5,7\}_2\}}, & P_2(7) = 7^{\{2^{\{0,6\}},\{3,4,5,6\}_2\}}. \end{array}$$

Из $P_2(0)$ видно, что любое из потенциально возможных отношений смежности не сможет изменить эксцентриситет $e(v_0)$ ракурсной вершины. Зададимся условием равенства эксцентриситетов для всех вершин графа: $\forall v_j \in V\ (e(v_j) = d(G) = 2)$. Это условие реализуется размещением всех вершин графа не более чем на 2-х уровнях любой проекции; меньшее значение диаметра нашего графа нереализуемо в соответствии с (2). Рассмотрим также все проекции синтезируемого графа с позиции обеспечения заданного значения его обхвата $g(G)$: в любой из проекций системы сумма номеров уровней, на которых расположена одна и та же вершина, не должна быть менее обхвата. В данном случае $g(G) = 4$, и вершины 1-го уровня не должны входить в потенциальные подмножества 2-го и наоборот. Покажем это зачеркиванием соответствующих вершин в проекциях:

$$
\begin{array}{ll}
P_2(0) = 0^{\{1^{\{4,5\}},2^{\{6,7\}},3^{\{4,5,6,7\}_2}\}}, & P_2(1) = 1^{\{0^{\{2,3\}},4^{\{3,\cancel{5},6,7\}_2},5^{\{3,\cancel{4},6,7\}_2}\}}, \\
P_2(2) = 2^{\{0^{\{1,3\}},6^{\{3,4,5,\cancel{7}\}_2},7^{\{3,4,5,\cancel{6}\}_2}\}}, & P_2(3) = 3^{\{0^{\{1,2\}},\{4,5,6,7\}_2\}}, \\
P_2(4) = 4^{\{1^{\{0,5\}},\{3,\cancel{5},6,7\}_2\}}, & P_2(5) = 5^{\{1^{\{0,4\}},\{3,\cancel{4},6,7\}_2\}}, \\
P_2(6) = 6^{\{2^{\{0,7\}},\{3,4,5,\cancel{7}\}_2\}}, & P_2(7) = 7^{\{2^{\{0,6\}},\{3,4,5,\cancel{6}\}_2\}}.
\end{array}
$$

Скорректируем список окружений, удалив из них «запрещенные» вершины (в дальнейшем все коррекции будем производить без визуального зачеркивания):

$$
\begin{array}{ll}
\mathcal{N}(0) = \{1, 2, 3\}, & \mathcal{N}(1) = \{0, 4, 5\}, \\
\mathcal{N}(2) = \{0, 6, 7\}, & \mathcal{N}(3) = \{0, \{4, 5, 6, 7\}_2\}, \\
\mathcal{N}(4) = \{1, \{3, \cancel{5}, 6, 7\}_2\}, & \mathcal{N}(5) = \{1, \{3, \cancel{4}, 6, 7\}_2\}, \\
\mathcal{N}(6) = \{2, \{3, 4, 5, \cancel{7}\}_2\}, & \mathcal{N}(7) = \{2, \{3, 4, 5, \cancel{6}\}_2\}.
\end{array}
$$

Из всех висячих в $P_2(0)$ вершин (это вершины с 3 по 7) выберем вершину меньшего уровня (вершина 3) и соединим ребром с одной из вершин ее потенциального окружения $\{4, 5, 6, 7\}_2$. Выбор в данном случае может быть произвольным, так как все эти вершины расположены на одном и том же 2-м уровне и являются на данный момент висячими. Соединив ребром вершины 3 и 4, скорректируем список окружений

$$
\begin{array}{lll}
\mathcal{N}(0) = \{1, 2, 3\}, & \mathcal{N}(1) = \{0, 4, 5\}, & \mathcal{N}(2) = \{0, 6, 7\}, \\
\mathcal{N}(3) = \{0, 4, \{5, 6, 7\}_1\}, & \mathcal{N}(4) = \{1, 3, \{6, 7\}_1\}, & \mathcal{N}(5) = \{1, \{3, 6, 7\}_2\}, \\
\mathcal{N}(6) = \{2, \{3, 4, 5\}_2\}, & & \mathcal{N}(7) = \{2, \{3, 4, 5\}_2\}
\end{array}
$$

и проекции графа

$$
\begin{array}{ll}
P_2(0) = 0^{\{1^{\{4,5\}},2^{\{6,7\}},3^{\{4,\{5,6,7\}_1\}}\}}, & P_2(1) = 1^{\{0^{\{2,3\}},4^{\{3,\{6,7\}_1\}},5^{\{3,6,7\}_2}\}}, \\
P_2(2) = 2^{\{0^{\{1,3\}},6^{\{3,4,5\}_2},7^{\{3,4,5\}_2}\}}, & P_2(3) = 3^{\{0^{\{1,2\}},4^{\{1,\{6,7\}_1\}},\{5,6,7\}_1\}}, \\
P_2(4) = 4^{\{1^{\{0,5\}},3^{\{0\{5,6,7\}_1\}},\{6,7\}_1\}}, & P_2(5) = 5^{\{1^{\{0,4\}},\{3,6,7\}_2\}}, \\
P_2(6) = 6^{\{2^{\{0,7\}},\{3,4,5\}_2\}}, & P_2(7) = 7^{\{2^{\{0,6\}},\{3,4,5\}_2\}}.
\end{array}
$$

Заметим, что в проекции $P_2(0)$ вершина 4 расположена в двух разных подмножествах 2-го уровня, порожденных вершинами 1 и 3 предшествующего уровня 1[1]. Физически

[1]Это предопределено изначальным выбором попарного размещения всех вершин из $\{4, 5, 6, 7\}$ в двух подмножествах, порожденных вершинами 1 и 2 проекции $P_2(0)$. При ином размещении возможна (но не обязательна) ситуация, когда кратность размещения одной из вершин множества $\{4, 5, 6, 7\}$ равнялась бы трем, а остальных — единице; соответственно решение нашей системы проекций с неизвестными ребрами было бы другим.

это означает, что она соединена с ракурсной вершиной $v_0 = 0$ двумя маршрутами одинаковой длины $\partial(0,4) = 2$. Очевидно, что лишь одна вершина подмножества $\{5,6,7\}_1$ тоже будет дублирована на этом уровне. Логично (с позиций подобия) распространить это условие (дублирования двух вершин на 2-м уровне) на остальные проекции синтезируемого графа. Тогда вершину 3, уже дважды включенную во 2-й уровень проекции $P_2(1)$, необходимо исключить из подмножества $\{3,6,7\}_2$, порожденного вершиной 5. Это равносильно запрету отношения смежности вершин 5 и 3 и введению двух ребер, соединяющих вершину 5 с двумя вершинами из $\{6,7\}_2 = \{6,7\}$. Учитывая полученные решения, скорректируем список окружений

$$\begin{array}{lll} \mathcal{N}(0) = \{1,2,3\}, & \mathcal{N}(1) = \{0,4,5\}, & \mathcal{N}(2) = \{0,6,7\}, \\ \mathcal{N}(3) = \{0,4,\{6,7\}_1\}, & \mathcal{N}(4) = \{1,3,\{6,7\}_1\}, & \mathcal{N}(5) = \{1,6,7\}, \\ \mathcal{N}(6) = \{2,\{3,4\}_1,5\}, & & \mathcal{N}(7) = \{2,\{3,4\}_1,5\} \end{array}$$

и систему проекций

$$\begin{array}{ll} P_2(0) = 0^{\{1^{\{4,5\}},2^{\{6,7\}},3^{\{4,(6,7)_1\}}\}}, & P_2(1) = 1^{\{0^{\{2,3\}},4^{\{3,\{6,7\}_1\}},5^{\{6,7\}}\}}, \\ P_2(2) = 2^{\{0^{\{1,3\}},6^{\{\{3,4\}_1,5\}},7^{\{\{3,4\}_1,5\}}\}}, & P_2(3) = 3^{\{0^{\{1,2\}},4^{\{1,\{6,7\}_1\}},\{6,7\}_1\}}, \\ P_2(4) = 4^{\{1^{\{0,5\}},3^{\{0,\{6,7\}_1\}},\{6,7\}_1\}}, & P_2(5) = 5^{\{1^{\{0,4\}},6^{\{2,\{3,4\}_1\}},7^{\{2,\{3,4\}_1\}}\}}, \\ P_2(6) = 6^{\{2^{\{0,7\}},\{3,4\}_1,5^{\{1,7\}}\}}, & P_2(7) = 7^{\{2^{\{0,6\}},\{3,4\}_1,5^{\{1.6\}}\}}. \end{array}$$

Заметим, что и известные, и потенциально допустимые вершины окружений $\mathcal{N}(6)$ и $\mathcal{N}(7)$ в новом списке совпадают — $\mathcal{N}\{6\} = \mathcal{N}(7) = \{2,\{3,4\}_1,5\}$, что может означать правомерность любой допустимой для этих окружений подстановки[2], поэтому добавим ребро, инцидентное вершинам 7 и 3. Скорректировав окружения и проекции с учетом этой подстановки, получим искомый граф (см. рис. 3), все отношения смежности в котором полностью определены списком окружений

$$\begin{array}{lll} \mathcal{N}(0) = \{1,2,3\}, & \mathcal{N}(1) = \{0,4,5\}, & \mathcal{N}(2) = \{0,6,7\}, \\ \mathcal{N}(3) = \{0,4,7\}, & \mathcal{N}(4) = \{1,3,6\}, & \mathcal{N}(5) = \{1,6,7\}, \\ \mathcal{N}(6) = \{2,4,5\}, & & \mathcal{N}(7) = \{2,3,5\} \end{array}$$

и соответствующей этому списку системой проекций

$$\begin{array}{ll} P_2(0) = 0^{\{1^{\{4,5\}},2^{\{6,7\}},3^{\{4,7\}}\}}, & P_2(1) = 1^{\{0^{\{2,3\}},4^{\{3,6\}},5^{\{6,7\}}\}}, \\ P_2(2) = 2^{\{0^{\{1,3\}},6^{\{4,5\}},7^{\{3,5\}}\}}, & P_2(3) = 3^{\{0^{\{1,2\}},4^{\{1,6\}},7^{\{2,5\}}\}}, \\ P_2(4) = 4^{\{1^{\{0,5\}},3^{\{0,7\}},6^{\{2,5\}}\}}, & P_2(5) = 5^{\{1^{\{0,4\}},6^{\{2,4\}},7^{\{2,3\}}\}}, \\ P_2(6) = 6^{\{2^{\{0,7\}},4^{\{1,3\}},5^{\{1,7\}}\}}, & P_2(7) = 7^{\{2^{\{0,6\}},3^{\{0,4\}},5^{\{1.6\}}\}}. \end{array}$$

В отличие от малоинформативного геометрического представления полученная в результате синтеза система проекций в явном виде указывает на равенство диаметру эксцентриситетов всех вершин графа: все вершины этого графа перечислены двумя уровнями любой его проекции.

Полное описание графа может быть задано также единственной (любой из составляющих систему) проекцией наращиванием последней до полноты, например:

$$P_3(4) = 4^{\{1^{\{0^{\{2,3\}},5^{\{6,7\}}\}},3^{\{0^{\{1,2\}},7^{\{2,5\}}\}},6^{\{2^{\{0,7\}},5^{\{1,7\}}\}}\}}.$$

[2]В этом несложно убедиться: подстановка в систему проекций ребра, соединяющего вершины 3 и 6, определяет последнее неизвестное ребро, инцидентное вершинам 4 и 7.

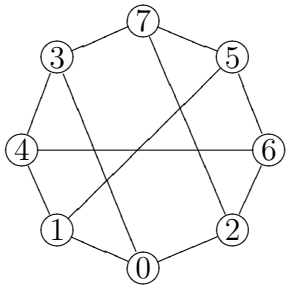

Рис. 3. Регулярный граф с $n = 8$, $s = 3$, $g = 4$

Подытожим и обобщим процесс синтеза примером графа (рис. 4) порядка $n = 16$, степени $s = 3$ и обхвата $g = 5$, полученного аналогичным образом[3].

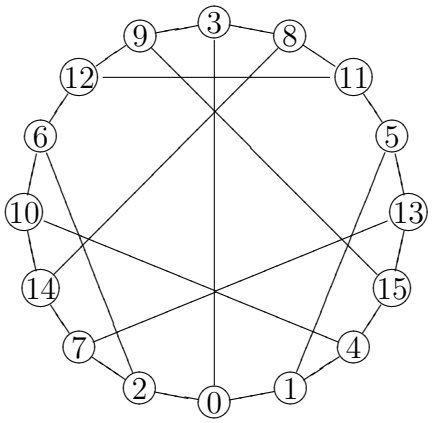

Рис. 4. Граф $n = 16, s = 3, g = 5$

1. Из (2) получим минимально возможное число уровней проекций $P_k(v_j)$, включающих в себя все вершины синтезируемого графа $G(V, E)$, обусловливающее минимальное значение его диаметра $d(G)$.

В данном случае ($k = e(v_j) = d(G) = 3$) для всех $v_j \in V$. Обхват графа $g(G)$ при этом не превышает значения $2d(G)-1$. Это значение $g(G) = 5$ принято здесь в качестве одного из условий синтеза.

2. Построим k-уровневую проекцию синтезируемого графа, выбрав в качестве ракурсной любую из произвольным образом пронумерованных вершин. Число k уровней получено на предыдущем шаге. Число $C_i(s)$ вершин на i-м уровне ($i < k$) проекции определяется выражением (1). Множество вершин k-го уровня состоит из вершин, дополняющих до V множество вершин, включенных во все предшествующие уровни, и нескольких потенциальных подмножеств вершин, число которых дополняет до $C_k(s)$ количество включенных в этот уровень известных вершин. В состав каждого подмножества включим вершины, окружения которых на текущий момент не определены полностью. Дополнительным условием включения вершины в состав подмножества является то, что сумма номера уровня, которому принадлежит подмножество, и минимального номера уровня, на котором эта вершина встречается в явном виде (не в составе потенциального подмножества), не превышает заданного обхвата.

[3]Заметим, что все полученные здесь графы являются гамильтоновыми непреднамеренно, и соответствующее этому требование в условия генерации в явном виде нами не закладывалось.

В нашем случае последний (3-й) уровень проекции состоит из шести известных вершин, дополняющих число вершин, включенных в предшествующие уровни, до $n = 13$, и шести потенциальных подмножеств, дополняющих число элементов этого уровня до $C_3(3) = 12$. В качестве ракурсной здесь взята вершина 0. Выявлено, что при общем числе ребер в графе $|E| = ns/2 = 24$ остовная проекция определяет лишь 15 из них, остальные 9 ребер предстоит определить в процессе синтеза.

3. Сформируем первоначальный список окружений вершин в графе, в состав которых входят как известные смежные вершины, так и потенциальные подмножества.

4. Используя полученный в п. 3 список окружений и учитывая принятое значение обхвата графа, выстраиваем остальные проекции системы, проводя при этом уточнение потенциальных подмножеств в каждой вновь построенной проекции и внося соответствующие изменения в список окружений графа и в построенные ранее проекции.

5. Задача синтеза графа будет решена, если список окружений его вершин не содержит потенциальных подмножеств. Если же после построения (уточнения) последней из n проекций хотя бы в одно из окружений входит потенциальное подмножество, что соответствует наличию в графе неизвестных ребер, то в одном из потенциальных подмножеств следует произвести подстановку, скорректировав затем в соответствии с п. 4 все остальные проекции и список окружений вершин графа. Заметим при этом, что подстановки, не совместные с определенными заданием условиями, делают и систему проекций несовместной. Это выражается, в частности, в том, что мощности некоторых потенциальных подмножеств в отдельных проекциях системы становятся меньше числа вершин, необходимых для определения соответствующих окружений. В этом случае следует произвести возврат к предшествующей подстановке и выбрать альтернативный в данном потенциальном подмножестве вариант.

В рассматриваемом примере в качестве первоначальной подстановки, не противоречащей заданным условиям синтеза графа, в систему проекций введено ребро, инцидентное вершинам 4 и 15. В последующем пришлось произвести еще две таких подстановки, соединив ребрами вершины 6, 10 и 5, 13.

Таким образом, в процессе синтеза данного регулярного графа потребовалось произвести три не противоречащие заданным свойствам подстановки, обусловившие размещения шести остающихся при этом неизвестными ребер.

Заключение

Формальной основой предложенного в работе подхода служит введенная автором скобочная форма представления графов в виде его проекций. Описаны принципы построения таких проекций и их основные используемые свойства. Даны оценки минимально возможного эксцентриситета ракурсной вершины и минимального числа уровней в проекции регулярного графа заданных степени и порядка, определены аналитические соотношения, связывающие эти параметры с диаметром графа и предельным значением его обхвата.

Задача синтеза регулярного графа с заданными значениями порядка, степени и обхвата сведена к абстрагированному, не требующему геометрического представления графа построению системы проекций, обладающих соответствующими заданным параметрам свойствами. Базовая остовная проекция объединяет в себе все вершины синтезируемого графа, часть которых представлена в явном (безальтернативном) виде, а часть в виде так называемых потенциальных подмножеств, дающих возможность альтернативного выбора из обладающих неполными окружениями вершин. Включение вершин в эти подмножества, их изъятие или выбор (подстановки) регламентирова-

ны определенными в работе свойствами, согласованными с параметрами графа. Дано доказательство свойства, устанавливающего критерии исключения вершины из потенциального подмножества, связанные с размещением этого подмножества в проекции (номером уровня) и с заданным обхватом графа.

Суть излагаемого подхода проиллюстрирована синтезом двух графов одного порядка и степени, но с разными обхватами. При этом аргументированы подстановки, произведенные в процессе решения системы проекций графа. Обобщенное изложение последовательности действий в процессе синтеза дано на примере синтеза графа той же степени, что и ранее рассмотренные, но бóльшего порядка.

Таким образом, в работе впервые предложен подход к детерминированному синтезу регулярных графов с заданными свойствами, которые не ограничиваются рассмотренными здесь и иллюстрирующими подход случаями генерации регулярных графов с минимальным диаметром. В качестве приоритетных могут рассматриваться и другие коммуникационные свойства: наличие гамильтонового цикла или цепи, ограничения на длины альтернативных непересекающихся маршрутов и т. д. Использование данного подхода в решении задач масштабирования систем, в том числе и нерегулярных, тоже выглядит достаточно прозрачным и перспективным. Не вызывает сомнений также и то, что разработка и внедрение аналитических методов решения перечисленных задач в теорию и практику построения отказоустойчивых систем повысит оптимальность, реактивность и предсказуемость последних.

ЛИТЕРАТУРА

1. *Valente A. X. C. N., Sarkar A., Stone H. A.* 2-Peak and 3-Peak Optimal Complex Networks // Phys. Rev. Lett. 2004. V. 92. No. 11.
2. *Мелентьев В. А.* Формальные основы скобочных образов в теории графов // Труды Второй Междунар. конф. «Параллельные вычисления и задачи управления» PACO'2004. М.: Ин-т проблем управления РАН им. В. А. Трапезникова, 2004. С. 694–706.
3. *Мелентьев В. А.* Формальный подход к исследованию структур вычислительных систем // Вестник Томского госуниверситета. Приложение. 2005. № 14. С. 167–172.
4. *Харари Ф.* Теория графов. М.: Мир, 1973. 300 с.

МАТЕМАТИЧЕСКИЕ ОСНОВЫ ИНФОРМАТИКИ И ПРОГРАММИРОВАНИЯ

DOI 10.17223/20710410/8/9

УДК 591.1

ЭЛАСТИЧНОСТЬ АЛГОРИТМОВ

В. В. Быкова

Институт математики Сибирского федерального университета, г. Красноярск, Россия

E-mail: bykvalen@mail.ru

Приведены характерные особенности эластичности субполиномиальных, полиномиальных, субэкспоненциальных, экспоненциальных и гиперэкспоненциальных классов алгоритмов. Дана методика сравнения алгоритмов по асимптотике поведения эластичности функций вычислительной сложности.

Ключевые слова: *сложность вычислений, анализ алгоритмов.*

Современная индустрия программного обеспечения и средств информационной безопасности компьютерных систем диктует необходимость развития специальных методов анализа и классификации алгоритмов. В теории сложности вычислений классификация алгоритмов традиционно осуществляется с точки зрения вычислительной сложности — трудоемкости продуцируемых алгоритмами вычислительных процессов. При этом вычислительная сложность алгоритма формально описывается функцией временной сложности $t(n)$, отражающей максимальное количество элементарных шагов, которое необходимо алгоритму для достижения запланированного результата в зависимости от n — длины входа алгоритма [1]. Обычно ограничиваются рассмотрением поведения функций сложности в асимптотике при стремлении n к бесконечности, а изложение результатов ведется в терминах O-большое и o-малое [2 – 4]. До недавнего времени алгоритмы подразделяли на низкозатратные (полиномиальные) и высокозатратные (экспоненциальные). В сегодняшней программной инженерии используется пять сложностных классов алгоритмов. Выделены субполиномиальные (быстрые) алгоритмы из полиномиального класса, субэкспоненциальные и гиперэкспоненциальные алгоритмы из экспоненциального класса. Субполиномиальные и субэкспоненциальные алгоритмы — область повышенного интереса современных криптографических систем [5 – 7]. Использование непосредственного асимптотического оценивания для распознавания всех пяти сложностных классов алгоритмов в большинстве случаев сопряжено с трудностями вычислительного характера. В данной работе в качестве меры вычислительной сложности алгоритма взята эластичность функции $t(n)$.

1. Характеристика исследуемого семейства функций сложности

Относительно функции сложности сделан ряд допущений. Во-первых, полагается, что $t(n)$ — монотонно неубывающая функция, областью значений которой выступает множество неотрицательных действительных чисел, а областью определения — множество неотрицательных целых чисел. Во-вторых, допускается отступление от дискретности изменения n (с формальной заменой n на x), т. е. предположение о том, что

аргумент x непрерывен, а необходимые значения функции $t(n)$ вычисляются в целочисленных точках $x = n$. В-третьих, анализируемое множество функций ограничивается семейством $\mathfrak{L}$ — «по-существу положительных» логарифмически-экспоненциальных функций. Установление указанных границ рассматриваемого множества функций обеспечивает существование эластичности и возможность сравнения любых двух функций сложности алгоритмов по скорости роста.

Семейство $\mathfrak{L}$-функций было введено и исследовано Г. Х. Харди [8]. Напомним, что $f(x)$ считается «по-существу положительной» функцией, если существует x_0, такое, что $f(x) > 0$ для всех $x > x_0$. Известно, что каждая $\mathfrak{L}$-функция непрерывна и дифференцируема в той области, где она определена. Теорема Харди о $\mathfrak{L}$-функциях констатирует, что эти функции образуют асимптотическую иерархию [8]: если $f(x)$, $q(x) \in \mathfrak{L}$, то при $x \to \infty$ верно одно из трех соотношений $f(x) \prec q(x)$, $f(x) \succ q(x)$, $f(x) = O[q(x)]$. Заметим, что здесь и далее

$$f(x) = O[g(x)] \Leftrightarrow f(x) \sim cq(x) \Leftrightarrow \lim_{x\to\infty} \frac{f(x)}{g(x)} = c > 0;$$

$$f(x) \prec g(x) \Leftrightarrow f(x) = o[g(x)] \Leftrightarrow \lim_{x\to\infty} \frac{f(x)}{g(x)} = 0.$$

В асимптотической иерархии $\mathfrak{L}$-функций можно выделить пять классов функций с различным порядком роста:

$$\text{Subpoly} = \left\{ f(x) \,\middle|\, f(x) \prec e^{O(\ln x)} \right\};$$

$$\text{Poly} = \left\{ f(x) \,\middle|\, f(x) = O[e^{O(\ln x)}] \right\};$$

$$\text{Subexp} = \left\{ f(x) \,\middle|\, e^{O(\ln x)} \prec f(x) \prec e^{O(x)} \right\};$$

$$\text{Exp} = \left\{ f(x) \,\middle|\, f(x) = O[e^{O(x)}] \right\};$$

$$\text{Hyperexp} = \left\{ f(x) \,\middle|\, e^{O(x)} \prec f(x) \right\}.$$

Эта классификация основана на том, что всякая $\mathfrak{L}$-функция $f(x)$ представима в виде $f(x) = e^{w(x)}$, где $w(x) = \ln f(x) \in \mathfrak{L}$, а экспоненты вида $e^{w(x)}$ подчиняются асимптотической иерархии, при этом

$$e^{w_1(x)} \prec e^{w_2(x)} \Leftrightarrow \lim_{x\to\infty} [w_1(x) - w_2(x)] = -\infty, \quad 1 \prec w_1(x) \prec w_2(x) \Rightarrow e^{w_1(x)} \prec e^{w_2(x)}. \quad (1)$$

Кроме того, для произвольных вещественных положительных констант ξ_1, ξ_2, ξ_3, τ_1, τ_2, τ_3 отношение

$$x^{\xi_1}(\ln x)^{\xi_2}(\ln\ln x)^{\xi_3} \prec x^{\tau_1}(\ln x)^{\tau_2}(\ln\ln x)^{\tau_3} \quad (2)$$

справедливо, если и только если $\xi_1 < \tau_1$, или если $\xi_1 = \tau_1$, $\xi_2 < \tau_2$, или если $\xi_1 = \tau_1$, $\xi_2 = \tau_2$, $\xi_3 < \tau_3$ [9].

2. Эластичность и ее свойства

Эластичный (гр. *elastikos*) — упругий, гибкий, легко приспособляющийся. С физической точки зрения эластичность — это свойство вещества оказывать механическое сопротивление силе, которая на него воздействует, и принимать исходную форму после спада данной силы. Изучается в теории упругости. С экономической точки зрения эластичность — это характеристика изменения одного показателя (например, спроса)

по отношению к другому показателю (например, цене товара). Используется в эконометрике для анализа производственных функций [10]. С математической точки зрения эластичность $E_x(y)$ — это коэффициент пропорциональности между темпами роста величин $y = t(x)$ и x. Формально, это дифференциальная характеристика функции $y = t(x)$, определяемая как предел отношения относительного приращения этой функции к относительному приращению аргумента [11]:

$$E_x(y) = \lim_{\Delta x \to 0} \left(\frac{\Delta y}{y} : \frac{\Delta x}{x} \right) = \frac{x}{y} \lim_{\Delta x \to 0} \frac{\Delta y}{\Delta x} = \frac{x}{y} y' = x(\ln y)' = \frac{(\ln y)'}{(\ln x)'}. \tag{3}$$

Таким образом, если $E_x(t)$ — эластичность функции временной сложности $y = t(x)$, то при повышении значения x (длины входа алгоритма) на один процент значение t (время выполнения алгоритма) увеличится приблизительно на $E_x(t)$ процентов.

Справедливы следующие свойства эластичности [11, 12].

1. Всякая постоянная имеет нулевую эластичность.
2. Эластичность — безразмерная величина: $E_x(y) = E_{ax}(by)$.
3. Эластичность обратной функции $x = f^{-1}(y)$ — обратная величина: $E_y(x) = 1/E_x(y)$
4. Эластичность произведения функций $u = u(x)$ и $w = w(x)$ равна сумме их эластичностей: $E_x(u \cdot w) = E_x(u) + E_x(w)$. Так, умножение функции на отличную от нуля константу не изменяет эластичности.
5. Эластичность отношения функций $u = u(x)$ и $w = w(x)$ равна разности их эластичностей: $E_x(u/w) = E_x(u) - E_x(w)$.
6. Эластичность суммы функций $u = u(x)$ и $w = w(x)$ — сумма эластичностей слагаемых, взятых с соответствующими весами:
   $E_x(u + w) = (u/(u + w))E_x(u) + (w/(u + w))E_x(w)$.
7. Эластичность показательно-степенной функции вида $y = u^w$, где $u = u(x)$ и $w = w(x)$, задается соотношением $E_x(y) = w(E_x(w) \ln u + E_x(u))$. В частности, $E_x[e^{w(x)}] = wE_x(w)$.
8. Эластичность композиции функций $y = f(w)$ и $w = w(x)$ равна
   $E_x(y) = E_w(f)E_x(w)$.

Непосредственное применение (3) и свойств 1–8 дает формулы эластичностей основных $\mathfrak{L}$-функций. В табл. 1 указаны асимптотические оценки эластичностей основных $\mathfrak{L}$-функций при $x \to \infty$. Здесь везде полагается, что значения x такие большие, что значения аргумента всякого логарифма и значения самого логарифма всегда остаются строго больше нуля.

Таблица 1

Эластичность основных $\mathfrak{L}$-функций

Функция $y = f(x)$	Эластичность $E_x[f(x)]$	Принадлежность к классу функций
Константа $c > 0$	0	$f(x) \in \mathrm{Subpoly}$
Каскад из k логарифмов $\underbrace{\ln\ln\ldots\ln}_{k\text{ раз}} x$, $x > 1, k = 1, 2, \ldots$	$\dfrac{1}{\ln x \cdot \ldots \cdot \underbrace{\ln\ln\ldots\ln}_{k\text{ раз}} x} = o(1)$	$f(x) \in \mathrm{Subpoly}$
Полилогарифм $(\ln x)^m$, $x > 1, m > 0$	$\dfrac{m}{\ln x} = o(1)$	$f(x) \in \mathrm{Subpoly}$
Показательный полилогарифм $e^{\lambda(\ln x)^m} = x^{\lambda(\ln x)^{m-1}}$ $x > 1, \lambda > 0, 0 < m < 1$	$\lambda m(\ln x)^{m-1} = o(1)$	$f(x) \in \mathrm{Subpoly}$
Степенная функция $e^{k(\ln x)^m} = x^{k(\ln x)^{m-1}} = x^k$ $x > 1, k > 0, m = 1$	$k = O(1)$	$f(x) \in \mathrm{Poly}$
Показательно-степенной логарифм $e^{m \ln x \ln\ln x} = x^{m \ln\ln x} =$ $= (\ln x)^{m \ln x}$, $x > 1, m > 0$	$m(1 + \ln\ln x) =$ $= O(\ln\ln x)$	$f(x) \in \mathrm{Subexp}$
Показательный полилогарифм $e^{\lambda(\ln x)^m} = x^{\lambda(\ln x)^{m-1}}$, $x > 1, \lambda > 0, m > 1$	$\lambda m(\ln x)^{m-1} = O[(\ln x)^{m-1}]$	$f(x) \in \mathrm{Subexp}$
Частный случай показательного полилогарифма $e^{\lambda x^\xi (\ln x)^{1-\xi}}$, $x > 1, 0 < \xi < 1$	$\lambda x^\xi (\ln x)^{1-\xi}(\xi + \dfrac{1-\xi}{\ln x}) =$ $= O[x^\xi (\ln x)^{1-\xi}]$	$f(x) \in \mathrm{Subexp}$
Экспонента конечного порядка роста $e^{\lambda x^k}$, $x > 0, \lambda > 0, k > 0$	$\lambda k x^k = O(x^k)$	$f(x) \in \mathrm{Subexp}, 0 < k < 1$ $f(x) \in \mathrm{Exp}, k = 1$ $f(x) \in \mathrm{Hyperexp}, k > 1$
«Башня» $\underbrace{2^{2^{\cdot^{\cdot^{\cdot^{2^x}}}}}}_{k\text{ этажей}}$, $k \geqslant 1$	$k 2^x 2^{2^x} \cdots \underbrace{2^{2^{\cdot^{\cdot^{\cdot^{2^x}}}}}}_{k-1\text{ этажей}} (\ln 2)^k$	$f(x) \in \mathrm{Exp}, k = 1$ $f(x) \in \mathrm{Hyperexp}, k > 1$

3. Классы логарифмически-экспоненциальных функций и их эластичности

В работе [12] доказана теорема, устанавливающая характеризацию эластичности для пяти классов $\mathfrak{L}$-функций.

Теорема 1 (о классификации $\mathfrak{L}$-функций) [12]. Разбиение семейства монотонно неубывающих «по-существу положительных» $\mathfrak{L}$-функций на классы Subpoly, Poly, Subexp, Exp, Hyperexp в соответствии с порядком их роста эквивалентно надлежащему разбиению по асимптотике эластичности этих функций на бесконечности:

$$\text{Subpoly} = \left\{ f(x) \,\middle|\, f(x) \prec e^{O(\ln x)} \right\} \equiv \left\{ f(x) \,|E_x(f) = o(1) \right\}; \tag{4}$$

$$\text{Poly} = \left\{ f(x) \,\middle|\, f(x) = O[e^{O(\ln x)}] \right\} \equiv \left\{ f(x) \,|E_x(f) = O(1) \right\}; \tag{5}$$

$$\text{Subexp} = \left\{ f(x) \,\middle|\, e^{O(\ln x)} \prec f(x) \prec e^{O(x)} \right\} \equiv \left\{ f(x) \,|1 \prec E_x(f) \prec x \right\}; \tag{6}$$

$$\text{Exp} = \left\{ f(x) \,\middle|\, f(x) = O[e^{O(x)}] \right\} \equiv \left\{ f(x) \,|E_x(f) = O(x) \right\}; \tag{7}$$

$$\text{Hyperexp} = \left\{ f(x) \,\middle|\, e^{O(x)} \prec f(x) \right\} \equiv \left\{ f(x) \,|x \prec E_x(f) \right\}. \tag{8}$$

Из данной теоремы и свойств эластичности вытекают важные следствия.

Следствие 1. Разбиение $\mathfrak{L}$-функций на классы Subpoly, Poly, Subexp, Exp, Hyperexp инвариантно относительно полиномиального преобразования, т. е. если $f(x) \in \mathfrak{F}$, $\mathfrak{F} \in \{\text{Subpoly, Poly, Subexp, Exp, Hyperexp}\}$, то также $p(f(x)) \in \mathfrak{F}$, где $p(x) \in \text{Poly}$.

Действительно, по свойству 8 имеем: $E_x[p(f(x))] = E_f(p)Ex(f) = O(1)Ex(f)$. Согласно (4) – (8), умножение эластичности на асимптотическую константу не меняет принадлежность $\mathfrak{L}$-функции к классу $\mathfrak{F} \in \{\text{Subpoly, Poly, Subexp, Exp, Hyperexp}\}$.

Следствие 2. Класс Subpoly замкнут относительно суперпозиции (композиции) $\mathfrak{L}$-функций, т. е. если $f(x), q(x) \in \text{Subpoly}$, то $q(f(x)) \in \text{Subpoly}$.

Следствие 3. Класс Poly замкнут относительно суперпозиции $\mathfrak{L}$-функций, т. е. если $f(x), q(x) \in \text{Poly}$, то $q(f(x)) \in \text{Poly}$.

Следствие 4. Класс Hyperexp замкнут относительно суперпозиции $\mathfrak{L}$-функций, т. е. если $f(x), q(x) \in \text{Hyperexp}$, то $q(f(x)) \in \text{Hyperexp}$.

Композиция $\mathfrak{L}$-функций из классов Subexp, Exp может изменить их принадлежность к этим классам.

4. Систематизация алгоритмов

Теорема о классификации $\mathfrak{L}$-функций позволяет формально описать пять современных сложностных классов алгоритмов. Класс быстрых алгоритмов — множество алгоритмов с функциями сложности $t(x) \in \text{Subpoly}$. Таким алгоритмам присуща тождественно нулевая или бесконечно малая эластичность. Класс полиномиальных алгоритмов — множество алгоритмов с $t(x) \in \text{Poly}$ и асимптотически постоянной эластичностью $E_x(t)$. Класс субэкспоненциальных алгоритмов — алгоритмы, для которых $t(x) \in \text{Subexp}$. Эластичность $E_x(t)$ субэкспоненциального алгоритма — бесконечно большая величина, такая, что $1 \prec E_x(t) \prec x$. Для такого алгоритма темп роста времени выполнения значительно выше темпа роста длины входа. Класс экспоненциальных алгоритмов — это алгоритмы, для которых $t(x) \in \text{Exp}$. Для них эластичность $E_x(t) = O(x)$ — бесконечно большая величина, асимптотически пропорциональная линейной функции. Функции с подобной эластичностью описывают законы естественного

роста: скорость увеличения такой функции прямо пропорциональна ей самой. Класс гиперэкспоненциальных алгоритмов — это алгоритмы, для которых $t(x) \in \text{Hyperexp}$ и $x \prec E_x(t)$. Темп роста гиперэкспоненциальных функций настолько высок, что не укладывается в законы естественного роста.

Исходя из следствий 1 – 4, классификации алгоритмов на основе асимптотического поведения эластичности функций сложности присущи следующие практически значимые особенности:

- инвариантность относительно модели вычислений, поскольку переход от одной модели вычислений к другой меняет вычислительную сложность алгоритма полиномиальным образом. Это отвечает традиции измерять вычислительную сложность алгоритма с точностью до $O(1)$ и сопоставлять алгоритмы с точностью до полинома [1];
- неизменность сложностного класса алгоритма при полиномиальном преобразовании входа алгоритма. Подобные преобразования могут возникать при учете в модели вычислений времени доступа к исходным данным алгоритма;
- суперпозиция быстрых алгоритмов приводит к быстрому алгоритму;
- суперпозиция полиномиальных алгоритмов приводит к алгоритму полиномиальной сложности.

5. Методика сравнения алгоритмов по асимптотическому поведению эластичности

Если необходимо установить класс, к которому принадлежит алгоритм с функцией сложности $t(n) \in \mathfrak{L}$, то следует выполнить следующие действия:

- осуществить формальный переход от $t(n)$ к $t(x)$, т. е. от дискретного аргумента n к непрерывному x;
- вычислить $E_x(t)$ и найти асимптотическую оценку для $E_x(t)$ при $x \to \infty$;
- определить класс функций, используя эквивалентности (4) – (8).

Пусть требуется сравнить алгоритмы α_1 и α_2, вычислительная сложность которых описывается функциями $t_1(x)$ и $t_2(x)$ соответственно. Для этого сначала надлежит установить классы сложности для α_1 и α_2. Если данные алгоритмы принадлежат разным классам сложности, то иерархия этих классов задает соответствующее отношение между α_1 и α_2 в смысле их быстродействия. Если оказалось, что алгоритмы α_1 и α_2 принадлежат одному классу, то многое зависит от самого этого класса:

- если $t_1(x), t_2(x) \in \text{Subpoly, Subexp, Hyperexp}$, то при $E_x(t_1) \prec E_x(t_2)$ всегда $t_1(x) \prec t_2(x)$, т. е. алгоритм α_1 асимптотически быстрее алгоритма α_2;
- если $t_1(x), t_2(x) \in \text{Poly, Exp}$, то $E_x(t_1) \sim cE_x(t_2), c > 0$. При большой длине входа и $0 < c < 1$ время выполнения алгоритма α_1 меньше времени работы алгоритма α_2 приблизительно в $1/c$ раз. При $c > 1$, наоборот, алгоритм α_1 работает медленнее алгоритма α_2 примерно в c раз. При $c = 1$ требуется исследование в $E_x(t_1), E_x(t_2)$ членов более низкого порядка, нежели константа.

Проиллюстрируем последний случай. Рассмотрим два алгоритма, осуществляющих операцию умножения двух длинных n-разрядных целых чисел: алгоритм Шенхаге — Штрассена, имеющий функцию сложности

$$t_1(n) = c_1 n \ln n \ln\ln n,$$

и алгоритм Тоома — Кука, характеризующийся трудоемкостью

$$t_2(n) = c_2 n e^{(2\ln n)^{1/2}} \ln n,$$

где $c_1 > 0$, $c_2 > 0$ — некоторые константы [4]. Если выполнить формальный переход от n к x, то $t_1(x)$, $t_2(x)$ — «по-существу положительные» монотонно возрастающие $\mathfrak{L}$-функции. Определим для них эластичности:

$$E_x(t_1) = E_x(c_1 x \ln x \ln\ln x) = E_x(c_1) + E_x(x) + E_x(\ln x) + E_x(\ln\ln x) =$$
$$= 0 + 1 + \frac{1}{\ln x} + \frac{1}{\ln x \ln\ln x} = 1 + \gamma_1(x) > 0,$$
$$E_x(t_2) = E_x(c_2 x e^{(2\ln x)^{1/2}} \ln x) = E_x(c_2) + E_x(x) + E_x(e^{(2\ln n)^{1/2}}) + E_x(\ln x) =$$
$$= 0 + 1 + \frac{\sqrt{2}}{(\ln x)^{1/2}} + \frac{1}{\ln x} = 1 + \gamma_2(x) > 0.$$

Поскольку $\gamma_1(x) = o(1)$, $\gamma_2(x) = o(1)$, то $t_1(x), t_2(x) \in \mathrm{Poly}$ и $E_x(t_1) \sim E_x(t_2)$. Выполним сравнение бесконечно малых величин $\gamma_1(x) > 0$, $\gamma_2(x) > 0$ при $x \to \infty$:

$$\lim_{x\to\infty} \frac{\gamma_1(x)}{\gamma_2(x)} = \lim_{x\to\infty} \left(\frac{\ln\ln x + 1}{\ln x \ln\ln x} : \frac{(2\ln x)^{1/2} + 1}{\ln x} \right) = \lim_{x\to\infty} \left(\frac{\ln\ln x + 1}{\ln x \ln\ln x} \cdot \frac{\ln x}{(2\ln x)^{1/2} + 1} \right) =$$
$$= \lim_{x\to\infty} \left(\frac{\ln\ln x + 1}{\ln\ln x} \cdot \frac{1}{(2\ln x)^{1/2} + 1} \right) = \lim_{x\to\infty} \left(1 + \frac{1}{\ln\ln x} \right) \cdot \left(\frac{1}{(2\ln x)^{1/2} + 1} \right) = 1 \cdot 0 = 0.$$

Отсюда $\gamma_1(x) \prec \gamma_2(x)$ и $E_x(t_1) \prec E_x(t_2)$ при $x \to \infty$. Следовательно, алгоритм Шенхаге — Штрассена асимптотически быстрее алгоритма Тоома — Кука. Как следует из данного примера, в ряде случаев при сравнении алгоритмов не удается избежать непосредственного сравнения функций при помощи O-большое и o-малое. Это касается полиномиальных и экспоненциальных алгоритмов с асимптотически пропорциональными эластичностями. Здесь могут оказаться полезными соотношения (1), (2), относящиеся к иерархии $\mathfrak{L}$-функций.

6. Сложность и эластичность теоретико-числовых алгоритмов

Разработка и анализ теоретико-числовых алгоритмов — предмет исследований алгоритмической теории чисел, имеющей приложения в криптографии [6, 7]. В криптографии (для обоснования стойкости криптографических систем и для разработки методов их вскрытия) важны следующие методы и алгоритмы [5]: тесты на простоту целых чисел, методы факторизации (разложения целых чисел на множители), алгоритмы дискретного логарифмирования, алгоритмы выполнения арифметических операций с длинными целыми числами, алгоритмы полиномиальной арифметики и др.

Традиционно функции сложности теоретико-числовых алгоритмов являются функциями от n — количества двоичных разрядов (битов), требуемых для записи исходного длинного целого числа N. Таким образом, $n = O(\log_2 N) = O(\ln N)$. В табл. 2 приведены классы сложности и эластичности наиболее известных тестов на простоту и алгоритмов факторизации длинных целых чисел. Функции сложности этих алгоритмов взяты из работы [5]. В функциях сложности осуществлен формальный переход от дискретного аргумента n к непрерывному x.

Таблица 2

Эластичность некоторых теоретико-числовых алгоритмов

Алгоритм	Функция сложности $t(x)$ алгоритма	Эластичность $E_x[t(x)]$ алгоритма	Класс сложности алгоритма
Тест на простоту Конягина — Померанса	$O(1)\dfrac{x^{17/7}}{\ln x}$	$o(1)+\dfrac{17}{7}-\dfrac{1}{\ln x}=$ $=O(1)$	Poly
Тесты на простоту Адлемана — Померанса — Румели и Ленстры	$O(1)x^{c\ln\ln x}, c>0$	$o(1)+$ $+c(1+\ln\ln x)=$ $=O(\ln\ln x)$	Subexp
Тест на простоту Аггравала — Кайала — Саксены	$O(1)x^{12}(\ln x)^c, c>0$	$o(1)+12+\dfrac{c}{\ln x}=$ $=O(1)$	Poly
Факторизация целых чисел по методу Диксона	$e^{cx^{\xi}(\ln x)^{1-\xi}}$ $\xi=\frac{1}{2}, c>0$	$o(1)+cx^{1/2}(\ln x)^{1/2}\cdot$ $\cdot(\dfrac{1}{2}+\dfrac{1}{2\ln x})=$ $=O\left[x^{1/2}(\ln x)^{1/2}\right]$	Subexp
Факторизация целых чисел по методу GNFS (general number field sieve)	$e^{cx^{\xi}(\ln x)^{1-\xi}}$ $\xi=\dfrac{1}{3}, c>0$	$o(1)+cx^{1/3}(\ln x)^{1/3}\cdot$ $\cdot(\dfrac{1}{3}+\dfrac{2}{3\ln x})=$ $=O\left[x^{1/3}(\ln x)^{2/3}\right]$	Subexp
Факторизация целых чисел по методу Шермана — Лемана	$O(1)e^{x/3}$	$o(1)+x/3=O(x)$	Exp
Факторизация целых чисел по методу Полларда — Штрассена	$O(1)x^4e^{x/4}$	$o(1)+4+x/4=O(x)$	Exp

Заключение

Классификация алгоритмов на основе асимптотического поведения эластичности функций сложности не разрушает прежней, традиционной классификации с полиномиальными и экспоненциальными алгоритмами, а лишь дополняет и уточняет ее. Свойства эластичности позволяют без особого труда находить эластичность для любой $\mathfrak{L}$-функции. К ограничениям рассмотренной классификации следует отнести требование принадлежности функций сложности алгоритмов к семейству Харди. Однако в большинстве реальных случаев это требование является вполне естественным и не вызывает особых трудностей при анализе алгоритмов.

Представленные в работе результаты (следствия 1 – 4, методика сравнения алгоритмов по асимптотическому поведению эластичности, формулы эластичностей для основных $\mathfrak{L}$-функций и теоретико-числовых алгоритмов) могут быть полезны в определении стойкости современных криптографических систем и для разработки методов их вскрытия.

ЛИТЕРАТУРА

1. *Юдин Д. Б., Юдин А. Д.* Математики измеряют сложность. М.: Книжный дом «Либроком», 2009. 192 с.
2. *Быкова В. В.* Математические методы анализа рекурсивных алгоритмов // Журнал СФУ. Математика и физика. 2008. № 1(3). С. 236–246.
3. *Гэри М., Джонсон Д.* Вычислительные машины и труднорешаемые задачи. М.: Мир, 1982. 416 с.
4. *Кормен Т., Лейзерсон Ч., Ривест Р.* Алгоритмы: построение и анализ. М.: МЦНМО, 1999. 960 с.
5. *Василенко О. Н.* Теоретико-числовые алгоритмы в криптографии. М.: МЦНМО, 2006. 336 с.
6. *Чмора А. Л.* Современная прикладная криптография. М.: Гелиос АРВ, 2001. 256 с.
7. *Варновский Н. П.* Криптография и теория сложности // Математическое просвещение. 1998. Сер. 3. Вып. 2. С. 71–86.
8. *Харди Г. Х.* Курс чистой математики. М.: ИЛ, 1949. 512 с.
9. *Грэхем Р., Кнут Д., Поташник О.* Конкретная математика. М.: Мир; Бином. Лаборатория знаний, 2006. 703 с.
10. *Доугерти К.* Введение в эконометрику. М.: ИНФРА-М, 2001. 402 с.
11. *Солодовников А. С., Бабайцев В. А., Браилов А. В., Шандра И. Г.* Математика в экономике. М.: Финансы и статистика, 2001. 376 с.
12. *Быкова В. В.* Метод распознавания классов алгоритмов на основе асимптотики эластичности функций сложности // Журнал СФУ. Математика и физика. 2009. № 2(1). С. 48–61.

2010 Математические основы информатики и программирования №2(8)

DOI 10.17223/20710410/8/10

УДК 681.3

ОБ ОБНАРУЖЕНИИ ОШИБОЧНОЙ РАБОТЫ С РЕСУРСАМИ В ПРОГРАММНОМ ОБЕСПЕЧЕНИИ[1]

В. В. Горелов

Томский государственный университет, г. Томск, Россия

E-mail: skylark@mail.tsu.ru

Предлагается метод обнаружения и при необходимости предотвращения ошибочной работы программы с ресурсами. Количество ресурсов, с которыми может работать система, теоретически не ограничено ввиду применения разработанной схемы по описанию широкого спектра ресурсов и функций по работе с ними. Анализируются операции, влияющие на надёжность, безопасность и оптимальность использования ресурсов в программном обеспечении.

Ключевые слова: *программное обеспечение, инструментирование, ресурсы, обнаружение ошибок.*

Введение

Как известно, подавляющее большинство программного обеспечения состоит из алгоритмов преобразования информации и механизмов использования разнообразных ресурсов для получения, обработки и передачи информации. Со стороны программного обеспечения в процессе использования ресурсов (и, как показывает практика, довольно часто) возникают ситуации опасных действий с ними. Было замечено, что для ряда ресурсов встречаются одни и те же шаблонные ситуации ошибочного использования. Предлагается метод автоматизированного обнаружения таких ситуаций; с его помощью создана экспериментальная система по выявлению опасных участков в программе. Результатом работы системы является информация о конкретных местах и типах найденных дефектов в программе. Принципиальной особенностью обнаружения является отделение самой системы от типов ресурсов, с которыми она работает. Такой подход необходим для распознавания шаблонных ошибок для теоретически неограниченного количества ресурсов без внесения изменений в текст программы либо с незначительными изменениями для некоторых нестандартных классов ресурсов. Таким образом, предложенная система состоит из трёх частей:

1) языка для описания ресурсов и функций по работе с ними;
2) программы-преобразователя, которая на вход принимает описание ресурсов и создаёт необходимый код по перехвату функций, работающих с интересующими ресурсами;
3) программы-анализатора, которая считывает соответствующие события, возникающие в процессе работы отлаживаемой программы, с дальнейшим выводом обнаруженных опасных участков для анализа человеком.

Типы обнаруживаемых ошибок

К основным опасным ситуациям при использовании ресурсов отнесены следующие состояния: утечка ресурсов; использование ресурсов после их освобождения; повторные освобождения ресурсов; использование (неинициализорованных) ресурсов без их

[1]Работа выполнена в рамках реализации ФЦП «Научные и научно-педагогические кадры инновационной России» на 2009–2013 годы. (Гос. контракт № П1010.)

предварительного захвата; использование ресурсов за их границами (относится к динамической памяти и адресному пространству) и другие. Некоторые из перечисленных ситуаций особо опасны, поскольку могут привести к исполнению произвольного кода, предварительно изготовленного злоумышленником, в адресном пространстве отлаживаемой программы.

Область применения и особенности реализации

Рассматриваемый метод реализован для языка программирования Си, однако при необходимости расширения системы возможна поддержка и других языков программирования. В настоящее время реализация [1] системы работает с прикладным программным обеспечением в сетевом и отсоединённом режимах. Под сетевым режимом подразумевается ситуация, когда отлаживаемая программа и программа-анализатор поступающих событий обмениваются сообщениями через TCP/IP-сеть. В отсоединённом режиме отлаживаемая программа автоматически протоколирует возникающие события в файл, который впоследствии передаётся на вход к программе-анализатору для создания отчёта, пригодного для чтения человеком. Система поддерживает 16-, 32- и 64-битные режимы адресации отлаживаемых POSIX- и Windows-программ. Программа-преобразователь и программа-анализатор работают в средах POSIX и Windows в 32- и 64-битных режимах. Сборка системы проверена с компиляторами GCC [2] и Microsoft Visual Studio [3]. Поскольку система обнаружения кроссплатформенная, вероятно, и другие компиляторы могут быть использованы. Единственной используемой внешней библиотекой является `libxml++` [4].

1. Структурное представление ресурсов

Для обеспечения раздельного описания ресурсов от системы обнаружения предусмотрен специальный язык, с помощью которого независимый разработчик может в довольно простом текстовом виде описать ресурсы, которые необходимо отследить. Формат описания ресурсов представляется в специально-оформленном XML-файле [5] с заполнением соответствующих полей и атрибутов.

1.1. Настройки режима работы

Данные настройки указываются один раз в начале файла-шаблона, который описывает ресурсы и функции по работе с ними. Далее перечислены атрибуты с описанием их значения.

— `format`: порядковый номер формата шаблона. Необходим для того, чтобы программа-преобразователь могла проверить, является ли переданный ей для обработки файл именно такого формата, который она понимает.
— `name`: название шаблона, отражающее его содержимое и предназначение; данная информация впоследствии помещается в создаваемые файлы с целью определения (при необходимости), по какому шаблону они были созданы.
— `prefix`: задаёт префикс для замещаемых функций. Требуется для исключения возможных столкновений между замещаемыми функциями с префиксом и существующими функциями с такими же именами. Например, для префикса «`art_`» и функции захвата памяти `malloc()` препроцессором будет добавлено определение функции `art_malloc()`, однако в пространстве имён отлаживаемого проекта может присутствовать функция с таким же именем. Для разрешения данной ситуации и предусмотрена возможность задать уникальный префикс.
— `errlogmode`: указывает, как реагировать служебному коду проекта в случае обнаружения невозможности протоколировать возникающие события. Имеет два за-

резервированных значения: «» — полностью игнорировать, «`console`» — выводить информацию через устройство вывода. Любое другое значение будет использовано как имя файла, в который будет произведена попытка записи информации о сбое протоколирования.

— `trap_on_io_err`: принимает значение истина или ложь. Влияет на поведение созданного кода при выявлении невозможности протоколирования. В первом случае приводит к записи информации об этом тем способом, который задан пунктом `errlogmode`. В противном случае ошибка игнорируется и осуществляется попытка продолжить работу.
— `remote_mode`: принимает значение истина или ложь. В первом случае даёт указание создавать код для работы в сетевом режиме. В противном случае создаётся код, записывающий события в файл протокола.
— `force_flush`: принимает значение истина или ложь. В первом случае даёт указание сбрасывать буфера при каждой операции записи или пересылки событий. Вне процесса отладки должно быть всегда установлено в значение истина. В противном случае будет происходить буферизирование передаваемой информации и в случае ошибки при её передаче часть информации может быть потеряна. Из-за этого программа-анализатор будет лишена важной информации, которая, вероятно, и привела к невозможности передать очередную порцию данных (например, код нарушил работу ядра операционной системы).
— `multithreaded`: принимает значение истина или ложь. В первом случае даёт указание создавать обёрточный код для блокирования одновременного вызова функций, за которыми ведётся наблюдение. Данное значение необходимо для многопоточных программ. Для программ, работающих в синхронном режиме, это значение должно быть задано как ложь.
— `threading`: указывает, какой создавать код для случая, когда отлаживается многонитевая программа. Поддерживаемые значение: `posix` и `win32` — для соответствующих сред.
— `compiler_type`: указывает, примитивы какого компилятора использовать. Принимает значения `gcc` и `win32` для соответствующих сред.
— `trace_target`: строка, которая для сетевого режима содержит адрес и порт, на который будут отсылаться события. Для отсоединённого режима данное поле означает имя файла, в который события будут записаны.

На этом настройки режима работы заканчиваются. Далее рассматривается непосредственно описание ресурсов и функций, которые с ними работают.

1.2. Шаблон ресурса

В файле-шаблоне для описания каждого отдельного ресурса применяется элемент `domain` с иерархическим включением других элементов. Элементы могут быть простыми — строки и двоичные значения — и составными, содержащими, в свою очередь, другие элементы. Таким образом, в шаблоне представлено дерево, описывающее все интересующие ресурсы, функции и прочую необходимую для создания кода информацию. Далее перечислены атрибуты элемента `domain`:

— `name`: человекопригодное имя для ресурса; например, для памяти, захватываемой из кучи, можно присвоить значение `heap memory`.
— `float_handle`: принимает значение истина или ложь. На этом атрибуте стоит остановиться подробнее. Все ресурсы с точки зрения системы по этому атрибуту делятся на два класса: так называемые «плавающие ресурсы» и «цельные». Под пер-

выми понимается класс ресурсов, к которым можно обращаться как к некой размерности. Это память и адресное пространство. Под вторым — ресурсы, которые не обладают данным свойством. В качестве примера можно привести дескриптор типа `int`, файловый дескриптор `FILE*` и т. д. Такое разделение введено для поддержки отслеживания границ адресных пространств и оценки эффективности их использования. Под последним имеется в виду, насколько захваченный ресурс был использован, как именно и был ли использован вообще.

— `handle`: тип ресурса для целевого языка программирования. Зависит от ресурса и может быть произвольным, например `void*`, `int`, `FILE*`, `some_type_t` и т. д.

— `includesGlobal`: принимает значение истина или ложь. Указывает, как именно препроцессор должен включать заголовочные файлы с определением ресурсов и функций. В первом случае используется для ресурсов, доступных в стандартных каталогах поиска заголовочных файлов (например, в среде `UNIX` это `/usr/include`). В противном случае включаются локальные заголовочные файлы. Последний вариант необходим при слежении за ресурсами, которые описаны и используются в самой отлаживаемой программе и не входят в стандартные библиотеки и API-системы.

— `includes`: представляет векторный тип с перечислением самих заголовочных файлов. Для ресурса типа «память из кучи» это будет вектор с одном элементом `stdlib.h`. Для своего собственного ресурса это может быть `myResource.h`.

— `bad_handle`: элемент, описывающий выражение, представляющее неверный ресурс. Необходим при создании кода, распознающего правильность ресурса. Например, для указателя это NULL, для дескриптора — -1. Однако самого значения недостаточно для проверки, является ли указатель «правильным». Далее отдельно рассмотрим атрибуты элемента `expr`.

1.3. Представление выражений

Элемент выражения `expr` служит для определения истинности некоторого значения с применением заданного критерия сравнения и состоит из следующих простых атрибутов:

— `oper`: строка, отражающая саму операцию. Например, `==`, `!=`, `<`.

— `type`: описывает имя типа, используемого в сравнении. Например, `int`, `ssize_t`, `gint`.

— `value`: произвольное значение пользователя для операции сравнения: 0, `NULL`, -1, `ERR_OK` и т. п.

Таким образом, для произвольного ресурса возможно следующее представление:
`type` t = «некоторое значение»;
`if`(t `oper value`) {код обработки истинности} `else` {код обработки неистинности}.

1.4. Представление операторов

Рассматриваются операторы по захвату, перераспределению, использованию и освобождению ресурса. Для их представления задаются элементы `tor`, каждый из которых имеет следующие атрибуты:

— `type`: принимает значение в зависимости от типа оператора (`allocator`, `reallocator`, `operator`, `deallocator`).

— `name`: имя оператора в рамках языка программирования. Например, `malloc`, `fopen`, `myAllocation`.

— `float_arg_n1`: неотрицательная величина, указывающая номер первого аргумента, отвечающего за задание размерности или смещения. Актуально только для «плавающих ресурсов».
— `float_arg_n2`: неотрицательная величина, указывающая номер второго (если есть) аргумента, отвечающего за задание размерности или смещения. Актуально только для «плавающих ресурсов».
— `validateDom_before`: векторная величина, указывающая, какие аргументы проверять по критерию правильности соответствующего ресурса. В качестве значений используется заданное в начале описания `domain` имя ресурса, например `heap memory`.
— `is_handle_arg_out`: принимает значение истина, если у функции существует возвращаемый идентификатор ресурса (для функций захвата и перераспределения ресурсов), или ложь — в этом случае следующий атрибут игнорируется.
— `handle_arg_out`: неотрицательное значение, указывающее номер выходного идентификатора ресурса: 0 — возвращаемый аргумент, 1 — первый аргумент функции и т. д.
— `args`: набор элементов, который служит для описания возвращаемого аргумента функции и аргументов самой функции. Каждый элемент `args` содержит тип переменной и её имя. Для возвращаемого значения имя не указывается.
— `badRetCode`: описывает выражение для распознавания возвращаемого значения функции, если оно есть. Распознаются значения «удачно» и «отказ» при проведении всех операций. Введение данного выражения необходимо в силу того, что формат возвращаемого статуса операции может быть представлен как через специально зарезервированное значение идентификатора ресурса, так и через раздельное использование ресурса и возвращаемого значения.

2. Преобразование описания ресурсов в код для инструментирования

Целью программы является преобразование описания ресурсов из файла-шаблона в исходный код на языке Си. На вход программе подаётся имя файла-шаблона и имена двух файлов, в которые должен быть помещён создаваемый код на Си. Поскольку файл-шаблон выражен в форме XML, то для его разбора используется библиотека `libxml++` (которая, в свою очередь, — удобная обвязка на языке Си++ для библиотеки `libxml2` [6], реализованной на языке программирования Си). Поскольку программе не требуется изменять содержимое шаблона, использован событийный разборщик (SAX parser). Для связи библиотечного движка разбора с самой программой преобразования реализованы классы, получающие события от библиотеки. Принятые события распознаются как элементы, атрибуты проверяются на логическое соответствие внутренней схеме представления. В процессе обхода файла-шаблона происходит проверка и формирование узлов шаблона, но уже в рамках языка программирования Си++, с применением его родных типов. Второй аргумент — создаваемый заголовочный файл, в который помещаются директивы препроцессора и определения замещаемых функций. В последнем аргументе — имя файла, куда помещаются тела реализации этих функций. Вызов преобразователя может быть осуществлён, например, так:

```
artgen posix-gcc-mt-file-lint.xml art.h art.c
```

В случае удачного обхода дерева описания из файла-шаблона управление передаётся части непосредственного создания кода, который будет использован для инструментирования отлаживаемой программы.

Последующее применение созданых файлов для отлаживаемой программы `main.c` на примере компилятора `GCC`:
`gcc -c art.c` (создаёт объектный файл `art.o`)
`gcc -include art.h main.c art.o -o main` (создаёт двоичный файл `main`).

2.1. Фаза создания заголовочного кода

Первое, что добавляется в файл — стандартная защита от повторного включения, а также указание препроцессору компилятора Си++, что все дальнейшие строки текста являются определениями на языке Си. Это позволяет использовать систему не только в чистых проектах на Си, но и в чистых Си++ или смешанных проектах на Си/Си++. Далее поведение кодогенератора практически полностью управляется значениями, считанными из шаблона, и, в зависимости от значений последнего, происходит обход дерева и комбинирование заготовок кода и логических конструкций из библиотеки кодогенератора.

Далее происходит подключение заголовочных файлов для поддержки многонитевости, если так указано в шаблоне. После этого подключаются заголовочные файлы отлаживаемых ресурсов. За ними следуют служебные определения функций системы. Основные из них следующие:

— `void art_trace(int n, ...)`: при вызове протоколирует n-е количество аргументов соответствующим образом (в файл или через сеть);
— `void art_start(char *appname)`: в случае необходимости (например, если задана отладка многонитевой программы) производит инициализацию служебных объектов синхронизации (`mutex`) и других переменных. Эта функция должна быть добавлена первой функцией в отлаживаемую программу;
— `void art_stop()`: функция автоматически вызывается последней при неаварийном завершении работы программы. Вызов этой функции регистрируется в функции `art_start()` через механизм `atexit()`.

Определение функций, которые будут вызываться в процессе работы программы, происходит следующим образом:

```
префикс-оригинальное название функции(оригинальные аргументы, char* file,
size_t line);
```

Последние два аргумента преследуют цель запротоколировать информацию о месте вызова в отлаживаемой программе.

Следующим блоком идут определения для препроцессора, который и проделает работу по замене вызовов оригинальных функций на «перехваченные»:

```
#define origName(zzz) префикс-системы_origName(zzz, __FILE__, __LINE__)
```

Последние два аргумента также являются ключевыми словами для препроцессора, в которые он подставит имя файла и номер строки, где произведена замена вызова оригинальной функции на функцию системы отладки.

2.2. Фаза создания тел реализации для инструментируемых функций

Первым создаётся код для служебных функций системы отладки, они относительно немногочисленны и носят характер поддержки сети, файловых дескрипторов, объектов синхронизации, функций инициализации, преобразования чисел в строки и т. д. Стоит остановиться лишь на паре ключевых служебных функций:

— `art_start()`: кроме прочих служебных функций, протоколирует описания ресурсов и функций по работе с ними в заголовке протокола. Эта информация необходима в фазе анализа;

— `art_stop()`: протоколирует информацию о том, что программа завершилась. Без этой информации на стороне сервера весьма затруднительно понять, пора делать выводы об утечке ресурсов или нет.

Работа по созданию инструментируемых функций разделена на четыре этапа.

1) *Пролог.* Заключается в создании начальной части кода с заданием и инициализацией вспомогательных переменных в соответствии с указаниями из шаблона. Создаваемый флаг (`mutex`) необходим для синхронизации протоколирования информации, поступающей от разных нитей. Применение его обязательно, поскольку запись в файл или гнездо должна быть произведена в определённом порядке, с сохранением очерёдности содержимого транзакций. Кроме того, протоколируется информация о типе ресурса и обрабатываемой функции с указанием имени файла и номера строки, откуда произведён вызов.
2) *Проверка.* По информации из шаблона создаётся код, проверяющий правильность переданных аргументов. Результат этой проверки протоколируется.
3) *Вызов оригинальной функции.* В соответствии с информацией из шаблона для произвольной функции происходит запись интересующих выходных данных.
4) *Эпилог.* Создаётся код для проверки правильности значения выходного ресурса и возвращаемых значений об успешности проведения операции. Полученная информация протоколируется, снимается блок с объекта синхронизации, происходит возврат из функции с передачей управления отлаживаемой программе.

На этом работа программы-преобразователя завершается.

3. Устройство программы-анализатора

Обработка протокола производится двумя методами — через чтение файла либо по сети. Протокол устроен таким образом, что для анализа нет принципиальной разницы, откуда поступают данные. Единственное различие заключается том, что в сетевом режиме можно влиять на разрешение выполнения произвольных действий на стороне отлаживаемой программы. Структурно программа анализа разделена на три части.

3.1. Воссоздание информации о ресурсах и функциях

Путём разбора заголовочной части происходит воссоздание необходимых для анализа элементов шаблона с информацией о ресурсах и функциях, с которыми они работают. Поскольку полностью начальный файл-шаблон в формате XML уже не нужен, в простом текстовом виде передаётся только необходимая информация о ресурсах, что сокращает размер заголовочной части и ускоряет начало непосредственной обработки информации. Также отпадает необходимость разбора XML.

3.2. Цикл обработки транзакций

После завершения работы по заполнению легковесной версии шаблона запускается основной цикл, в рамках которого анализируются одна за другой транзакции. Под транзакцией понимается вызов функции с тремя обязательными полями: пролог, проверка, эпилог.

В процессе работы цикла для каждого из ресурсов создаётся база данных (БД) в памяти со всеми необходимыми полями. На каждом шаге проверяется, насколько тот или иной вызов верен. В результате анализа самого вызова и, при необходимости, анализа БД выявляются следующие типы ошибок: «многократное освобождение ресурса», «ситуация, когда функция захвата возвращает уже захваченный ресурс», «использование незахваченного ресурса», «использование ресурса после освобождения», «освобождение неиспользованного ресурса», «использование ошибочного ресурса» (например,

`NULL` для памяти, -1 для дескрипторов, ...), «освобождение ошибочного ресурса», «использование ресурса за его границами» (например, памяти, отображений, ...). Кроме того, происходит анализ «КПД» использования ресурса, т. е. того, насколько ресурс используется и используется ли вообще. В случае низкого КПД это позволяет сократить требование к ресурсам или полностью исключить использование «паразитного» ресурса в определённом месте программы. Для всех вышеперечисленных ошибок система сообщает полную информацию о местах в программе, в которых они были замечены. Например, для ситуации повторного освобождения ресурса будет выдан примерно такой отчёт: "`deallocation of handle 0x12345678 by heap memory:free at bar.c:16 while the handle allocated at baz.c:10 and freed at foo.c:12`". Список обнаруживаемых ошибок в будущем может быть расширен.

3.3. Обработка оставшихся записей в БД после завершения анализируемой программы

На данном этапе все неосвобождённые ресурсы, информация о которых находится в БД, помечаются как утечки, и полная информация предоставляется разработчику для анализа. Формируется окончательная статистика о степени полноты использования ресурсов. Работа по анализу поступивших данных на этом заканчивается.

4. Планы на будущее

Планируется расширить сферу применения системы анализа на ядра других операционных систем (QNX, Solaris, *BSD, Linux и др.). Добиться этого поможет незначительное изменение какого-либо эмулятора с открытым исходным кодом [7], например QEMU [8] или VirtualBox [9]. Единственной необходимой модификацией должна быть поддержка обработки некоторой фиктивной машинной инструкции, с помощью которой будет возможен вывод протокольной информации из ядра в область среды исполнения эмулятора. Этот вывод легко затем перенаправить в файл или гнездо для передачи по сети. Также небезынтересно обнаружение ситуаций, когда вызов какой-либо функции не завершается. Кроме этого, возможно искусственное создание ситуации «нехватки ресурсов» для отлаживаемой программы. В процессе этого могут быть дополнительно выявлены ошибки, вызванные недостаточной проверкой кода завершения операции со стороны отлаживаемой программы. Дополнительно может быть добавлена функциональность, которая позволит собирать и временну́ю информацию, выявляя «узкие места» в работе функций.

ЛИТЕРАТУРА

1. `http://skylark.tsu.ru/art/` — Домашняя страница проекта A Resource Tracer. 2009.
2. *Von Hagen W.* The Definitive Guide to GCC. Second Ed. N.Y.: APRESS, 2006. 584 p.
3. *Powers L., Snell M.* Microsoft Visual Studio 2008 Unleashed. USA: Sams, 2008. 1248 p.
4. `http://libxmlplusplus.sourceforge.net/` — Домашняя страница библиотеки libxml++. 2010.
5. *Elliotte R. H., Means W. S.* XML in a Nutshell. Third Ed. USA: O'Reilly Media, 2004. 600 c.
6. `http://xmlsoft.org/` — Домашняя страница библиотеки libxml2. 2010.
7. `http://www.opensource.org/` — Сайт «Open Source Initiative». 2010.
8. `http://www.qemu.org/` — Домашняя страница проекта QEMU. 2010.
9. `http://www.virtualbox.org/` — Домашняя страница проекта VirtualBox. 2010.

ПРИКЛАДНАЯ ТЕОРИЯ АВТОМАТОВ

DOI 10.17223/20710410/8/11

УДК 519.713+519.766

О ПОСТРОЕНИИ МИНИМАЛЬНЫХ ДЕТЕРМИНИРОВАННЫХ КОНЕЧНЫХ АВТОМАТОВ, РАСПОЗНАЮЩИХ ПРЕФИКСНЫЙ КОД ЗАДАННОЙ МОЩНОСТИ

И. Р. Акишев, М. Э. Дворкин

Санкт-Петербургский государственный университет информационных технологий, механики и оптики, г. Санкт-Петербург, Россия

E-mail: akishev@rain.ifmo.ru, dvorkin@rain.ifmo.ru

Рассматривается задача о построенни минимального по числу состояний детерминированного конечного автомата, который принимает произвольный префиксный код заданной мощности над алфавитом $\{0, 1\}$. Доказывается, что данная задача является эквивалентной задаче о поиске кратчайшей аддитивной цепочки, заканчивающейся числом n.

Ключевые слова: *префиксный код, детерминированный конечный автомат, автомат Мура, аддитивная цепочка.*

Введение

В настоящее время префиксные коды находят широкое применение в различных областях информационных технологий, таких, как обработка и передача информации [1], сжатие данных [2] и многих других. В связи с этим изучение их свойств, а также способов эффективного построения и распознавания префиксных кодов представляет большой интерес.

Один из наиболее простых и естественных подходов к распознаванию префиксных кодов основан на применении конечных автоматов. Имеется ряд статей [3, 4], посвященных исследованию задачи минимизации числа состояний автомата, распознающего некоторый заданный префиксный язык.

В настоящей работе представлен способ генерации минимального (по числу состояний) конечного автомата, распознающего произвольный префиксный код заданной мощности, а также изучается зависимость минимального возможного числа состояний автомата от мощности распознаваемого кода n.

Исследование этой зависимости представляет большую значимость, так как она характеризует нижнюю оценку размера автомата, распознающего любой префиксный код заданной мощности. Практический интерес представляет также изучение структуры самих префиксных кодов, которые соответствуют минимальным автоматам, так как такие коды являются в некотором смысле оптимальными с точки зрения их распознавания посредством конечных автоматов.

В работе будут рассматриваться *детерминированные* конечные автоматы (ДКА), так как они в большей степени приспособлены для применения на практике (в отличие от *недетерминированных*). Также будут рассмотрены минимальные *автоматы Мура* (с выходными символами), которые (в отличие от ДКА) можно применять не

для принятия отдельных слов кода, а для декодирования непрерывной цепочки из последовательных кодовых слов.

1. Основные определения

Перед тем как сформулировать решаемую задачу, введем некоторые определения.

Определение 1. *Алфавитом* Σ называют некоторое непустое конечное множество *символов*.

В настоящей работе рассматривается случай, когда $\Sigma = \{0, 1\}$.

Определение 2. *Словом* называют некоторую конечную (возможно, пустую) последовательность символов алфавита: $w = \sigma_1\sigma_2 \dots \sigma_l \in \Sigma^*$. Количество символов в слове (число l) называют *длиной* слова. Пустое слово принято обозначать ε.

Определение 3. *Языком* L называют множество слов $\{w_i\} \subseteq \Sigma^*$. Язык конечной мощности $C = \{w_1, w_2, \dots, w_n\}$ называют *кодом*, а $n = |C|$ — *мощностью* кода.

Определение 4. Слово $w = \sigma_1\sigma_2 \dots \sigma_l$ является *префиксом* слова $w' = \sigma'_1\sigma'_2 \dots \sigma'_{l'}$, если $l \leqslant l'$ и $\sigma_i = \sigma'_i$ для всех $i \leqslant l$.

Определение 5. Язык (код) C называют *префиксным*, если для всех $w, w' \in C$ слово w не является префиксом w'.

В последнем определении присутствует некоторая путаница, так как подобный язык (код), по мнению авторов, логичнее было бы называть *беспрефиксным*. Однако, в силу исторических причин, принято использовать именно указанное название.

Определение 6. *Детерминированным конечным автоматом* называется пятерка $\langle Q, \Sigma, \delta, s, F\rangle$, где:

— Q — конечное множество состояний;
— Σ — алфавит;
— $\delta : Q \times \Sigma \to Q$ — функция переходов;
— $s \in Q$ — начальное состояние;
— $F \subseteq Q$ — множество терминальных состояний.

Обработка слова $w = \sigma_1\sigma_2 \dots \sigma_l$ ДКА A происходит следующим образом. Сначала автомат A находится в стартовом состоянии s. Затем на каждом шаге обработки автомат считывает очередной символ σ_i слова w и переходит из своего текущего состояния q в состояние $\delta(q, \sigma_i)$. К моменту, когда все символы входного слова w обработаны, автомат находится в некотором состоянии p. Говорят, что автомат A *принимает* слово w, если $p \in F$, и *не принимает* в обратном случае.

Определение 7. Множество слов, которое принимает ДКА A, обозначается $L(A)$ и называется *языком, принимаемым автоматом* A.

Определение 8. Язык L называется *регулярным*, если существует некоторый ДКА A, такой, что $L(A) = L$.

2. Постановка задачи

После того как были введены базовые определения, можно более строго сформулировать исследуемую задачу.

Имеется некоторое натуральное число n. Необходимо построить ДКА, принимающий некоторый префиксный код C, такой, что $|C| = n$, и имеющий при этом наименьшее возможное число состояний ($|Q| \to \min$).

3. Свойства искомого автомата

Известно [5], что для каждого языка, принимаемого некоторым ДКА, существует минимальный (по числу состояний) ДКА, принимающий этот язык. Более того, такой минимальный ДКА — единственный (с точностью до изоморфизма состояний).

Очевидно, что в качестве решения рассматриваемой задачи может выступать только ДКА, являющийся минимальным для языка, который он принимает. В противном случае к данному ДКА можно было бы применить алгоритм минимизации [5], получив таким образом новый ДКА с меньшим числом состояний, принимающий тот же язык, что и исходный.

По этой причине при построении оптимального ДКА представляется возможным использование свойств минимальных автоматов, наиболее важное из которых (в рамках данной задачи) связано с правыми контекстами.

Определение 9. *Правым контекстом* состояния q автомата A (обозначается R_q) называется язык всех слов, которые переводят этот автомат из состояния q в какое-либо терминальное состояние.

В частности, язык $L(A)$ есть не что иное, как R_s, где s — стартовое состояние автомата A.

Вернемся к минимальным автоматам.

Лемма 1. Детерминированный конечный автомат A является минимальным для языка $L(A)$ тогда и только тогда, когда все состояния этого автомата достижимы из начального и имеют различные правые контексты [5].

Объяснение данного свойства заключается в том, что если в ДКА есть два состояния q и q', такие, что $R_q = R_{q'}$, то они *эквивалентны* (также говорят: *неразличимы*), и их можно объединить в одно. Если же некоторое состояние q недостижимо из s, то его можно удалить из автомата, и это никак не повлияет на язык $L(A)$.

Определение 10. *Тупиковым состоянием* ДКА называют состояние d, такое, что $R_d = \varnothing$. (Иногда такое состояние также называют *дьявольским*.)

Лемма 2. Пусть детерминированный конечный автомат A — минимальный автомат, принимающий некоторый непустой язык. Этот язык является префиксным тогда и только тогда, когда в автомате A ровно одно терминальное состояние, причем все переходы из него ведут в тупиковое состояние.

Доказательство. Так как язык $L(A)$ не пуст и содержит хотя бы одно слово u, то этому слову должно соответствовать некоторое терминальное состояние автомата $f \in F$. Рассмотрим правый контекст R_f. В него входит пустое слово ε, поскольку состояние терминальное. Если в него входит любое другое слово w, то, по определению, слово uw также принимается автоматом, а значит, принадлежит языку $L(A)$, как и слово u. Таким образом, язык $L(A)$ содержит два слова, одно из которых является префиксом другого. Следовательно, язык $L(A)$ не префиксный, и имеет место противоречие.

Итак, правый контекст терминального состояния f состоит из единственного элемента — ε. Если в автомате A, помимо f, присутствует еще одно терминальное состояние f', то для него также верно аналогичное рассуждение, а значит, $R_f = R_{f'} = \{\varepsilon\}$. Таким образом, состояния f и f' эквивалентны, а следовательно, автомат A не может быть минимальным, что противоречит условию теоремы. Это означает, что f — единственное терминальное состояние автомата A, а поскольку $R_f = \{\varepsilon\}$, то, находясь в состоянии f и считав любой очередной символ, автомат может перейти лишь в тупиковое состояние.

Докажем теперь обратное утверждение. Пусть минимальный ДКА A имеет ровно одно терминальное состояние f, а все переходы из него ведут в тупиковое состояние. Докажем, что язык $L(A)$ является префиксным.

Предположим, что это не так — существуют некоторые неравные слова u и uw, принадлежащие $L(A)$. Поскольку автомат принимает слово u, а терминальное состояние в автомате только одно, то по слову u из начального состояния автомат обязан переходить именно в состояние f.

Теперь рассмотрим поведение автомата в случае, когда он считывает слово uw, находясь в начальном состоянии. Обработав префикс u, автомат, как было показано выше, переходит в состояние f. Затем, считав первый символ непустого слова w, автомат перейдет в тупиковое состояние и навсегда останется там, считывая все оставшиеся символы слова w. Так как тупиковое состояние не является терминальным, то автомат A не примет слово uw. Получаем противоречие. ■

Итак, минимальный автомат, принимающий некоторый префиксный язык, имеет ровно одно терминальное состояние f. Более того, это состояние достижимо из любого другого состояния, кроме тупикового. Действительно, если существует состояние q, из которого не достижимо f, то $R_q = \varnothing$, а следовательно, q — тупиковое состояние.

Лемма 3. Пусть детерминированный конечный автомат A — минимальный автомат, принимающий конечный префиксный язык. Тогда в этом автомате нет циклов, кроме петель, ведущих из тупикового состояния в само себя.

Доказательство. Предположим, что в автомате A имеется цикл, проходящий через состояние q, отличное от тупикового. Рассмотрим символы, соответствующие переходам, образующим этот цикл. Эти символы образуют некоторое непустое слово v, которое переводит автомат из состояния q в него же.

Поскольку A минимальный, состояние q достижимо из начального по некоторому слову u (лемма 1). Кроме того, так как $R_q \neq \varnothing$, то найдется некоторое слово w, переводящее автомат из q в терминальное состояние.

В таком случае языку $L(A)$ принадлежат следующие слова: uw, uvw, $uvvw$ и т. д. Получаем $uv^*w \subseteq L(A)$. Следовательно, язык $L(A)$ бесконечен, и имеет место противоречие. ■

Из леммы 3 следует, что искомый автомат A, после удаления из него тупикового состояния, будет представлять собой ациклический граф. Следовательно, его состояния можно отсортировать в порядке обратной топологической сортировки [6], то есть присвоить им такие последовательные номера, чтобы все переходы вели из состояния с бóльшим номером в состояние с меньшим номером.

Пусть в ДКА A всего k состояний. Выпишем их все, кроме тупикового, в порядке присвоенных им номеров (использование нумерации с нуля будет оправданно далее):

$$q_0, q_1, q_2, \ldots, q_{k-2}.$$

Поскольку из любого состояния (кроме тупикового) достижимо терминальное, то оно будет находиться на первом месте в нашей последовательности ($q_0 = f$). Аналогично, поскольку любое состояние достижимо из начального, то начальное состояние будет иметь наибольший номер ($q_{k-2} = s$).

Рассмотрим теперь соответствующую последовательность мощностей правых контекстов для выписанных состояний:

$$a_0, a_1, a_2, \ldots, a_{k-2},$$

где $a_i = |R_{q_i}|$. Из равенств $R_{q_0} = R_f = \{\varepsilon\}$ следует, что $a_0 = 1$.

Рассмотрим некоторое нетерминальное состояние q_i. Из него выходят два перехода, каждый из которых может вести либо в некоторое состояние с меньшим номером q_j, либо в тупиковое состояние d.

Если оба перехода из q_i ведут в d, то R_{q_i} пуст, а следовательно q_i — тоже тупиковое состояние, чего не может быть, так как рассматриваемый автомат A является минимальным.

Если один из переходов ведет в тупиковое состояние, а другой — в некоторое состояние q_j $(j < i)$, то мощность правого контекста q_i равна мощности правого контекста q_j (так как любому слову $w \in R_{q_j}$ взаимнооднозначно соответствует слово $\sigma w \in R_{q_i}$, где σ — символ, которому соответствует переход из q_i в q_j).

Если оба перехода ведут в отличные от тупикового состояния q_j и q_k $(j, k < i$, возможно, $j = k)$, то мощность правого контекста q_i равна сумме мощностей правых контекстов состояний q_j и q_k.

Итак, рассматриваемая последовательность a_i обладает следующими свойствами:

1) нулевой элемент последовательности равен единице;
2) каждый элемент, кроме нулевого, либо совпадает с некоторым предыдущим, либо является суммой двух (возможно, одинаковых) предыдущих элементов.

Определение 11. Будем называть последовательность чисел, обладающую данными свойствами, *квазиаддитивной цепочкой*. *Длиной* квазиаддитивной цепочки $a_0 = 1, a_1, a_2, \ldots, a_r$ будем называть число r.

Итак, длина полученной квазиаддитивной цепочки на два меньше числа состояний в искомом автомате (так ее элементы соответствуют всем состояниям, кроме тупикового). При этом мощность языка $L(A)$, принимаемого автоматом, равна последнему элементу цепочки, соответствующему начальному состоянию.

Таким образом, мы доказали следующую лемму.

Лемма 4. Любому минимальному детерминированному конечному автомату A с k состояниями, принимающему префиксный код заданной мощности n, соответствует некоторая квазиаддитивная цепочка длины $k-2$, заканчивающаяся заданным числом n.

Докажем теперь обратное утверждение.

Лемма 5. Любой квазиаддитивной цепочке длины k, оканчивающейся заданным числом n, соответствует некоторый детерминированный конечный автомат A с $k+2$ состояниями, принимающий префиксный язык мощности n.

Доказательство. Проведем обратное построение, аналогичное рассмотренному выше. В качестве множества состояний Q автомата A возьмем множество $\{d, q_0, q_1, \ldots, q_k\}$ мощности $k+2$. Состояние $q_k = s$ сделаем стартовым, $q_0 = t$ — единственным терминальным, а d — тупиковым. Функцию переходов δ для состояний $q_1, \ldots, q_k$ определим следующим образом:

— если в квазиаддитивной последовательности a_i получалось повторением a_j $(j < i)$, то переходы из вершины q_i будут вести в вершины q_j и d;
— если же a_i получалось как сумма a_j и a_k $(j, k < i)$, то переходы из вершины q_i будут вести в q_j и q_k.

Соответствие символов и переходов из заданной вершины можно выбрать произвольным образом.

Все переходы из состояний t и d направим в состояние d.

Так как из каждого из состояний $q_1, \ldots, q_k$ в полученном автомате A как минимум один из переходов ведет в нетупиковое состояние с меньшим номером, то очевидно, что по таким переходам из стартового состояния достижимо терминальное. Следовательно, язык $L(A)$ содержит хотя бы одно слово и не является пустым.

Построенный автомат A, возможно, является неминимальным. Тогда применим к нему процедуру минимизации [5], получив новый автомат A'. Так как в процессе минимизации автомата возможно лишь удаление некоторых состояний, но не появление новых, то автомат A' будет по-прежнему содержать единственное терминальное состояние t', все переходы из которого будут вести в тупиковое состояние d'. Следовательно, по лемме 2, язык $L(A') = L(A)$ — префиксный.

Нетрудно также убедиться (с помощью индукции, аналогичной примененной при доказательстве леммы 4), что в автомате A выполнено равенство $|R_{q_i}| = a_i$, а значит, $|L(A)| = |R_s| = |R_{q_k}| = a_k = n$. ■

Из лемм 4 и 5 следует эквивалентность задачи нахождения минимальной квазиаддитивной цепочки с заданным последним числом и минимального конечного автомата, принимающего префиксный код заданной мощности.

Заметим один важный факт: кратчайшая квазиаддитивная цепочка, заканчивающаяся заданным числом, не может содержать одно и то же число несколько раз. В противном случае мы можем удалить из нее все вхождения повторяющегося числа, кроме самого первого, уменьшив тем самым ее длину. (При этом, если удаляемые вхождения использовались для получения последующих элементов цепочки, мы сможем использовать вместо них первое вхождение, не нарушив тем самым структуру оставшейся цепочки.)

Следовательно, в кратчайшей квазиаддитивной цепочке всегда имеет смысл получать очередной элемент лишь как сумму двух предыдущих.

Определение 12. Квазиаддитивная цепочка, в которой каждый элемент равен сумме двух предыдущих, называется *аддитивной цепочкой*.

Итак, кратчайшая квазиаддитивная цепочка, заканчивающаяся заданным числом, всегда является аддитивной.

Из этого наблюдения следует еще одно свойство искомого минимального ДКА.

Лемма 6. В детерминированном конечном автомате, принимающем некоторый префиксный код заданной мощности n и имеющем минимальное число состояний, нет переходов в тупиковое состояние, кроме как из единственного терминального и из тупикового состояний.

Доказательство. Предположим, что в автомате A имеется переход в тупиковое состояние из вершины q_i, не являющейся ни терминальной, ни тупиковой. Как было установлено ранее в процессе доказательства леммы 4, из q_i может вести лишь один такой переход. Значит, в соответствующей квазиаддитивной цепочке число a_i получено путем повторения предыдущего элемента (соответствующего состоянию, в которое ведет второй переход из q_i). Но наличие повторений в цепочке означает, что она неминимальна (повторное число можно удалить), а следовательно, автомат A имеет неминимальное число состояний. ■

Из того факта, что любая кратчайшая квазиаддитивная цепочка является аддитивной, а также из лемм 4 и 5 вытекает следующая важная теорема.

Теорема 1. Задача нахождения детерминированного конечного автомата A с минимальным числом состояний, принимающего некоторый префиксный код заданной

мощности n, эквивалентна задаче построения кратчайшей аддитивной цепочки, заканчивающейся заданным числом n.

В качестве примера на рис. 1 приведен минимальный ДКА, принимающий префиксный код мощности 15. Он соответствует кратчайшей аддитивной цепочке, заканчивающейся числом 15: $(1, 2, 3, 6, 12, 15)$.

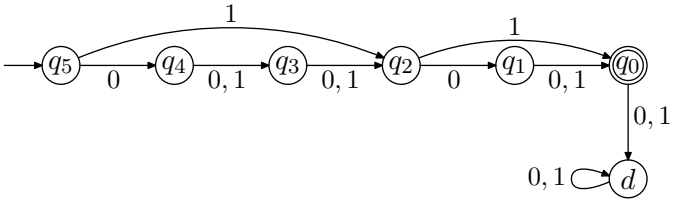

Рис. 1. Минимальный ДКА, принимающий префиксный код мощности 15

Подытожим изученные свойства искомого минимального ДКА. В автомате присутствует единственное терминальное состояние t, переходы из которого ведут в единственное тупиковое состояние d. Если удалить из автомата состояние d, то получится ациклический ориентированный граф, из каждой вершины которого (кроме t) выходит ровно два ребра.

4. Задача о нахождении кратчайшей аддитивной цепочки

Задача о нахождении кратчайшей аддитивной цепочки является классической задачей дискретной математики [7]. Наиболее известным ее применением является задача об оптимальном алгоритме возведения произвольного числа в заданную степень, что немаловажно для криптографии [8].

Действительно, в процессе вычисления для данного числа x числа x^n будут получены различные степени числа x, причем каждая следующая степень x^k получается как произведение уже вычисленных степеней x^i и x^j, то есть $x^k = x^i \cdot x^j = x^{i+j}$. Поэтому показатели вычисляемых степеней составляют аддитивную цепочку, заканчивающуюся числом n. И минимизация длины такой аддитивной цепочки соответствует минимизации числа умножений при возведении в n-ю степень.

Классический бинарный метод возведения в степень [7] использует следующие соотношения:

$$\begin{cases} x^{2k} = x^k \cdot x^k; \\ x^{2k+1} = x^{2k} \cdot x^1. \end{cases}$$

Например, для возведения в 15-ю степень данный метод предлагает вычислить последовательность $x^1, x^2, x^3, x^6, x^7, x^{14}, x^{15}$, что соответствует аддитивной цепочке $(1, 2, 3, 6, 7, 14, 15)$.

Этот метод позволяет возводить в n-ю степень с помощью $O(\log n)$ операций умножения. Поскольку всякое число в аддитивной цепочке не превосходит максимального из предыдущих чисел, умноженного на два, аддитивная цепочка растет не быстрее, чем последовательность степеней двойки. Поэтому получить число n за $o(\log n)$ операций умножения невозможно, следовательно, бинарный метод возведения в степень является асимптотически оптимальным.

В то же время получаемая аддитивная цепочка не всегда является кратчайшей из возможных. Так, при $n = 15$ существует более короткая аддитивная цепочка $(1, 2, 3, 6, 12, 15)$. (Здесь используется тот факт, что $15 = 3 \cdot 5$ — число 15 получается из тройки, которая, в свою очередь, получается из единицы.)

Если искать кратчайшую (не асимптотически, а абсолютно) аддитивную цепочку, заканчивающуюся числом n, сложность задачи возрастает коренным образом. Полиномиального решения этой задачи на данный момент неизвестно. Активно применяются методы поиска приближенного ответа, в том числе ведутся исследования по применению генетических алгоритмов [9] и «муравьиных алгоритмов» [10].

Длина кратчайшей аддитивной цепочки, заканчивающейся числом n, обозначается $l(n)$. Из анализа метода бинарного возведения в степень вытекает неравенство $l(n) \leqslant \lfloor \log n \rfloor + \nu(n) - 1$, где $\nu(n)$ — это число единиц в двоичной записи числа n. Более точные асимптотические оценки для $l(n)$ были доказаны в работе [11]:

$$\log(n) + \log(\nu(n)) - 2{,}13 \leqslant l(n) \leqslant \log(n) + \frac{\log(n)(1 + o(1))}{\log(\log(n))}.$$

Достаточно полный обзор имеющихся результатов о кратчайших цепочках можно найти на портале [12].

В работе [13] показано, что задача нахождения кратчайшей аддитивной цепочки, которая содержит в качестве подпоследовательности данную последовательность $b_1, b_2, \ldots, b_k$, является NP-полной [5]. То есть естественное обобщение рассматриваемой задачи не имеет полиномиального решения, если $P \neq NP$.

5. Автомат Мура, обрабатывающий поток слов

Рассмотрим теперь следующую задачу: пусть имеется некоторый поток символов, который является последовательностью слов заданного префиксного кода, следующих подряд друг за другом. Необходимо по мере чтения этого потока разбивать его на отдельные кодовые слова для дальнейшей обработки.

Классический детерминированный конечный автомат, описанный выше, не подходит для данной задачи, так как он умеет лишь принимать или не принимать некоторые конечные слова языка.

В данной же ситуации требуется такой автомат, который смог бы получать бесконечный входной поток и уведомлять о том, когда прочитано очередное кодовое слово и начинается новое.

Для этой цели удобно использовать автомат Мура, который является частным случаем автомата с выходными символами.

Определение 13. Автоматом Мура называют шестерку $\langle Q, \Sigma, \delta, s, \Lambda, \mu \rangle$, где

— Q — конечное множество состояний;
— Σ — входной алфавит;
— Λ — выходной алфавит;
— $\delta : Q \times \Sigma \to Q$ — функция переходов;
— s — начальное состояние;
— $\mu : Q \to \Lambda$ — функция выходов.

Как следует из этого определения, у автомата Мура, в отличие от обычного детерминированного конечного автомата, отсутствуют терминальные состояния. Вместо них появляются выходные символы, которые задаются в каждом состоянии функцией μ. Автомат Мура обрабатывает бесконечный входной поток и выдает последовательность соответствующих выходных символов.

Применительно к исследуемой в данной работе задаче выходной алфавит автомата Мура Λ будет состоять из трех символов $\ominus$, $\oplus$ и $\oslash$.

Символ $\ominus$ соответствует начатому, но неоконченному процессу чтения кодового слова.

Символ $\oplus$ соответствует случаю, когда из входного потока только что было считано очередное кодовое слово или же еще не было обработано ни одного входного символа, и следующим шагом начнется чтение нового кодового слова. В частности, $\mu(s) = \oplus$.

Символ $\oslash$ соответствует случаю, когда в процессе обработки очередного входного символа была получена последовательность, которая не может являться началом ни одного слова из языка. Это означает, что входной поток содержит ошибку и не может быть распознан и разбит на слова. Как только автомат Мура сгенерировал выходной символ $\oslash$, дальнейшее распознавание входного потока становится невозможно, и вне зависимости от последующих входных символов автомат будет всегда продолжать выдавать выходной символ $\oslash$. Будем по аналогии с ДКА называть состояние d, для которого $\mu(d) = \oslash$, *тупиковым*. Все переходы из тупикового состояния будут вести в него же. Очевидно, что в автомате Мура достаточно одного такого состояния.

Пример автомата Мура, распознающего префиксный код $\{00, 10, 11\}$, показан на рис. 2. Для входной последовательности 10100011011 этот автомат сгенерирует следующую последовательность выходных символов: $\oplus \ominus \oplus \ominus \oplus \ominus \oplus \ominus \oplus \ominus \oslash\oslash$.

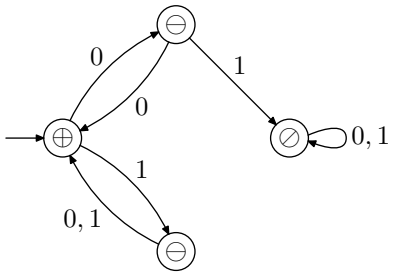

Рис. 2. Автомат Мура для кода $\{00, 10, 11\}$

Как и в случае с ДКА, поставленная задача — построить минимальный автомат Мура, принимающий некоторый префиксный код заданной мощности n.

Структуры минимального автомата Мура, распознающего поток слов некоторого префиксного кода (в описанном выше смысле), и минимального детерминированного конечного автомата, принимающего слова этого кода, взаимосвязаны — существует простой способ построить второй автомат на основе первого и наоборот.

Перед тем как описать такое построение, отметим следующее свойство минимального автомата Мура, распознающего префиксный код.

Лемма 7. В любом минимальном автомате Мура, распознающем некоторый префиксный код, существует ровно одно состояние q, такое, что $\mu(q) = \oplus$, и это состояние — s.

Доказательство. Покажем, что такое состояние единственно. Пусть существует два различных состояния q и q', таких, что $\mu(q) = \mu(q') = \oplus$. Это означает, что находящийся в этих состояниях автомат только что принял очередное кодовое слово и готов начать считывать следующее. Однако начиная с каждого следующего кодового слова корректность цепочки входных символов никак не зависит от набора ранее принятых слов. Иными словами, для произвольной бесконечной цепочки символов $w = \sigma_1\sigma_2\ldots$, подаваемой на вход автомату в состоянии q, всегда получится та же самая цепочка выходных символов, как и в случае, если бы на вход автомату подавалась цепочка w, когда тот находился в состоянии q'. Таким образом, эти состояния неразличимы автоматом, и, следовательно, можно удалить одно из них, например q', перенаправив при этом все переходы, которые вели в него, в состояние q.

Однако так как $\mu(s) = \oplus$, состояние s — это и есть искомое состояние. ■

Теорема 2. Пусть k — минимальное возможное число состояний для ДКА, принимающего префиксный код заданной мощности n. Тогда минимальное возможное число состояний автомата Мура, распознающего префиксный код мощности n, равно $k-2$, причем любому ДКА с k состояниями, принимающему некоторый префиксный код мощности n, взаимнооднозначно соответствует автомат Мура с $k-2$ состояниями, распознающий тот же самый код.

Доказательство. Рассмотрим минимальный ДКА, принимающий префиксный код мощности n и содержащий k состояний. Исключим из него тупиковое состояние d и единственное терминальное состояние t. Все переходы, которые раньше вели в t, направим в стартовое состояние s. Так как по лемме 6 в исходном ДКА в состояние d могли вести лишь переходы из удаленных состояний t и d, то все остальные переходы останутся корректно определены.

Назначим $\mu(s) = \oplus$ и $\mu(q) = \ominus$ для всех $q \neq s$. Построенный таким образом автомат Мура будет содержать $k-2$ состояния. Нетрудно убедиться (по построению), что он будет распознавать тот же префиксный код, что и исходный ДКА.

Заметим, что в построенном таким образом автомате Мура будет отсутствовать тупиковое состояние.

Проведем теперь обратное построение. Рассмотрим некоторый минимальный автомат Мура, состоящий из k состояний (не считая тупикового, если оно в нем имеется) и распознающий слова префиксного кода заданной мощности n. По лемме 7 в нем есть всего одно состояние s, такое, что $\mu(s) = \oplus$, и оно является стартовым. Введем новое состояние t, а также тупиковое состояние d, если оно отсутствовало в исходном автомате Мура. Все переходы, которые вели в состояние s, направим теперь в t, а все переходы из состояний t и d — в d.

Сделаем t единственным терминальным состоянием. Полученный ДКА будет иметь $k+2$ состояния и будет принимать тот же префиксный код, что и исходный.

Осталось показать, что при описанных выше построениях минимальному автомату будет ставиться в соответствие минимальный. Пусть имелся минимальный ДКА A, принимающий префиксный код длины n и содержащий k состояний. Применив описанное построение, получим автомат Мура B, состоящий из $k-2$ состояний. Предположим, что B не является минимальным. Тогда имеется некоторый автомат Мура B', содержащий не более $k-3$ состояний и распознающий некоторый префиксный код длины n. Но из него можно построить ДКА A', который будет содержать не более $k-1$ состояния. Следовательно, исходный автомат A не минимальный. Получаем противоречие. ■

Определение 14. Префиксный код C называется *полным*, если для любого слова $v \in \Sigma^*$ существует слово $w \in C$, такое, что одно из них является префиксом другого.

Лемма 8. Префиксный код C заданной мощности, соответствующий автомату с минимальным числом состояний, является полным.

Доказательство. При доказательстве теоремы 2 было замечено, что у искомого минимального автомата Мура отсутствует тупиковое состояние. Если подать на вход автомату произвольное слово $v \in \Sigma^*$, получится некоторая последовательность выходных символов, состоящая лишь из символов $\oplus$ и $\ominus$, причем начинается она с символа $\oplus$. Если символ $\oplus$ встречается в ней не только в начале, то соответствующий префикс v является кодовым словом. В противном случае существует такое слово w,

которое переводит автомат в состояние s из того состояния, в котором он оказался после прочтения v. А значит, слово vw — кодовое. ■

На рис. 3 изображен минимальный автомат Мура, распознающий префиксный код длины 15, соответствующий ДКА, изображенному на рис. 1.

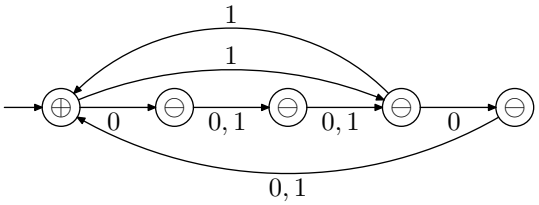

Рис. 3. Минимальный автомат Мура, принимающий префиксный код мощности 15

6. Счетный префиксный язык

Выше была рассмотрена задача о построении минимального автомата, принимающего префиксный язык заданной *конечной* мощности.

Однако префиксные языки бывают как конечные, так и бесконечные. Если префиксный язык над конечным алфавитом бесконечен, то он представляет собой счетное множество. Для полноты исследования найдем минимальный по числу состояний ДКА, принимающий некоторый счетный префиксный язык.

Лемма 9. В детерминированном конечном автомате, принимающем счетный префиксный язык, не менее трех состояний.

Доказательство. Поскольку язык счетный и, следовательно, непустой, в ДКА должно быть хотя бы одно терминальное состояние, достижимое из начального. Назовем (любое) такое состояние t. Пусть оно достижимо из начального по слову u.

Рассмотрим R_t. В него входит пустое слово ε, поскольку $t \in F$. Если в него входит еще хотя бы одно слово w, то языку $L(A)$ принадлежат и слово u, и слово uw — язык $L(A)$ не префиксный, имеем противоречие. Следовательно, правый контекст состояния t — это язык $\{\varepsilon\}$.

Следовательно, оба перехода из t должны вести в состояние с правым контекстом $\varnothing$, то есть в тупиковое состояние d, которое не совпадает с t.

Наконец, поскольку язык $L(A)$ счетный, то правый контекст начального состояния не может равняться ни $\varnothing$, ни $\{\varepsilon\}$. Следовательно, начальное состояние не совпадает ни с t, ни с d, и доказано наличие в автомате A не менее трех состояний. ■

На рис. 4 приведен пример ДКА с тремя состояниями, принимающего счетный префиксный язык. Он принимает язык 0^*1 — язык слов, состоящих из произвольного числа нулей и единицы в конце. Этот язык префиксный и имеет счетную мощность.

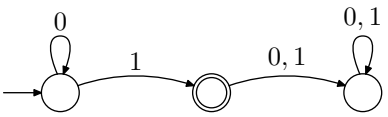

Рис. 4. ДКА, принимающий язык 0^*1

Автомат Мура для этого языка имеет два состояния. Он приведен на рис. 5. Автомат Мура с одним состоянием не может принимать счетный язык: он будет либо

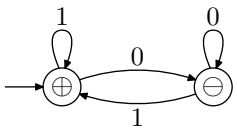

Рис. 5. Автомат Мура, принимающий язык 0^*1

выдавать выходной символ $\oplus$ после каждого считанного символа, либо не будет выдавать его вовсе. В первом случае мощность языка равна двум, во втором — нулю.

Наконец, рассмотрим вырожденный случай — префиксный язык мощности ноль. Этот случай является особенным, поскольку в автомате, принимающем пустой язык, не требуется терминальное состояние. ДКА, принимающий (префиксный) язык мощности ноль, приведен на рис. 6. Автомат Мура для данного случая совпадает с ДКА; выходной символ в единственном его состоянии следует положить равным $\oslash$.

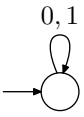

Рис. 6. ДКА, принимающий пустой язык

Заключение

В данной работе рассмотрена задача о построении минимального автомата, принимающего произвольный префиксный код заданной мощности; доказана её эквивалентность задаче генерации кратчайшей аддитивной цепочки, заканчивающейся заданным числом.

Исследована также аналогичная задача для автоматов Мура.

ЛИТЕРАТУРА

1. Unicode specification. `http://unicode.org/`.
2. *Elias P.* Universal codeword sets and representations of the integers // IEEE Trans. Inform. Theory. 1975. V. 21. P. 194–203.
3. *Golin M. J., Na H.* Optimal prefix-free codes that end in a specified pattern and similar problems: The uniform probability case (extended abstract) // Data Compression Conference. 2001. P. 143–152.
4. *Han Y.-S., Salomaa K., Wood D.* State complexity of prefix-free regular languages // Proc. of the 8th Int. Workshop on Descriptional Complexity of Formal Systems. 2006. P. 165–176.
5. *Хопкрофт Д. Э., Мотвани Р., Ульман Д. Д.* Введение в теорию автоматов, языков и вычислений. М.: Вильямс, 2002. 528 с.
6. *Кормен Т., Лейзерсон Ч., Ривест Р.* Алгоритмы: построение и анализ. М.: МЦНМО, БИНОМ, 2004. 960 с.
7. *Кнут Д. Э.* Искусство программирования. Т. 2: Получисленные алгоритмы. М.: Вильямс, 2004. 832 с.
8. *Bleichenbacher D.* Efficiency and security of cryptosystems based on number theory. Zürich, 1996.
9. *Cruz-Cortes N., Rodriguez-Henriquez F., Juarez-Morales R., Coello C. A.* Finding optimal addition chains using a genetic algorithm approach. LNCS. 2005. V. 3801. P. 208–215.
10. *Nedjah N., de Macedo M. L.* Finding minimal addition chains using ant colony // IDEAL / ed. by R. Y. Zheng, R. M. Everson, Y. Hujun. LNCS. 2004. V. 3177. P. 642–647.

11. *Schonhage A.* A lower bound for the length of addition chains // Theoretical Computer Science. 1975. V. 1. No. 1. P. 1–12.
12. *Flammenkamp A.* Shortest addition chains. `http://wwwhomes.uni-bielefeld.de/achim/addition_chain.html`.
13. *Downey P., Leong B., Sethi R.* Computing sequences with addition chains // SIAM J. Computing. 1981. V. 10. No. 3. P. 638–646.

2010 Прикладная теория автоматов №2(8)

DOI 10.17223/20710410/8/12

УДК 519.713

РЕШЕНИЕ ПАРАЛЛЕЛЬНЫХ УРАВНЕНИЙ ДЛЯ ω-ЯЗЫКОВ[1]

В. Г. Бушков, Н. В. Евтушенко

Томский государственный университет, г. Томск, Россия

E-mail: v.bushkov@gmail.com, ninayevtushenko@yahoo.com

Рассмотрена проблема решения параллельного уравнения для ω-языков. В частности, доказано, что, подобно уравнениям для регулярных языков, разрешимое уравнение всегда имеет наибольшее решение, формула которого имеет такой же вид, как и формула наибольшего решения для формальных языков. Показано также, что решение уравнения для ω-регулярных языков сводится к последовательности операций над полуавтоматами.

Ключевые слова: *автоматные уравнения, параллельная композиция ω-языков, уравнения для ω-языков, условие Бюхи.*

Введение

Ряд задач анализа и синтеза дискретных систем сводится к решению уравнений в алгебре регулярных языков. Как известно, любое разрешимое уравнение имеет наибольшее решение [1], которое может рассматриваться как резервуар, содержащий все возможные решения, интересные с теоретической или практической точек зрения. Для описания поведения реактивных систем используются так называемые ω-регулярные языки [2, 3], которые состоят из слов бесконечной длины и описываются полуавтоматами, использующими условие распознавания Бюхи (Büchi) [4]. Можно показать, что методы решения уравнений для регулярных языков не могут быть напрямую применены к уравнениям для ω-регулярных языков [5], и соответственно представляет интерес задача решения уравнений для таких языков. В данной работе введена операция параллельной (асинхронной) композиции для ω-языков и доказано, что, подобно уравнениям для регулярных языков, разрешимое параллельное уравнение всегда имеет наибольшее решение. Формула для наибольшего решения уравнения над ω-языками имеет такой же вид, как и формула наибольшего решения для формальных языков. Также установлено, что уравнение для ω-регулярных языков можно решать как уравнение для соответствующих полуавтоматов.

1. Основные понятия

Алфавитом Σ называется непустое конечное множество символов, которые будем называть также буквами. *Словом* u в алфавите Σ называется конечная последовательность символов из Σ; количество букв в слове u называется длиной слова и обозначается $|u|$. Пустое слово, т. е. слово, не содержащее ни одного символа, будем обозначать ε. По определению считается, что длина слова ε равна 0. Как обычно, множество всех конечных слов над алфавитом Σ обозначается Σ^*.

Бесконечным словом или *ω-словом* над алфавитом Σ называется бесконечная последовательность символов из алфавита Σ. Бесконечное слово α может быть представлено как отображение $\alpha : \mathbb{N} \to \Sigma$, где $\mathbb{N} = \{1, 2, \ldots\}$ есть множество натуральных

[1]Работа частично поддержана программой «У.М.Н.И.К.» Фонда содействия развитию МП НТС (госконтракт № 7450р/10262) и проектом ФЦП (госконтракт № 02.514.12.4002).

чисел. Пусть α есть бесконечное или конечное слово, тогда $\alpha(i)$, где $i \in \mathbb{N}$, обозначает символ на i-й позиции в слове α. Множество всех бесконечных слов над алфавитом Σ будем обозначать Σ^ω.

Символы алфавитов далее обозначаются первыми буквами латинского алфавита, а именно $a, b, \ldots$; слова конечной длины обозначаются последними буквами латинского алфавита, а именно $u, v, \ldots$; бесконечные слова обозначаются греческими буквами, а именно $\alpha, \beta, \ldots$ Подмножество $U \subseteq \Sigma^*$ называется *формальным языком*, или просто *языком*; соответственно подмножество $L \subseteq \Sigma^\omega$ называется *ω-языком*, или *сверхъязыком* [6]. Для любых $u \in \Sigma^*$ и $\alpha \in (\Sigma^* \cup \Sigma^\omega)$ *конкатенация* $u{\cdot}\alpha$ (или просто $u\alpha$) есть такое слово из $(\Sigma^* \cup \Sigma^\omega)$, что $(u\alpha)(i) = u(i)$ для всех $1 \leqslant i \leqslant |u|$ и $(u\alpha)(i) = \alpha(i - |u|)$ для всех $i > |u|$. *Конкатенация языков* $U \subseteq \Sigma^*$ и $L \subseteq (\Sigma^* \cup \Sigma^\omega)$ определяется следующим образом: $UL = \{u\alpha \in (\Sigma^* \cup \Sigma^\omega) : u \in U \wedge \alpha \in L\}$. Пусть $U \subseteq \Sigma^*$; тогда *ω-замыканием* множества U называется множество слов $U^\omega = \{\alpha \in \Sigma^\omega : \alpha = u_1 u_2 \ldots u_i \ldots, u_i \in U \backslash \{\varepsilon\}, i \in \mathbb{N}\}$ из Σ^ω, образованных конкатенацией бесконечного числа слов из $U \backslash \{\varepsilon\}$. По определению, если $U = \{\varepsilon\}$, то $U^\omega = \varnothing$.

Подобно регулярным языкам, ω-регулярные языки можно описать с помощью полуавтоматов, использующих правило распознавания Бюхи.

Полуавтоматом называется пятерка $\mathcal{S} = (S, A, s_0, T_\mathcal{S}, F_\mathcal{S})$, где A — алфавит, S — конечное непустое множество состояний, s_0 — начальное состояние, $T_\mathcal{S} \subseteq S \times A \times S$ — отношение переходов, $F_S \subseteq S$ — множество финальных состояний. Отношение переходов можно распространить на слова конечной длины: в полуавтомате $\mathcal{S}$ есть переход из состояния s в состояние s' под действием слова $u \in A^*$, если и только если существует последовательность состояний $s = s_1, s_2, \ldots, s_{n+1} = s'$, такая, что $(s_i, u(i), s_{i+1}) \in T_S$ для любого $i \in \{1, \ldots, n\}$.

В данной работе полуавтомат рассматривается как распознаватель бесконечных слов по правилу Бюхи [4]. Для слов конечной длины в полуавтомате справедливы все утверждения из классической теории автоматов.

Пусть $\mathcal{S} = (S, A, s_0, T_\mathcal{S}, F_\mathcal{S})$ есть полуавтомат и $\alpha \in A^\omega$ есть бесконечное слово. *Обходом полуавтомата* $\mathcal{S}$ под действием слова α (или просто *α-обходом*) называется бесконечное слово ρ_α в множестве состояний полуавтомата $\mathcal{S}$, такое, что $\rho_\alpha(1) = s_0$ и для любого $i \in \mathbb{N}$ имеет место $(\rho_\alpha(i), \alpha(i), \rho_\alpha(i+1)) \in T_\mathcal{S}$. Согласно определению, полуавтомат $\mathcal{S}$ распознает бесконечное слово α, если существует состояние из множества $F_\mathcal{S}$, которое встречается бесконечное число раз в α-обходе полуавтомата $\mathcal{S}$. Язык $L(\mathcal{S})$, распознаваемый полуавтоматом $\mathcal{S}$, или язык полуавтомата $\mathcal{S}$, есть множество всех бесконечных слов, распознаваемых полуавтоматом $\mathcal{S}$. Далее язык $L(\mathcal{S})$ называется просто языком полуавтомата $\mathcal{S}$. Полуавтоматы $\mathcal{S}$ и $\mathcal{B}$ называются *эквивалентными* ($\mathcal{S} \equiv \mathcal{B}$), если их языки совпадают. Полуавтомат $\mathcal{S}$ называется *редукцией* полуавтомата $\mathcal{B}$ ($\mathcal{S} \leqslant \mathcal{B}$), если язык полуавтомата $\mathcal{S}$ содержится в языке полуавтомата $\mathcal{B}$.

Язык $L \subseteq \Sigma^\omega$ называется *ω-регулярным* [7], если его можно представить в виде объединения конечного числа языков UV^ω, где U и V суть регулярные языки над алфавитом Σ. Иными словами, язык L есть ω-регулярный язык, если существует такое натуральное число n, что $L = \bigcup\limits_{i \in \{1,\ldots,n\}} U_i V_i^\omega$, где U_i и V_i суть регулярные языки над алфавитом Σ, $i = 1, \ldots, n$.

Теорема 1 [7]. Язык распознаваем некоторым полуавтоматом, если и только если язык является ω-регулярным.

В работе [7] при доказательстве теоремы 1 показано, что язык любого полуавтомата $\mathcal{S} = (S, A, s_0, T_\mathcal{S}, F_\mathcal{S})$ можно представить в виде $\bigcup\limits_{s \in F_\mathcal{S}} V_{s_0 s} V_{ss}^{\omega}$, где $V_{ss'} = \{u \in A^* : (s, u, s') \in T_\mathcal{S}\}$ есть множество всех слов конечной длины, переводящих полуавтомат $\mathcal{S}$ из состояния $s \in S$ в состояние $s' \in S$.

Пусть $\mathcal{S} = (S, A, s_0, T_\mathcal{S}, F_\mathcal{S})$ есть полуавтомат с ε-переходами [8] и $\alpha \in A^\omega$ есть бесконечное слово. Полуавтомат $\mathcal{S}$ с ε-переходами распознает бесконечное слово α, если существует некоторое состояние $s \in F_\mathcal{S}$, которое встречается бесконечное число раз при α-обходе полуавтомата $\mathcal{S}$. Язык $L(\mathcal{S})$, распознаваемый полуавтоматом $\mathcal{S}$ с ε-переходами, есть множество всех бесконечных слов, распознаваемых полуавтоматом $\mathcal{S}$.

Введем понятие ε-замыкания ε-closure$(\mathcal{S})$ полуавтомата $\mathcal{S}$ с ε-переходами. Определим отношение ε-closure$(T_\mathcal{S})$, положив $(s, a, s') \in \varepsilon$-closure$(T_\mathcal{S})$ для s, $s' \in S$ и $a \in A$ тогда и только тогда, когда в полуавтомате $\mathcal{S}$ из состояния s в состояние s' можно перейти по последовательности $\varepsilon \ldots \varepsilon a \varepsilon \ldots \varepsilon$. Полуавтомат $(S, A, s_0, \varepsilon\text{-closure}(T_\mathcal{S}), F_\mathcal{S})$ называется ε-замыканием полуавтомата $\mathcal{S}$.

Утверждение 1. Полуавтоматы ε-closure$(\mathcal{S})$ и $\mathcal{S}$ эквивалентны.

2. Операции над ω-языками

2.1. Операции распространения и ограничения

Для ω-языков операция распространения вводится так же, как для формальных языков. Пусть A и B — непересекающиеся алфавиты и ω-язык L определен над алфавитом A. Рассмотрим отображение $\psi : A^\omega \to 2^{(A \cup B)^\omega}$, где для $\alpha = a_1 a_2 \ldots \in A^\omega$ имеет место $\psi(\alpha) = \{u_1 a_1 u_2 a_2 \ldots : u_i \in B^*, i \in \mathbb{N}\}$. Тогда язык $L_{\Uparrow B} = \bigcup\limits_{\alpha \in L} \psi(\alpha)$ есть *распространение* ω-языка L на алфавит B, или *B-распространение* ω-языка L, т. е. при B-распространении L перед любым символом любого сверхслова в L можно вставить любое конечное слово в алфавите B.

Пусть L — ω-язык, заданный над алфавитом $(A \cup B)$. Введем понятие *ограничения* $L_{\Downarrow A}$ на алфавит A, называемого также *A-ограничением* ω-языка L. Если $L \subseteq (A \cup B)^* B^\omega$, то $L_{\Downarrow A} = \varnothing$. Если $L \nsubseteq (A \cup B)^* B^\omega$, то $L_{\Downarrow A} = \{h(a_1) h(a_2) \ldots : a_1 a_2 \ldots \in \in L \backslash (A \cup B)^* B^\omega\}$, где отображение $h : (A \cup B) \to A$ определяется как $h(a) = a$ для всех $a \in A$ и $h(a) = \varepsilon$ для всех $a \notin A$. Здесь условие $a_1 a_2 \ldots \in L \backslash (A \cup B)^* B^\omega$ гарантирует получение ω-языка в качестве результата операции.

Утверждение 2. Пусть L есть ω-язык над алфавитом $(A \cup B)$. Тогда

$$L_{\Downarrow A} = \varnothing \Leftrightarrow L \subseteq (A \cup B)^* B^\omega.$$

Доказательство. Достаточность очевидна. Необходимость. Предположим, что $L \nsubseteq (A \cup B)^* B^\omega$. Тогда существует $\alpha \in L$, которое можно представить в виде $\alpha = = u_1 a_1 u_2 a_2 \ldots u_i a_i \ldots$, где $u_i \in B^*, a_i \in A, i \in \mathbb{N}$. Из определения операции ограничения следует, что $h(u_1) h(a_1) h(u_2) h(a_2) \ldots h(u_i) h(a_i) \ldots = a_1 a_2 \ldots a_i \ldots \in L_{\Downarrow A}$ и $L_{\Downarrow A} \neq \varnothing$. ■

Операция ограничения обратна операции распространения, поэтому верно

Утверждение 3. Пусть A и B — непересекающиеся алфавиты и $L \subseteq A^\omega$. Тогда $(L_{\Uparrow B})_{\Downarrow A} = L$.

Утверждение 4. Пусть $L_1 \subseteq (A \cup B)^\omega$ и $L_2 \subseteq (A \cup B)^\omega$ суть ω-языки, причем $(L_{2 \Downarrow A})_{\Uparrow B} = L_2$. Тогда имеет место $(L_1 \cap L_2)_{\Downarrow A} = L_{1 \Downarrow A} \cap L_{2 \Downarrow A}$.

Доказательство. Необходимость. Пусть $\alpha \in (L_1 \cap L_2)_{\Downarrow A}$, т. е. существует такое $\gamma \in (L_1 \cap L_2)$, что $h(\gamma) = \alpha$. Тогда $\alpha = h(\gamma) \in L_{1\Downarrow A}$ и $\alpha = h(\gamma) \in L_{2\Downarrow A}$. Следовательно, $\alpha \in L_{1\Downarrow A} \cap L_{2\Downarrow A}$.

Достаточность. Пусть $\alpha = a_1 a_2 \dots a_i \dots \in L_{1\Downarrow A} \cap L_{2\Downarrow A}$, т. е. $\alpha \in L_{1\Downarrow A}$ и $\alpha \in L_{2\Downarrow A}$. Поскольку $\alpha \in L_{1\Downarrow A}$, существует слово $\gamma = u_1 a_1 u_2 a_2 \dots u_i a_i \dots \in L_1$, в котором $u_i \in B^*$, $i \in \mathbb{N}$, такое, что $h(\gamma) = \alpha$. По определению распространения множество $\psi(\alpha)$ содержит слово γ. Из $\alpha \in L_{2\Downarrow A}$ следует $\psi(\alpha) \subseteq (L_{2\Downarrow A})_{\Uparrow B} = L_2$, поэтому $\gamma \in L_2$. Поскольку $\gamma \in L_1$ и $\gamma \in L_2$, верно $\gamma \in (L_1 \cap L_2)$. Следовательно, $\alpha = h(\gamma) \in (L_1 \cap L_2)_{\Downarrow A}$. ■

2.2. Параллельная композиция

Рассмотрим попарно непересекающиеся множества I, V, U и O и ω-языки $L_1 \subseteq (I \cup V \cup U)^\omega$ и $L_2 \subseteq (O \cup V \cup U)^\omega$. Положим $E = (I \cup V \cup O)$; тогда *параллельной композицией* ω-языков L_1 и L_2 называется ω- язык $L_1 \diamond_E L_2 = (L_{1\Uparrow O} \cap L_{2\Uparrow I})_{\Downarrow E}$.

Параллельная композиция подразумевает, что компоненты взаимодействуют посредством общих действий $(U \cup V)$, т. е. действие из $(U \cup V)$ выполняется тогда и только тогда, когда обе компоненты готовы выполнить это действие, в то время как действия из $(I \cup O)$ выполняются независимо от другой компоненты. Параллельная композиция отвечает асинхронному взаимодействию систем, т. е. взаимодействию, допускающему произвольные задержки по времени.

Из определения параллельной композиции непосредственно следует

Утверждение 5. Если $L_2' \subseteq L_2$, то $L_1 \diamond_E L_2' \subseteq L_1 \diamond_E L_2$.

3. Уравнения для ω-языков

Пусть I, U, V, O — непересекающиеся алфавиты, $C \subseteq (I \cup V \cup U)^\omega$ и $S \subseteq (I \cup O \cup V)^\omega$ — ω-языки, $E = I \cup O \cup V$ и $H = U \cup O \cup V$. Выражение вида $C \diamond_E X = S$, в котором неизвестная компонента X есть ω-язык в алфавите H, называется *уравнением.*

Как обычно, ω-язык $B \subseteq H^\omega$ называется *решением* уравнения $C \diamond_E X = S$, если справедливо $C \diamond_E B = S$. Это решение *наибольшее*, если в нём содержится любое решение данного уравнения. Уравнение $C \diamond_E X = S$ называется разрешимым, если для него существует решение.

Теорема 2. Разрешимое уравнение $C \diamond_E X = S$ для ω-языков имеет наибольшее решение, и это решение есть ω-язык $\overline{C \diamond_H \overline{S}}$, где $\overline{S}$ есть дополнение языка S.

Доказательство. Рассмотрим произвольное слово $\alpha \in H^\omega$ и покажем, что $C \diamond_E \{\alpha\} \subseteq S \Leftrightarrow \alpha \in \overline{C \diamond_H \overline{S}}$. Имеем

$$
\begin{gathered}
C \diamond_E \{\alpha\} \subseteq S \Leftrightarrow \\
(C_{\Uparrow O} \cap \{\alpha\}_{\Uparrow I})_{\Downarrow E} \cap \overline{S} = \varnothing \Leftrightarrow \text{/ввиду } \overline{S} = (\overline{S}_{\Uparrow U})_{\Downarrow E} \text{ по утверждению 3/} \\
(C_{\Uparrow O} \cap \{\alpha\}_{\Uparrow I})_{\Downarrow E} \cap (\overline{S}_{\Uparrow U})_{\Downarrow E} = \varnothing \Leftrightarrow \text{/по утверждению 4 ввиду } ((\overline{S}_{\Uparrow U})_{\Downarrow E})_{\Uparrow U} = \overline{S}_{\Uparrow U}\text{/} \\
(C_{\Uparrow O} \cap \{\alpha\}_{\Uparrow I} \cap \overline{S}_{\Uparrow U})_{\Downarrow E} = \varnothing.
\end{gathered}
$$

Согласно утверждению 2, последнее равенство выполняется тогда и только тогда, когда $C_{\Uparrow O} \cap \{\alpha\}_{\Uparrow I} \cap \overline{S}_{\Uparrow U} \subseteq (O \cup V \cup U \cup I)^* U^\omega$. Если допустить, что $\varnothing \neq C_{\Uparrow O} \cap \{\alpha\}_{\Uparrow I} \cap \overline{S}_{\Uparrow U} \subseteq (O \cup V \cup U \cup I)^* U^\omega$, то в $\overline{S}_{\Uparrow U}$ найдётся слово из $(O \cup V \cup U \cup I)^* U^\omega$, что возможно лишь тогда, когда в $\overline{S}$ есть слово из $(O \cup V \cup U \cup I)^* U^\omega$, а это не так. Следовательно, $(C_{\Uparrow O} \cap \{\alpha\}_{\Uparrow I} \cap \overline{S}_{\Uparrow U})_{\Downarrow E} = \varnothing \Leftrightarrow C_{\Uparrow O} \cap \{\alpha\}_{\Uparrow I} \cap \overline{S}_{\Uparrow U} = \varnothing$.

Если для $H = U \cup V \cup O$ допустить, что $(C_{\Uparrow O} \cap \{\alpha\}_{\Uparrow I} \cap \overline{S}_{\Uparrow U})_{\Downarrow H} = \varnothing$ и $C_{\Uparrow O} \cap \cap \{\alpha\}_{\Uparrow I} \cap \overline{S}_{\Uparrow U} \neq \varnothing$, то ввиду утверждения 2 $\varnothing \neq C_{\Uparrow O} \cap \{\alpha\}_{\Uparrow I} \cap \overline{S}_{\Uparrow U} \subseteq (O \cup V \cup U \cup I)^* I^\omega$

и в $\{\alpha\}_{\Uparrow I}$ найдётся слово из $(O \cup V \cup U \cup I)^* I^\omega$, что невозможно. Таким образом, $C_{\Uparrow O} \cap \{\alpha\}_{\Uparrow I} \cap \overline{S}_{\Uparrow U} = \varnothing \Leftrightarrow (C_{\Uparrow O} \cap \{\alpha\}_{\Uparrow I} \cap \overline{S}_{\Uparrow U})_{\Downarrow H} = \varnothing$. Кроме того,

$$(C_{\Uparrow O} \cap \{\alpha\}_{\Uparrow I} \cap \overline{S}_{\Uparrow U})_{\Downarrow H} = \varnothing \Leftrightarrow \text{/по утверждению 4 ввиду } ((\{\alpha\}_{\Uparrow I})_{\Downarrow H})_{\Uparrow I} = \{\alpha\}_{\Uparrow I}/$$
$$(\{\alpha\}_{\Uparrow I})_{\Downarrow H} \cap (C_{\Uparrow O} \cap \overline{S}_{\Uparrow U})_{\Downarrow H} = \varnothing \Leftrightarrow \text{/ввиду } \{\alpha\} = (\{\alpha\}_{\Uparrow I})_{\Downarrow H} \text{ по утверждению 3/}$$
$$\{\alpha\} \cap (C_{\Uparrow O} \cap \overline{S}_{\Uparrow U})_{\Downarrow H} = \varnothing \Leftrightarrow \alpha \in \overline{(C_{\Uparrow O} \cap \overline{S}_{\Uparrow U})_{\Downarrow H}} \Leftrightarrow \alpha \in \overline{C \diamond \overline{S}}.$$

Таким образом, $C \diamond_E \{\alpha\} \subseteq S \Leftrightarrow \alpha \in \overline{C \diamond_H \overline{S}}$.

Рассмотрим решение B разрешимого уравнения $C \diamond_E X = S$ и слово $\alpha \in B$. Поскольку $C \diamond_E B = S$, то $C \diamond_E \{\alpha\} \subseteq S$ (утверждение 5) и, следовательно, $\alpha \in \overline{C \diamond_H \overline{S}}$. Таким образом, $B \subseteq \overline{C \diamond_H \overline{S}}$, т. е. любое решение уравнения $C \diamond_E X = S$ содержится в $\overline{C \diamond_H \overline{S}}$. Поскольку $B \subseteq \overline{C \diamond_H \overline{S}}$ и $C \diamond_E B = S$, то $S = C \diamond_E B \subseteq C \diamond_E \overline{C \diamond_H \overline{S}}$ (утверждение 5). С другой стороны, для любого слова $\alpha \in \overline{C \diamond_H \overline{S}}$ имеет место $C \diamond_E \{\alpha\} \subseteq S$, т. е. $C \diamond_E \overline{C \diamond_H \overline{S}} \subseteq S$. Следовательно, $S \subseteq C \diamond_E \overline{C \diamond_H \overline{S}} \subseteq S$ и $\overline{C \diamond_H \overline{S}}$ есть решение и, значит, наибольшее решение уравнения $C \diamond_E X = S$. ■

Как и в случае формальных языков, любое решение уравнения $C \diamond_E X = S$ содержится в языке $\overline{C \diamond_H \overline{S}}$, однако не каждый язык, содержащийся в $\overline{C \diamond_H \overline{S}}$, является решением уравнения. Далее покажем, каким образом решение уравнения для ω-регулярных языков можно свести к соответствующим операциям над полуавтоматами.

4. Операции над полуавтоматами, распознающими ω-регулярные языки

4.1. Операция ограничения

Пусть $\mathcal{S} = (S, A \cup B, s_0, T_\mathcal{S}, F_\mathcal{S})$ — полуавтомат. *Ограничением* полуавтомата $\mathcal{S}$ на алфавит A называется полуавтомат $\mathcal{S}_{\Downarrow A}$, который может быть получен из $\mathcal{S}$ заменой для каждого состояния $s \in S$ и каждого символа $b \in B \backslash A$ всех переходов вида (s, b, s') на переходы (s, ε, s') и взятием ε-замыкания полученного полуавтомата с ε-переходами, которое эквивалентно полученному полуавтомату с ε-переходами (согласно утверждению 1).

Утверждение 6. Ограничение полуавтомата $\mathcal{S} = (S, A \cup B, s_0, T_\mathcal{S}, F_\mathcal{S})$ на алфавит A распознает язык $\bigcup\limits_{s \in F_\mathcal{S}} V_{s_0 s \Downarrow A} (V_{ss \Downarrow A})^\omega$, который является A-ограничением языка $L(\mathcal{S})$.

Доказательство. Согласно [7], $L(\mathcal{S}) = \bigcup\limits_{s \in F_\mathcal{S}} V_{s_0 s} V_{ss}^\omega$, поэтому ввиду утверждения 1 $L(\mathcal{S}_{\Downarrow A}) = \varepsilon\text{-closure } L(\mathcal{S})_{\Downarrow A} = L(\mathcal{S})_{\Downarrow A} = \bigcup\limits_{s \in F_\mathcal{S}} V_{s_0 s \Downarrow A} (V_{ss \Downarrow A})^\omega$. ■

4.2. Операция распространения

Пусть $\mathcal{S} = (S, A, s_0, T_\mathcal{S}, F_\mathcal{S})$ — полуавтомат и A и B — непересекающиеся алфавиты. Тогда распространением полуавтомата $\mathcal{S}$ на алфавит B будет полуавтомат $\mathcal{S}_{\Uparrow B}$, который может быть получен из $\mathcal{S}$ посредством следующей процедуры. Для каждого состояния $s \in S \backslash F_\mathcal{S}$ и каждого символа $b \in B$ добавляется петля (s, b, s); для каждого состояния $s \in F_\mathcal{S}$ добавляются нефинальное состояние s', переход (s, b, s') и петля (s', b, s') для каждого символа $b \in B$, а также для всех $s'' \in S$ и $a \in A$, таких, что $(s, a, s'') \in T_\mathcal{S}$ для некоторого $s \in S$, добавляется по переходу (s', a, s'').

На рис. 1 показано распространение на алфавит $B = \{b\}$ полуавтомата $\mathcal{S}$, заданного над алфавитом $A = \{a, c\}$, представленного на рис. 2.

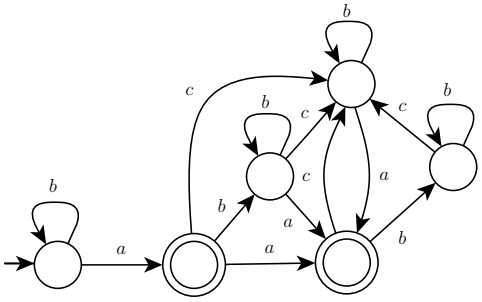

Рис. 1. Полуавтомат $S_{\Uparrow B}$

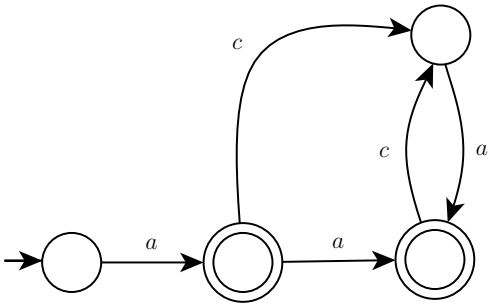

Рис. 2. Полуавтомат S

Добавление петель в финальные состояния привело бы к появлению в языке $L(S_{\Uparrow B})$ бесконечных слов из $(A \cup B)^* B^\omega$, что означает возможность функционирования полуавтомата $S_{\Uparrow B}$ в композиции с другим без участия второго, что недопустимо в реактивных системах.

Утверждение 7. Распространение полуавтомата $S = (S, A, s_0, T_S, F_S)$ на алфавит B распознает язык $\bigcup\limits_{s \in F_S} V_{s_0 s \Uparrow B}((V_{ss} \backslash \{\varepsilon\})_{\Uparrow B})^\omega$, который является B-распространением языка $L(S)$.

Доказательство. Согласно [7], $L(S) = \bigcup\limits_{s \in F_S} V_{s_0 s} V_{ss}^\omega$, поэтому $L(S_{\Uparrow B}) = L(S)_{\Uparrow B} = \bigcup\limits_{s \in F_S} V_{s_0 s \Uparrow B}(V_{ss \Uparrow B})^\omega$. ■

4.3. Операция параллельной композиции

Ω-Пересечение $S_1 \cap S_2$ полуавтоматов $S_1 = (S_1, A, s_{0,1}, T_{S_1}, F_{S_1})$ и $S_2 = (S_2, A, s_{0,2}, T_{S_2}, F_{S_2})$ есть полуавтомат $S = (S, A, s_0, T_S, F_S)$ [7], в котором $S = S_1 \times S_2 \times \{1, 2\}$, $s_0 = (s_{0,1}, s_{0,2}, 1)$, $F_S = S_1 \times F_{S_2} \times \{2\}$, и отношение переходов определяется по следующим правилам:

$$((s_1, s_2, 1), a, (s_1', s_2', 1)) \in T_S, \text{если} (s_1, a, s_1') \in T_{S_1}, (s_2, a, s_2') \in T_{S_2}, \text{и } s_1 \notin F_{S_1};$$
$$((s_1, s_2, 1), a, (s_1', s_2', 2)) \in T_S, \text{если} (s_1, a, s_1') \in T_{S_1}, (s_2, a, s_2') \in T_{S_2}, \text{и } s_1 \in F_{S_1};$$
$$((s_1, s_2, 2), a, (s_1', s_2', 2)) \in T_S, \text{если} (s_1, a, s_1') \in T_{S_1}, (s_2, a, s_2') \in T_{S_2}, \text{и } s_2 \notin F_{S_2};$$
$$((s_1, s_2, 2), a, (s_1', s_2', 1)) \in T_S, \text{если} (s_1, a, s_1') \in T_{S_1}, (s_2, a, s_2') \in T_{S_2}, \text{и } s_2 \in F_{S_2}.$$

Согласно [7], пересечение $S_1 \cap S_2$ полуавтоматов S_1 и S_2 распознает пересечение ω-языков полуавтоматов S_1 и S_2.

Параллельной композицией полуавтоматов $S_1 = (S_1, I \cup V \cup U, s_{0,1}, T_{S_1}, F_{S_1})$ и $S_2 = (S_2, O \cup V \cup U, s_{0,2}, T_{S_2}, F_{S_2})$ называется полуавтомат $S_1 \diamond_E S_2 \equiv (S_{1 \Uparrow O} \cap S_{2 \Uparrow I})_{\Downarrow E}$, где $E = (I \cup V \cup O)$.

Утверждение 8. Параллельная композиция полуавтоматов распознает параллельную композицию их ω-регулярных языков.

Доказательство. Согласно утверждению 7, полуавтоматы $S_{1 \Uparrow O}$ и $S_{2 \Uparrow I}$ распознают языки $L(S_1)_{\Uparrow O}$ и $L(S_2)_{\Uparrow I}$ соответственно. Их Ω-пересечение $S_{1 \Uparrow O} \cap S_{2 \Uparrow I}$ распознает язык $L(S_1)_{\Uparrow O} \cap L(S_2)_{\Uparrow I}$ [7]. По утверждению 6 ограничение $(S_{1 \Uparrow O} \cap S_{2 \Uparrow I})_{\Downarrow E}$ распознает язык $(L(S_1)_{\Uparrow O} \cap L(S_2)_{\Uparrow I})_{\Downarrow E}$, который является параллельной композицией соответствующих ω-регулярных языков. ■

Далее через $\overline{\mathsf{S}}$ обозначается полуавтомат, который распознает дополнение ω-языка полуавтомата S. Известно, что в случае ω-регулярных языков полуавтомат $\overline{\mathsf{S}}$ строится значительно сложнее, чем в случае формальных языков [7].

Пусть I, U, V, O — непересекающиеся алфавиты, $C \subseteq (I \cup V \cup U)^\omega$ и $S \subseteq (I \cup O \cup V)^\omega$ — ω-языки, $E = I \cup O \cup V$, $H = U \cup O \cup V$, C и S суть полуавтоматы, распознающие ω-языки.

Теорема 3. Наибольшее решение разрешимого уравнения $C \diamond_E X = S$ для ω-регулярных языков есть ω-регулярный язык, распознаваемый полуавтоматом $\overline{\mathsf{C} \diamond_H \overline{\mathsf{S}}}$.

Доказательство. Полуавтомат $\overline{\mathsf{S}}$ распознает ω-регулярный язык $\overline{L(\mathsf{S})} = \overline{S}$ [7]. По утверждению 8 полуавтомат $\mathsf{C} \diamond_H \overline{\mathsf{S}}$ распознает параллельную композицию $C \diamond_H \overline{S}$. Полуавтомат $\overline{\mathsf{C} \diamond_H \overline{\mathsf{S}}}$ распознает ω-регулярный язык $\overline{C \diamond_H \overline{S}}$, который является наибольшим решением уравнения $C \diamond_E X = S$ для ω-регулярных языков. ■

Заметим, что так же, как для регулярных языков, на основании теоремы 3 могут быть разработаны алгоритмы решения соответствующего уравнения для ω-регулярных языков; однако необходимо иметь в виду, что операция построения полуавтомата, распознающего дополнение ω-регулярного языка, реализуется достаточно сложным алгоритмом.

Заключение

В данной работе введена операция параллельной композиции ω-языков и рассмотрена проблема решения параллельных уравнений. Доказано, что разрешимое уравнение всегда имеет наибольшее решение и вид формулы наибольшего решения для ω-языков совпадает с видом такой формулы при решении уравнений для формальных языков. Показано также, что в случае ω-регулярных языков такое решение может быть выражено аналогичной формулой, где вместо языков стоят полуавтоматы, их распознающие.

ЛИТЕРАТУРА

1. *Yevtushenko N. V., Villa T., Brayton R. K., et al.* Solution of Parallel Language Equations for Logic Synthesis // Proceedings of ICCAD 2001, San Jose. 2001. P. 103–110.
2. *Wang G., Mishchenko A., Brayton R. K., et al.* Sequential synthesis with co-Buchi specifications // ERL Technical Report, EECS Dept., UC Berkeley, April 2006. 6 p.
3. *Thistle J. G., Wonham W. M.* Supervision of infinte behavior of discrete-event systems // SIAM, Control and Optimization. 1994. V. 32. No. 4. P. 1098–1113.
4. *Buchi J. R.* On a decision method in restricted second order arithmetic // Z. Math. Logik Grundlag. Math. 1960. P. 66–92.
5. *Бушков В. Г., Евтушенко Н. В.* Решение параллельных уравнений для ω-регулярных языков // Прикладная дискретная математика. Приложение. 2009. № 1. С. 6–7.
6. *Трахтенброт Б. А., Барздинь Я. М.* Конечные автоматы (поведение и синтез). М.: Наука, 1970. 400 с.
7. *Mukund M.* Finite-state automata on infinite inputs // Tutorial talk, NSTCS 6, Banasthali, Rajasthan, India, August 1996. 32 p.
8. *Хопкрофт Д., Мотвани Р., Ульман Д.* Введение в теорию автоматов, языков и вычислений. М.: Вильямс, 2008. 528 с.

СВЕДЕНИЯ ОБ АВТОРАХ

АКИШЕВ Искандер Рустемович — бакалавр, магистрант кафедры компьютерных технологий факультета информационных технологий и программирования Санкт-Петербургского государственного университета информационных технологий, механики и оптики, г. Санкт-Петербург. E-mail: **akishev@rain.ifmo.ru**

АЛЕКСЕЙЧУК Антон Николаевич — доктор технических наук, профессор кафедры Института специальной связи и защиты информации Национального технического университета Украины «Киевский политехнический институт», г. Киев.
E-mail: **alex-crypto@mail.ru**

БОРИСЕНКО Борис Борисович — сотрудник Института проблем информационной безопасности МГУ, г. Москва. E-mail: **fepem@yandex.ru**

БУШКОВ Виктор Георгиевич — магистрант Томского государственного университета, г. Томск. E-mail: **v.bushkov@gmail.com**

БЫКОВА Валентина Владимировна — доцент, кандидат технических наук, профессор Института математики Сибирского федерального университета, г. Красноярск.
E-mail: **bykvalen@mail.ru**

ГОРЕЛОВ Владимир Владимирович — аспирант кафедры защиты информации и криптографии Томского государственного университета, г. Томск. E-mail: **skylark@mail.tsu.ru**

ДВОРКИН Михаил Эдуардович — бакалавр, магистрант кафедры компьютерных технологий факультета информационных технологий и программирования Санкт-Петербургского государственного университета информационных технологий, механики и оптики, г. Санкт-Петербург. E-mail: **dvorkin@rain.ifmo.ru**

ДИМИТРИЕВ Юрий Константинович — доцент, доктор технических наук, ведущий научный сотрудник Института физики полупроводников СО РАН, г. Новосибирск.
E-mail: **dimi@isp.nsc.ru**

ЕВТУШЕНКО Нина Владимировна — профессор, доктор технических наук, заведующая кафедрой информационных технологий в исследовании дискретных структур Томского государственного университета, г. Томск.
E-mail: **ninayevtushenko@yahoo.com**

ЗАДОРОЖНЫЙ Анатолий Филиппович — доцент, кандидат технических наук, заведующий кафедрой Новосибирского архитектурно-строительного университета, г. Новосибирск. E-mail: **3af@ngs.ru**

МЕЛЕНТЬЕВ Виктор Александрович — кандидат технических наук, старший научный сотрудник Института физики полупроводников СО РАН, г. Новосибирск.
E-mail: **melva@isp.nsc.ru**

ПАРВАТОВ Николай Георгиевич — кандидат физико-математических наук, доцент Томского государственного университета, г. Томск.
E-mail: **parvatov@mail.tsu.ru**

ПРОСКУРОВСКИЙ Роман Васильевич — кандидат технических наук, доцент кафедры Института специальной связи и защиты информации Национального технического университета Украины «Киевский политехнический институт», г. Киев.
E-mail: **roman-crypto@mail.ru**

СУХИНИН Борис Михайлович — аспирант кафедры информационной безопасности Московского государственного технического университета им. Н. Э. Баумана, г. Москва. E-mail: **b.sukhinin@gmail.com**

ФОМИЧЕВ Владимир Михайлович — старший научный сотрудник, доцент, доктор физико-математических наук, ведущий научный сотрудник Учреждения Российской академии наук «Институт проблем информатики РАН», г. Москва. E-mail: **fomichev@nm.ru**

ЧЕРЕМУШКИН Александр Васильевич — доктор физико-математических наук, профессор, член-корреспондент Академии криптографии, заведующий кафедрой Института криптографии, связи и информатики, г. Москва. E-mail: **avc238@mail.ru**

АННОТАЦИИ СТАТЕЙ НА АНГЛИЙСКОМ ЯЗЫКЕ

Alekseychuk A. N., Proskurovskiy R. V. **NECESSARY AND SUFFICIENT CONDITIONS OF LINEAR STRUCTURE TRIVIALITY FOR MONOMIAL MAPPING ON THE FIELD OF 2^{2^t} ELEMENTS.** The necessary and sufficient conditions are obtained in terms of binary structure of natural number d, under which all non-zero linear combinations of coordinate functions of mapping $x \mapsto x^d$, $x \in \mathbf{GF}(2^{2^t})$, don't have linear translators.
Keywords: *linear structure of discrete mappings, finite field, monomial mapping.*

Parvatov N. G. **GALOIS RELATION FOR CLOSED CLASSES OF DISCRETE FUNCTIONS.** Galois theory for S-closed classes of discrete functions is formulated. In particular cases, these classes coincide with the superposition-closed classes of functions of multivalued logics, with clones, with the hereditary systems of discrete functions and with the classes of functions computed by switching circuits.
Keywords: *closed class, clone, Galois connection, discrete function.*

Fomichev V. M. **THE PROPERTIES OF h-PERIODIC SEQUENCES.** The notion of a sequence h-periodicity is introduced with a function h mapping the set of words composing the sequence into a set. The properties of h-periodic sequences are investigated. In the case of additive h a connection between the period length and the h-period length of a sequence is established, and the h-period length of linear recurring sequences and of de Bruijn sequences are determined. It is stated that cryptoproperties of some gamma generators depend on h-period length of control sequence where h is the function marking the symbols of the sequence.
Keywords: *sequence period, additive function, linear substitution.*

Cheremushkin A. V. **AN ADDITIVE APPROACH TO NONLINEAR DEGREE OF DISCRETE FUNCTION.** An additive approach to the definition of the nonlinearity degree of a discrete function is proposed. For elementary abelian groups, this notion is equivalent to ordinary "multiplicative" one. As a consequence we obtain the method for describing a stabilizer groups of a p^m-valued function in the transition groups.
Keywords: *discrete functions, nonlinearity degree, stabilizer group.*

Sukhinin B. M. **HIGH-SPEED PSEUDORANDOM SEQUENCE GENERATORS BASED ON CELLULAR AUTOMATA.** We investigate a number of properties of uniform two-dimensional boolean cellular automata and propose a new method for pseudorandom sequences generation based on such automata. Generated sequences show good statistical properties. Moreover, hardware implementation of the method on a typical FPGA has very high performance of up to 25 Gbps at 100 MHz frequency.
Keywords: *PRNG, cellular automata, avalance effect.*

Borisenko B. B. **MODIFIED HOTELLING'S CHART EXCLUDING TREND INFLUENCE AND ITS APPLICATION FOR DIGITAL WATERMARKS DETECTION.** A modified Hotelling's T^2-statistics is suggested for application in control charts in order to exclude trend influence. Also, a method of using control charts for digital

watermarks detection is discussed.
Keywords: *computer security, digital watermarks, control charts method.*

Dimitriev Yu. K., Zadorozhny A. F. **APPLICATION OF UNRELIABLE TESTS FOR SELF-DIAGNOSTICS OF MODULAR COMPUTING SYSTEMS AT MULTIPLE FAULTS.** Self-diagnosing of modular computing systems (CS) is investigated on the system level at the presence of multiple faults and with the use of unreliable tests. Dependence of self-diagnosing efficiency on properties of the unreliable tests is studied by the method of imitation-statistical simulation. We compare the self-diagnosing efficiency for CS using the unreliable tests corresponding to the known PMC-model and the unreliable tests suggested by the authors, and state conditions under which the use of the last tests is more effective.
Keywords: *self-diagnosing, modular computing systems, multiple faults, unreliable tests.*

Melent'ev V. A. **AN ANALYTICAL APPROACH TO THE SYNTHESIS OF REGULAR GRAPHS WITH PRESET VALUES OF THE ORDER, DEGREE AND GIRTH.** The approach is based on the representation of the graph by its projections and is similar to the solving of an equation system composed of the graph projections.
Keywords: *graph order, graph diameter, graph girth, graph projection, synthesis of the regular graph.*

Bykova V. V. **ELASTICITY OF ALGORITHMS.** We present the characterization of elasticity for rapid, polynomial, subexponential, exponential and hyperexponential algorithms. We give a method for comparing algorithms by their elasticity.
Keywords: *computation complexity, algorithms analysis.*

Gorelov V. V. **THE DISCOVERY OF FAULTY WORK WITH RESOURCES IN THE SOFTWARE.** Crossplatform method to trace software errors in runtime during resource manipulation is shown. A scheme for describing abstract resources and their functions in XML is presented. Due to separation it's possible to describe vast classes of resources without changing the analyzer core. Detected errors often have an influence on security, safety, robustness and resource optimal usage.
Keywords: *software, instrumenting, resources, debugging, tracing.*

Akishev I. R., Dvorkin M. E. **ON CONSTRUCTING MINIMAL DETERMINISTIC FINITE AUTOMATON RECOGNIZING A PREFIX-CODE OF A GIVEN CARDINALITY.** The article considers constructing minimal deterministic finite automaton recognizing a prefix-code of a given cardinality over the alphabet $\{0, 1\}$. The considered problem is proved to be equivalent to the problem of finding the shortest addition-chain ending with a given number. Several interesting properties of the desired minimal finite automaton are proved, and the identical problem concerning Moore automata is discussed.
Keywords: *prefix code, finite-state machine, Moore automaton, addition chain.*

Bushkov V. G., Yevtushenko N. V. **SOLVING PARALLEL EQUATIONS OVER ω-LANGUAGES.** The paper is devoted to the problem of solving equations over ω-languages. Similar to formal languages, a solvable equation over ω-languages has the largest solution and the formula of the largest solution is similar to that for formal languages.
Keywords: *automata equations, parallel composition of ω-languages, equations over ω-languages, Büchi automata.*

Журнал «Прикладная дискретная математика» распространяется по подписке; его подписной индекс 38696 в объединённом каталоге «Пресса России». Полнотекстовые электронные версии вышедших номеров журнала доступны на его сайте vestnik.tsu.ru/pdm и на Общероссийском математическом портале www.mathnet.ru. На сайте журнала можно найти также и правила подготовки рукописей статей в журнал.

Тематика публикаций журнала:

- *Теоретические основы прикладной дискретной математики*
- *Математические методы криптографии*
- *Математические методы стеганографии*
- *Математические основы компьютерной безопасности*
- *Математические основы надежности вычислительных и управляющих систем*
- *Прикладная теория кодирования*
- *Прикладная теория автоматов*
- *Прикладная теория графов*
- *Логическое проектирование дискретных автоматов*
- *Математические основы информатики и программирования*
- *Вычислительные методы в дискретной математике*
- *Дискретные модели реальных процессов*
- *Математические основы интеллектуальных систем*
- *Исторические очерки по дискретной математике и ее приложениям*