МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

ПРИКЛАДНАЯ ДИСКРЕТНАЯ МАТЕМАТИКА

Научный журнал

2011 №1(11)

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

ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

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

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

Периодичность выхода журнала: 4 номера в год.

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

Подписано к печати 10.03.2011.
Формат $60 \times 84\frac{1}{8}$. Усл. п. л. 13,8. Уч.-изд. л. 15,47. Тираж 300 экз.

Отпечатано в типографии ТПУ.

СОДЕРЖАНИЕ

CONTENTS

ТЕОРЕТИЧЕСКИЕ ОСНОВЫ ПРИКЛАДНОЙ ДИСКРЕТНОЙ МАТЕМАТИКИ

DOI 10.17223/20710410/11/1

УДК 519.1.

ПРОИЗВОДЯЩИЕ ФУНКЦИИ ПОСЛЕДОВАТЕЛЬНОСТИ ЧИСЕЛ ПОКРЫТИЙ КОНЕЧНОГО МНОЖЕСТВА

Р. М. Ганопольский

Тюменский государственный университет, г. Тюмень, Россия

E-mail: rodion@utmn.ru

Рассматриваются производящие функции последовательности комбинаторных чисел, исчисляющих количество покрытий конечного множества подмножествами с заданными мощностями. Проведен анализ производящих функций, приведены частные случаи, получен ряд рекуррентных соотношений.

Ключевые слова: *покрытие, конечное множество, комбинаторные числа, производящие функции.*

Введение

В работе [1] введены комбинаторные числа неупорядоченных покрытий конечного множества мощности n подмножествами с фиксированными мощностями

$$ {}_nN(k_1, k_2, \ldots, k_n), \tag{1} $$

где k_i — количество подмножеств мощности i в покрытии. В случае, когда часть коэффициентов $k_i = 0$, предложено использовать другое обозначение — ${}_nN^{k_1 k_2 \cdots k_m}_{l_1\, l_2 \cdots l_m}$, где k_i — количество подмножеств мощности l_i в покрытии. Для введенных комбинаторных чисел получены формула

$$ {}_nN^{k_1 k_2 \cdots k_m}_{l_1\, l_2 \cdots l_m} = \prod_{i=1}^{m} C^{k_i}_{C_n^{l_i}} + \sum_{i \geqslant 1} (-1)^i C_n^i \prod_{j=1}^{m} C^{k_j}_{C_{n-i}^{l_j}}, \tag{2} $$

где $C_j^i = \dfrac{j!}{i!(j-i)!}$ — биномиальный коэффициент, и соотношение

$$ \sum_{i \geqslant 0} C_n^i \, {}_{n-i}N^{k_1 k_2 \cdots k_m}_{l_1\, l_2 \cdots l_m} = \prod_{i=1}^{m} C^{k_i}_{C_n^{l_i}}. \tag{3} $$

В случае, когда в (1) $k_n = 1$, формула (2) имеет вид

$$ {}_nN(k_1, k_2, \ldots, k_{n-1}, 1) = \prod_{i=1}^{n-1} C^{k_i}_{C_n^i}. \tag{4} $$

Кроме того, принято, что

$$ {}_0N_0^0 = 1, \tag{5} $$

то есть число покрытий пустого множества нулевым количеством пустых подмножеств равно 1.

1. Производящие функции

Производящую функцию последовательности чисел (1) будем искать в виде

$$F(x;A_1,A_2,A_3,\ldots)=\sum_{n\geqslant 0}x^n\sum_{k\geqslant 0}{}_nN(k_1,k_2,\ldots,k_n)\prod_i A_i^{k_i},$$

где второе суммирование идет по всевозможным наборам $(k_1,k_2,\ldots,k_n)$, $k\geqslant 0$ — общее условие для каждого числа из набора. Таким образом, комбинаторное число ${}_nN(k_1,k_2,\ldots,k_n)$ является коэффициентом перед $x^n\prod\limits_i A_i^{k_i}$ в разложении производящей функции по степеням переменных x и A_i. Разложение функции (1) по степеням x представляет собой сумму $F(x;A_1,A_2,A_3,\ldots)=D_0+D_1x+D_2x^2+D_3x^3+\cdots$, где

$$D_n=\sum_{k\geqslant 0}{}_nN(k_1,k_2,\ldots,k_n)\prod_i A_i^{k_i} \tag{6}$$

— производящая функция последовательности чисел покрытий множества мощности n.

Подставим формулу для комбинаторных чисел (2) в правую часть выражения (6), учтём (5), поменяем порядок суммирования и воспользуемся биномиальной теоремой:

$$\sum_{k\geqslant 0}{}_nN(k_1,k_2,\ldots,k_n)\prod_i A_i^{k_i}=\sum_{k\geqslant 0}\sum_{j=0}^{n}(-1)^jC_n^j\prod_{i=1}^{n-j}C_{C_{n-j}^i}^{k_i}A_i^{k_i}=$$
$$=\sum_{j=0}^{n}(-1)^jC_n^j\sum_{k\geqslant 0}\prod_{i=1}^{n-j}C_{C_{n-j}^i}^{k_i}A_i^{k_i}=\sum_{j=0}^{n}(-1)^jC_n^j\prod_{i=1}^{n-j}(1+A_i)^{C_{n-j}^i}.$$

Таким образом,

$$D_n=\sum_{j\geqslant 0}(-1)^jC_n^j\prod_{i=1}^{n-j}(1+A_i)^{C_{n-j}^i}, \tag{7}$$

а производящая функция равна

$$F(x;A_1,A_2,A_3,\ldots)=\sum_{n\geqslant 0}x^n\sum_{j\geqslant 0}(-1)^jC_n^j\prod_{i=1}^{n-j}(1+A_i)^{C_{n-j}^i}. \tag{8}$$

Поменяем в (8) порядок суммирования:

$$F(x;A_1,A_2,A_3,\ldots)=\sum_{j\geqslant 0}\prod_{i=1}^{j}(1+A_i)^{C_j^i}\sum_{n\geqslant j}(-1)^{n-j}C_n^jx^n.$$

Воспользовавшись значением суммы $\sum\limits_{n\geqslant j}(-1)^{n-j}C_n^jx^n=\dfrac{x^j}{(1+x)^{j+1}}$, получаем выражение для производящей функции последовательности чисел (1)

$$F(x;A_1,A_2,A_3,\ldots)=\sum_{j\geqslant 0}\frac{x^j\prod\limits_{i=1}^{j}(1+A_i)^{C_j^i}}{(1+x)^{j+1}}. \tag{9}$$

С помощью аналогичных преобразований определим для чисел (1) экспоненциальную производящую функцию. Будем искать её в виде

$$E(x;A_1,A_2,A_3,\ldots)=\sum_{n\geqslant 0}\frac{x^n}{n!}\sum_{k\geqslant 0}{}_nN(k_1,k_2,\ldots,k_n)\prod_i A_i^{k_i} \tag{10}$$

или, воспользовавшись функциями (7),

$$E(x; A_1, A_2, A_3, \ldots) = \sum_{n \geqslant 0} D_n \frac{x^n}{n!},$$

то есть экспоненциальной функция является только по переменной x. Таким образом,

$$E(x; A_1, A_2, A_3, \ldots) = \sum_{n \geqslant 1} \frac{x^n}{n!} \sum_{j=0}^{n} (-1)^j C_n^j \prod_{i=1}^{n-j} (1 + A_i)^{C_{n-j}^i}. \tag{11}$$

Используем равенство $\dfrac{1}{n!} C_n^j = \dfrac{1}{j!(n-j)!}$, произведем замену $n - j \to j$ и поменяем порядок суммирования в (11). Получим выражение для экспоненциальной производящей функции последовательности чисел (1):

$$E(x; A_1, A_2, A_3, \ldots) = e^{-x} \sum_{j \geqslant 0} \frac{x^j}{j!} \prod_{i=1}^{j} (1 + A_i)^{C_j^i}. \tag{12}$$

2. Анализ производящих функций

Проанализируем выражение (12) для экспоненциальной производящей функции. Переменные A_i можно интерпретировать как множества мощности i. Тогда произведение

$$\prod_{i=1}^{n} (1 + A_i)^{C_n^i} \tag{13}$$

— это всевозможные покрытия множества мощности n и всех его подмножеств, включая пустое, а функция D_n (7) — покрытие множества мощности n после исключения из семейства покрытий (13) всех покрытий собственных подмножеств исходного множества.

В (7) переменная A_n содержится только в первом слагаемом (13), из этого факта следует формула (4). А из соотношения (3) следует, что

$$\sum_{i}^{n} C_n^i D_i = \prod_{j=1}^{n} (1 + A_j)^{C_n^j}.$$

Приравнивая нулю произвольные переменные A_i, получаем покрытия, включающие подмножества только определенных мощностей, например,

$$D_n(A_l) = \sum_{j=0}^{n} (-1)^j C_n^j (1 + A_l)^{C_{n-j}^l}$$

— покрытие множества мощности n подмножествами мощности l,

$$D_n(A_l, A_{l+1}) = \sum_{j=0}^{n} (-1)^j C_n^j (1 + A_l)^{C_{n-j}^l} (1 + A_{l+1})^{C_{n-j}^{l+1}}$$

— покрытие подмножествами мощности l и $l + 1$. Соответственно

$$E(x; A_l) = e^{-x} \sum_{j \geqslant 0} \frac{x^j}{j!} (1 + A_l)^{C_n^l} \tag{14}$$

— экспоненциальная производящая функция последовательности чисел покрытий множеств подмножествами мощности l, а

$$E(x; A_l, A_{l+1}) = e^{-x} \sum_{j \geqslant 0} \frac{x^j}{j!} (1 + A_l)^{C_n^l} (1 + A_{l+1})^{C_n^{l+1}} \tag{15}$$

— подмножествами мощности l и $l + 1$.

Приравняв в (13) все переменные A_i переменной A, с учетом равенства $\prod_{i=1}^{n} (1+A)^{C_n^i} = (1+A)^{2^n-1}$ получим экспоненциальную производящую функцию для последовательности чисел k-покрытий (покрытия, содержащие ровно k подмножеств [2]):

$$E(x; A) = e^{-x} \sum_{n \geqslant 0} \frac{x^n}{n!} (1 + A)^{2^n-1} = \sum_{n \geqslant 0} \frac{x^n}{n!} \sum_{j=0}^{n} (-1)^j C_n^j (1 + A)^{2^{n-j}-1}.$$

Раскроем скобки и поменяем порядок суммирования:

$$E(x; A) = \sum_{n \geqslant 0} \frac{x^n}{n!} \sum_{j=0}^{n} (-1)^j C_n^j \sum_{k=0}^{2^{n-j}-1} C_{2^{n-j}-1}^k A^k = \sum_{n \geqslant 0} \frac{x^n}{n!} \sum_{k=0}^{2^n-1} A^k \sum_{j=0} (-1)^j C_n^j C_{2^{n-j}-1}^k. \tag{16}$$

Коэффициент перед мономом $x^n A^k / n!$ в функции (16) — это количество k-покрытий множества мощности n [2]: $C_k = \sum\limits_{j=0} (-1)^j C_n^j C_{2^{n-j}-1}^i$.

Приравняв переменную A единице, получим экспоненциальную производящую функцию последовательности чисел покрытий множества мощности n [2, 3]:

$$E(x; 1) = \sum_{n \geqslant 0} \frac{x^n}{n!} \sum_{j=0}^{n} (-1)^j C_n^j 2^{2^{n-j}-1}. \tag{17}$$

Выражения, подобные (16) и (17), после аналогичных преобразований получаются из производящей функции (9).

Если производящая функция какой-либо последовательности чисел покрытий с фиксированными мощностями подмножеств является сложной функцией от функций E (12) (или от функции F (9), или от функций D_i (6)), т. е. $G(x, y; A_1, A_2, \ldots)) = G(E(x; A_1, A_2, \ldots), y)$, где y — дополнительная переменная, задающая ограничение на покрытия, то приравнивание всех переменных A_i переменной A дает следующую производящую функцию последовательности чисел k-покрытий:

$$A_i \to A \Rightarrow G(E(x; A_1, A_2, \ldots), y) \to G(E(x; A), y).$$

Справедливо и обратное преобразование

$$G(E(x; A), y) \to G(E(x; A_1, A_2, \ldots), y).$$

Таким же образом можно трансформировать производящую функцию, зависящую от $E(x; A)$, в производящую функцию, зависящую от $E(x; 1)$, то есть от производящей функции последовательности числа покрытий множества мощности n (17):

$$G(E(x; A), y) \to G(E(x; 1), y)$$

и обратно:

$$G(E(x;1),y) \to G(E(x;A),y).$$

Разложим производящую функцию (7) по степеням переменной A_1:

$$\begin{aligned} D_n = \sum_{k=0}^{n}(-1)^{n-k}C_n^k \sum_{i=0}^{k} C_k^i A_1^i \prod_{j=2}^{k}(1+A_j)^{C_k^j} = \sum_{k=0}^{n}(-1)^{n-k} \sum_{i=0}^{k} C_n^i C_{n-i}^{n-k} A_1^i \prod_{j=2}^{k}(1+A_j)^{C_k^j} = \\ = \sum_{i=0}^{n} C_n^i A_1^i \sum_{k=0}^{n-i}(-1)^k C_{n-i}^k \prod_{j=2}^{n-k}(1+A_j)^{C_{n-k}^j}. \end{aligned} \tag{18}$$

Здесь мы воспользовались свойством биномиальных коэффициентов [4] $C_n^k C_k^i = C_n^i C_{n-i}^{n-k}$ и поменяли порядок суммирования, произведя замену $n-k \to k$. Так как по определению $C_j^i = 0$ для $i > j \geqslant 0$, то сумму по k в (18) можно продолжить до n:

$$\sum_{k=0}^{n-i}(-1)^k C_{n-i}^k \cdots = \sum_{k=0}^{n}(-1)^k C_{n-i}^k \cdots$$

Для дальнейшего преобразования воспользуемся тождествами [4]

$$C_s^r = (-1)^r C_{r-s-1}^r, \quad \sum_r C_s^r C_p^{t-r} = C_{s+p}^t.$$

Преобразуем $(-1)^k C_{n-i}^k$ и разложим в сумму:

$$(-1)^k C_{n-i}^k = C_{k-n+i-1}^k = \sum_{l=0}^{i} C_i^l C_{k-n-1}^{k-l} = \sum_{l=0}^{i} C_i^l (-1)^{k-l} C_{n-l}^{k-l}.$$

Используем получившееся выражение для преобразования суммы в (18), затем поменяем порядок суммирования и произведем замену $k-l \to k$:

$$\begin{aligned} \sum_{k=0}^{n}(-1)^k C_{n-i}^k \prod_{j=2}^{n-k}(1+A_j)^{C_{n-k}^j} = \sum_{k=0}^{n}\sum_{l=0}^{i} C_i^l (-1)^{k-l} C_{n-l}^{k-l} \prod_{j=2}^{n-k}(1+A_j)^{C_{n-k}^j} = \\ = \sum_{l=0}^{i} C_i^l \sum_{k=0}^{n-l}(-1)^k C_{n-l}^k \prod_{j=2}^{n-k-l}(1+A_j)^{C_{n-k-l}^j}. \end{aligned}$$

Воспользуемся значением производящих функций D_n при $A_1 = 0$:

$$D_n(A_1=0) = \sum_{k=0}^{n}(-1)^k C_n^k \prod_{j=2}^{n-k}(1+A_j)^{C_{n-k}^j}.$$

Окончательно получим разложение D_n по степеням переменной A_1:

$$D_n = \sum_{i=0}^{n} C_n^i A_1^i \sum_{l=0}^{i} C_i^l D_{n-l}(A_1=0). \tag{19}$$

3. Получение рекуррентных соотношений

С помощью преобразований выражения для производящей функции (12) получим несколько рекуррентных соотношений.

Найдем частную производную по x от левой и правой частей выражения (12), предварительно умножив обе части на e^x:

$$\frac{\partial(e^x E)}{\partial x} = e^x\left(E + \frac{\partial E}{\partial x}\right) = \sum_{j\geqslant 0}\frac{x^j}{j!}\prod_{i=1}^{j+1}(1+A_i)^{C_{j+1}^i}.$$

Получившуюся в правой части сумму можно вывести с помощью следующих операций над функцией E:

$$\sum_{j\geqslant 0}\frac{x^j}{j!}\prod_{i=1}^{j+1}(1+A_i)^{C^i_{j+1}}=\frac{\partial(e^xE)}{\partial A_1}\frac{1+A_1}{x}.$$

Отсюда следует, что

$$x\left(E+\frac{\partial E}{\partial x}\right)=\frac{\partial E}{\partial A_1}(1+A_1). \tag{20}$$

Коэффициент перед каждым произведением

$$\frac{x^n}{n!}\prod_{i=1}^{n}A_i^{k_i} \tag{21}$$

в (10) при произвольных операциях над производящей функцией (вычисление производных и умножение на переменные) меняется следующим образом [4, 5]:

$$\begin{aligned}
E &\to {}_nN(k_1,\ldots,k_n);\\
xE &\to n\,{}_{n-1}N(k_1,\ldots,k_{n-1})\ \text{для } n>0;\\
x^sE &\to n^{\underline{s}}\,{}_{n-s}N(k_1,\ldots,k_{n-s})\ \text{для } n\geqslant s;\\
A_lE &\to {}_nN(\ldots,k_l-1,\ldots)\ \text{для } k_l>0;\\
\frac{\partial E}{\partial x} &\to {}_{n+1}N(k_1,\ldots,k_n,0);\\
\frac{\partial^m E}{\partial x^m} &\to {}_{n+m}N(k_1,\ldots,k_n,0,\ldots,0);\\
\frac{\partial E}{\partial A_l} &\to (k_l+1)\,{}_nN(\ldots,k_l+1,\ldots);\\
x\frac{\partial E}{\partial x} &\to n\,{}_nN(k_1,\ldots,k_n);\\
x^s\frac{\partial^m E}{\partial x^m} &\to n^{\underline{s}}\,{}_{n+m-s}N(k_1,\ldots,k_{n+m-s})\ \text{для } s\geqslant m;\\
A_l\frac{\partial E}{\partial A_l} &\to k_l\,{}_nN(\ldots,k_l,\ldots),
\end{aligned} \tag{22}$$

где $n^{\underline{s}}=n(n-1)\cdots(n-s+1)$ — убывающая степень [4]. Применяя правила (22) к (20), получаем рекуррентное соотношение

$$n\,{}_{n-1}N(k_1,\ldots)+n\,{}_nN(k_1,\ldots)=(k_1+1)\,{}_nN(k_1+1,\ldots)+k_1\,{}_nN(k_1,\ldots),$$

или, перейдя от n к $n+1$,

$${}_{n+1}N(k_1+1,\ldots)(k_1+1)={}_{n+1}N(k_1,\ldots)(n-k_1+1)+{}_nN(k_1,\ldots)(n+1).$$

В случае производной степени s от произведения e^xE

$$\frac{\partial^s(e^xE)}{\partial x^s}=e^x\left(E+\sum_{i=1}^{s}C^i_s\frac{\partial^iE}{\partial x^i}\right)=\sum_{j\geqslant 0}\frac{x^j}{j!}\prod_{i=1}^{j+s}(1+A_i)^{C^i_{j+s}}$$

правую часть выражения можно получить с помощью преобразований

$$\sum_{j\geqslant 0}\frac{x^j}{j!}\prod_{i=1}^{j+s}(1+A_i)^{C^i_{j+s}}=s!\frac{\partial(e^xE)}{\partial A_s}\frac{1+A_s}{x^s}.$$

Таким образом,

$$x^s\left(E+\sum_{i=1}^{s}C_s^i\frac{\partial^i E}{\partial x^i}\right)=s!\frac{\partial(e^xE)}{\partial A_s}\frac{1+A_s}{x^s}.$$

Используя правила (22) и тождество $\dfrac{n^{\underline{s}}}{s!}=C_n^s$, получаем соотношение

$$C_n^s\sum_{i=0}^{s}C_s^i\,{}_{n+m-i}N(k_1,\ldots,k_n)=(k_s+1)_n N(\ldots,k_s+1,\ldots)+k_s\,{}_nN(\ldots,k_s,\ldots).$$

Для получения следующего рекуррентного соотношения воспользуемся свойством биномиальных коэффициентов $C_n^l+C_n^{l+1}=C_{n+1}^{l+1}$. В производящей функции (15) при замене всех переменных A_l на переменные A_{l+1}

$$A_l\to A_{l+1}\Rightarrow(1+A_l)^{C_n^l}(1+A_{l+1})^{C_n^{l+1}}\to(1+A_{l+1})^{C_{n+1}^{l+1}}$$

получаем произведение, стоящее в выражении для экспоненциальной производящей функции (14) перед x^{n+1}:

$$\sum_n\frac{x^n}{n!}(1+A_{l+1})^{C_{n+1}^{l+1}}=\frac{\partial}{\partial x}\left(\sum_n\frac{x^n}{n!}(1+A_{l+1})^{C_n^{l+1}}\right).$$

Таким образом,

$$E(x;A_l,A_{l+1})\big|_{A_l\to A_{l+1}}=E(x;A_{l+1})+\frac{\partial E(x;A_{l+1})}{\partial x}.$$

При проведении замены всех переменных A_l переменными A_{l+1} коэффициент перед произведением (21) равен $\sum\limits_{k_1+k_2=k}{}_nN_{l,\ l+1}^{k_1,k_2}$. Получаем равенство

$$\sum_{k_1+k_2=k}{}_nN_{l,\ l+1}^{k_1,k_2}={}_nN_{l+1}^{k}+{}_{n+1}N_{l+1}^{k},$$

или, после преобразования,

$${}_{n+1}N_{l+1}^{k}=\sum_{\substack{k_1+k_2=k,\\ k_1>0}}{}_nN_{l,\ l+1}^{k_1,k_2}.$$

В выражении (15) можно не все переменные A_l заменить переменными A_{l+1}, а оставить одну переменную в каждом слагаемом неизмененной:

$$\left[C_n^l-1\right]A_l\to A_{l+1}\Rightarrow(1+A_l)^{C_n^l}(1+A_{l+1})^{C_n^{l+1}}\to C_n^l(1+A_l)(1+A_{l+1})^{C_{n+1}^{l+1}-1}. \qquad (23)$$

Здесь $[k]A_l$ означает, что следующее действие производим с k переменными A_l; появляется коэффициент C_n^l, так как выбрать одну неизменяемую переменную A_l можно именно таким числом способов.

Такое же выражение с точностью до константы можно получить в результате замены одной из переменных A_{l+1} переменной A_l:

$$[1]\,A_{l+1}\to A_l\Rightarrow(1+A_{l+1})^{C_{n+1}^{l+1}}\to C_{n+1}^{l+1}(1+A_l)(1+A_{l+1})^{C_{n+1}^{l+1}-1}. \qquad (24)$$

После преобразований (23) и (24) сумма (15) будет равна в первом случае

$$\sum_{n\geqslant l} C_n^l \frac{x^n}{n!}(1+A_l)(1+A_{l+1})^{C_{n+1}^{l+1}-1} = \sum_{n\geqslant l} \frac{x^n}{l!(n-l)!}(1+A_l)(1+A_{l+1})^{C_{n+1}^{l+1}},$$

во втором —

$$\sum_{n\geqslant l} C_{n+1}^{l+1} \frac{x^{n+1}}{(n+1)!}(1+A_l)(1+A_{l+1})^{C_{n+1}^{l+1}-1} = \sum_{n\geqslant l} \frac{x^{n+1}}{(l+1)!(n-l)!}(1+A_l)(1+A_{l+1})^{C_{n+1}^{l+1}}.$$

Сравнивая полученные выражения, выведем соотношение для функции E:

$$x\ E(x; A_l, A_{l+1})|_{[\max -1]A_l \to A_{l+1}} = (l+1)\ E(x; A_{l+1})|_{[1]A_{l+1}\to A_l}\,.$$

Здесь $[\max -1]\,A_l$ — следующее действие производим со всеми переменными A_l в мономе, кроме одной. В каждом мономе $A_l^{k_1}A_{l+1}^{k_2}$ можно оставить без изменения одну из k_1 переменных A_l, а в мономе A_{l+1}^k — изменить одну из k переменных A_{l+1}. Следовательно, верно равенство

$$n \sum_{k_1+k_2=k} k_1\, {}_{n-1}N_{l,\ l+1}^{k_1,k_2} = k(l+1)\, {}_nN_{l+1}^k,$$

или после преобразования

$${}_{n+1}N_{l+1}^k = \frac{(n+1)}{k(l+1)} \sum_{k_1+k_2=k} k_1\, {}_nN_{l,\ l+1}^{k_1,k_2}.$$

Для вывода следующего рекуррентного соотношения сравним правые части выражений (6) и (19) при одинаковых степенях A_1:

$$\sum_{k\geqslant 0} {}_nN(k_1, k_2, \ldots, k_n) \prod_{i>1} A_i^{k_i} = C_n^{k_1} \sum_{j=0}^{k_1} C_{k_1}^j D_{n-j}(A_1 = 0). \tag{25}$$

Подставим значение $D_{n-j}(A_1 = 0)$ в (25) и приравняем коэффициенты при одинаковых мономах $\prod A_i^{k_i}$:

$${}_nN(k_1, k_2, \ldots, k_{n-k_1}, 0, \ldots, 0) = C_n^{k_1} \sum_{j=0}^{k_1} C_{k_1}^j\, {}_{n-j}N(0, k_2, \ldots, k_{n-k_1}, 0, \ldots, 0).$$

Заключение

В работе выведены выражения для обычной и экспоненциальной производящих функций последовательности чисел покрытий. Полученные выражения могут использоваться для анализа различного рода неповторяющихся покрытий конечного множества. Интерпретируя переменные A_i как подмножества, можно более наглядно конструировать различные покрытия и множества покрытий с определенными ограничениями. Показано, как с помощью разнообразных преобразований выражений для производящих функций можно получать рекуррентные соотношения. Все выведенные в статье рекуррентные соотношения можно использовать для вычисления чисел покрытий конечного множества с помощью меньшего количества операций и без больших промежуточных результатов, возникающих из-за быстро растущих значений $C_{C_n^l}^k$.

ЛИТЕРАТУРА

1. *Ганопольский Р. М.* Число неупорядоченных покрытий конечного множества подмножествами фиксированного размера // Прикладная дискретная математика. 2010. № 4(10). С. 5–17
2. *Macula A. J.* Covers of a finite set // Mathematics Magazine. V. 67. No. 2. P. 141–144.
3. *Comtet L.* Advanced Combinatorics. The Art of Finite and Infinate Expansions. Dordrecht, Holland: D. Reidel Publishing Company, 1974.
4. *Кнут Д., Грэхем Ф., Поташник О.* Конкретная математика. Основание информатики. М.: Мир, 2006.
5. *Ландо С. К.* Лекции о производящих функциях. М.: МЦНМО, 2002.

2011 Теоретические основы прикладной дискретной математики №1(11)

DOI 10.17223/20710410/11/2

УДК 519.7

О ВЫДЕЛЕНИИ МАКСИМАЛЬНЫХ ПОДКЛОНОВ[1]

Н. Г. Парватов

Томский государственный университет, г. Томск, Россия

E-mail: parvatov@mail.tsu.ru

Рассматривается задача выделения максимальных (предполных) подклонов в произвольном клоне, важная в связи с проблемой полноты в нём. Вводятся и-описания и расширенные и-описания как средства задания подклона в клоне. Устанавливаются необходимые и достаточные условия максимальности подклона, заданного своим расширенным и-описанием. Рассматриваются примеры.

Ключевые слова: *клон, подклон, предполный подклон, максимальный подклон, проблема полноты, критериальная система, описание подклона, и-описание подклона.*

Введение

Пусть E — непустое конечное множество. Будем рассматривать множество P_E функций $f : E^n \to E$, где n — произвольное натуральное число. Множество функций из P_E, замкнутое операциями суперпозиции из [1, 2] и включающее множество S_E селекторных функций, тождественно равных некоторой переменной, называется *клоном*.

Пусть B — произвольный клон функций из P_E. Множество функций клона B будем рассматривать с *замыканием относительно S_E-суперпозиции* таким, что замыканием произвольного множества X функций из клона B относительно S_E-суперпозиции является наименьший по включению среди содержащих это множество клон $[X]_{\cup S_E} = [X \cup S_E]$. Для него множество X называется *порождающим*.

Проблема полноты в клоне B состоит в описании всех его порождающих подмножеств. Она может быть решена указанием *критериальной системы* $\mathcal{S}$ собственных подклонов клона B (то есть подклонов, собственным образом в нём содержащихся), такой, что всякий собственный подклон клона B включён в некоторый из клонов системы $\mathcal{S}$.

В работе рассматривается задача выделения в клоне B максимальных собственных подклонов (далее называемых просто *максимальными в B*), очевидно содержащихся в любой критериальной системе клона B, а потому представляющих интерес в связи с проблемой полноты в нём.

Структура статьи следующая. В п. 1 рассматриваются соответствия Галуа из [3, 4] между множествами частичных или полностью определённых функций k-значной логики и множествами предикатов. На основе этого формулируется лемма о доопределении из [5] и в качестве средства задания подклонов вводятся и-описания. В п. 2 вводятся расширенные и-описания; устанавливается лемма о выделении и теорема о выделении, дающие условия максимальности подклона, заданного своим расширенным и-описанием. Использование доказанной теоремы иллюстрируется примером. В п. 3 формулируются достаточные условия максимальности, рассматриваются примеры.

[1]Работа выполнена в рамках реализации ФЦП «Научные и научно-педагогические кадры инновационной России» на 2009–2013 гг. (гос. контракт № П1010).

1. Соответствия Галуа и и-описания

Напомним для дальнейшего использования о классических соответствиях Галуа из [3, 4], определяемых для функций многозначной логики и для частичных таких функций отношением сохранения функцией предиката. Сформулируем лемму о доопределении из [5], которая даёт условия, когда частичная функция имеет доопределение в заданном клоне. На основе этого в качестве средства выделения подклонов введём в рассмотрение и-описания и расширенные и-описания.

Соответствие Галуа для полностью определённых функций. Будем обозначать через Π_E множество предикатов $p : E^m \to \{\text{И,Л}\}$ при всевозможных натуральных m. Говорят, что функция f из P_E, зависящая от n переменных, *сохраняет предикат* p из Π_E, зависящий от m переменных, если для любых наборов $X_1, \ldots, X_n$, удовлетворяющих предикату p, ему удовлетворяет также набор $X_0 = f^{[m]}(X_1, \ldots, X_n)$, полученный последовательным выписыванием значений функции f, вычисленных от строк матрицы $(X_1^T, \ldots, X_n^T)$, где верхний индекс T означает транспонирование. Отношение сохранения функцией из множества P_E предиката из множества Π_E определяет соответствие Галуа [6] из [4] между упорядоченными включением системами подмножеств этих множеств, важное при изучении клонов. При этом соответствии произвольному множеству X функций из P_E сопоставляется множество $\mathrm{inv}_E(X)$ сохраняемых ими предикатов из Π_E, а произвольному множеству Y предикатов из Π_E сопоставляется множество $\mathrm{pol}_E(Y)$ сохраняющих их функций из P_E. Галуа-замкнутыми классами функций в этом случае оказываются всевозможные *клоны* функций из P_E. Иными словами, замыкание Галуа в множестве P_E совпадает с замыканием относительно S_E-суперпозиции. Галуа-замкнутыми классами предикатов оказываются всевозможные множества предикатов из Π_E, замкнутые операциями подстановки переменных и включающие все диагонали (к числу которых относятся тождественно истинные или ложные предикаты, а также предикаты, выражаемые формулами вида $x_i = x_j \wedge \ldots \wedge x_l = x_m$, где $\{i, j, \ldots, l, m\} = \{1, \ldots, n\}$ для некоторого натурального n). В дальнейшем такие множества предикатов будем называть *замкнутыми*. Отметим, что *замкнутый класс* предикатов из Π_E, *порождённый* множеством X, то есть наименьший по включению среди включающих это множество, совпадает с классом $\mathrm{inv}_E(\mathrm{pol}_E(X))$ и состоит из предикатов, выражаемых формулами первого порядка (здесь достаточно предварённых формул), в которых переменные принимают значения в множестве E, предикатные символы интерпретируются в множестве $X \cup \{\text{И,Л}, =\}$, функциональные символы отстутствуют, а из логических символов возможны только квантор существования и конъюнкция.

Соответствие Галуа для частичных функций. Наряду с функциями из P_E станем рассматривать функции $f : E^n \to E \cup \{*\}$, где n — произвольное натуральное число и $*$ — фиксированный элемент, не принадлежащий множеству E. Интерпретируя этот элемент как неопределённое значение, станем называть и считать такие функции *частичными*, а их множество обозначать через P_E^*. Говорят, что частичная функция f из P_E^*, зависящая от n переменных, *сохраняет предикат* p из Π_E, зависящий от m переменных, если для любых наборов $X_1, \ldots, X_n$, удовлетворяющих предикату p, набор $X_0 = f^{[m]}(X_1, \ldots, X_n)$, полученный последовательным выписыванием значений функции f, вычисленных от строк матрицы $(X_1^T, \ldots, X_n^T)$, либо также удовлетворяет предикату p, либо содержит неопределённую компоненту, равную $*$.

Отношение сохранения функцией из множества P_E^* предиката из множества Π_E определяет соответствие Галуа из [3] между упорядоченными включением систе-

мами подмножеств этих множеств. При этом соответствии произвольному множеству X функций из P_E^* сопоставляется множество $\mathrm{inv}_E(X)$ сохраняемых ими предикатов из Π_E, а произвольному множеству Y предикатов из Π_E сопоставляется множество $\mathrm{pol}_E^*(Y)$ сохраняющих их функций из P_E^*. Галуа-замкутыми классами функций из P_E^* оказываются теперь всевозможные *строго частичные клоны* — замкнутые операциями суперпозиции классы частичных функций из P_E^*, включающие множество S_E всех селекторов и содержащие вместе с каждой своей функцией всякое её ограничение (получаемое заменой некоторых значений, принимаемых функцией, неопределённым значением $*$). Галуа-замкнутыми классами предикатов оказываются всевозможные множества предикатов из Π_E, замкнутые операциями подстановки переменных и включающие все диагонали. Такие классы предикатов в дальнейшем называются *и-замкнутыми классами*, или, короче, *и-классами* предикатов. Отметим, что *и-класс, порождённый* множеством X предикатов из Π_E, то есть наименьший по включению среди включающих это множество, совпадает с классом $\mathrm{inv}_E(\mathrm{pol}_E^*(X))$ и состоит из предикатов, выражаемых бескванторными формулами первого порядка, в которых переменные принимают значения в множестве E, предикатные символы интерпретируются в множестве $X \cup \{\text{И},\text{Л},=\}$, функциональные символы отсутствуют, а из логических символов возможна только конъюнкция.

Описания, и-описания и доопределения. Далее будут сформулированы условия из [5] существования доопределения в заданном клоне для произвольной частичной функции и с их помощью будут введены новые методы задания подклонов — посредством и-описаний.

Для произвольных клонов B и C функций из P_E, таких, что $B \subseteq C$, множество Y предикатов из Π_E (на самом деле из $\mathrm{inv}_E(B)$, как выяснится позднее) назовём *описанием* клона B в клоне C, если выполняются равенства

$$B = C \cap \mathrm{pol}_E(Y) \quad \text{и} \quad \mathrm{inv}_E(B) = \mathrm{inv}_E \mathrm{pol}_E(\mathrm{inv}_E(C) \cup Y),$$

равносильные в силу рассмотренного выше соответствия Галуа из [4]. Первое из этих равенств указывает на то, что клон B состоит из функций клона C, сохраняющих все предикаты из множества Y; второе говорит о том, что замкнутый класс $\mathrm{inv}_E(B)$ инвариантных для B предикатов порождается (с помощью операций подстановки переменных, конъюнкции и проектирования) предикатами из множества $\mathrm{inv}_E(C) \cup Y$ (отсюда и включение $Y \subseteq \mathrm{inv}_E(B)$).

По аналогии с этим для произвольных строго частичных клонов B' и C' функций из P_E^*, таких, что $B' \subseteq C'$, множество Y предикатов из Π_E (точнее, из $\mathrm{inv}_E(B')$) назовём *и-описанием* частичного клона B' в частичном клоне C', если выполняются равенства

$$B' = C' \cap \mathrm{pol}_E^*(Y) \quad \text{и} \quad \mathrm{inv}_E(B') = \mathrm{inv}_E \mathrm{pol}_E^*(\mathrm{inv}_E(C') \cup Y), \tag{1}$$

равносильные в силу рассмотренного выше соответствия Галуа из [3]. Первое из этих равенств указывает на то, что частичный клон B' состоит из функций частичного клона C', сохраняющих все предикаты из множества Y; второе говорит о том, что и-замкнутый класс $\mathrm{inv}_E(B')$ инвариантных для строго частичного клона B' предикатов порождается (с помощью операций подстановки переменных и конъюнкции, без операции проектирования) предикатами из множества $\mathrm{inv}_E(C') \cup Y$.

Далее. Для произвольного клона D обозначим через D^* частичный клон, состоящий из всех частичных функций, имеющих доопределение в клоне D. При этом под *доопределением* частичной функции f из P_E^*, зависящей от n переменных, как обычно,

понимаем всякую зависящую от n переменных частичную функцию g из P_E^* (в частности, из P_E), такую, что для любого набора a из множества E^n значения $f(a)$ и $g(a)$ совпадают или значение $f(a)$ не определено (равно $*$).

Сделанное выше определение и-описания имеет смысл и для частичных клонов $B' = B^*$ и $C' = C^*$, где по-прежнему B и C — клоны функций из P_E и $B \subseteq C$. В этой ситуации множество Y будем называть *и-описанием* клона B в клоне C. Заметим, что в этом случае $\mathrm{inv}_E(B) = \mathrm{inv}_E(B^*)$ и $\mathrm{inv}_E(C) = \mathrm{inv}_E(C^*)$, вследствии чего равенствам в (1) равносильно ещё одно:

$$\mathrm{inv}_E(B) = \mathrm{inv}_E \mathrm{pol}_E^*(\mathrm{inv}_E(C) \cup Y), \tag{2}$$

означающее, что и-замкнутый класс $\mathrm{inv}_E(B)$ инвариантных для клона B предикатов порождается (с помощью операций конъюнкции и подстановки переменных) предикатами из множества $\mathrm{inv}_E(C) \cup Y$. Таким образом, получаем следующую лемму из [5].

Лемма 1. Для любых клонов B и C, таких, что $B \subseteq C$, и любого множества Y предикатов из $\mathrm{inv}_E(B)$ равносильны условия

$$B^* = C^* \cap \mathrm{pol}_E^*(Y) \quad \text{и} \quad \mathrm{inv}_E(B) = \mathrm{inv}_E \mathrm{pol}_E^*(\mathrm{inv}_E(C) \cup Y).$$

В частности, для любого клона B и любого множества Y предикатов из $\mathrm{inv}_E(B)$ равносильны условия

$$B^* = \mathrm{pol}_E^*(Y) \quad \text{и} \quad \mathrm{inv}_E(B) = \mathrm{inv}_E \mathrm{pol}_E^*(Y).$$

Иными словами, второе утверждение леммы говорит о том, что для любого клона B и любого множества Y предикатов из $\mathrm{inv}_E(B)$ равносильны следующие условия:

1) произвольная частичная функция из P_E^* имеет доопределение в клоне B, если она сохраняет все предикаты из множества Y;
2) множество Y порождает и-класс $\mathrm{inv}_E(B)$.

Первое утверждение леммы допускает аналогичную переформулировку.

Таким образом, лемма 1 даёт условия, при которых множество предикатов порождает инвариантный и-класс, и одновременно условия, при которых частичная функция имеет доопределение в клоне. Из-за этого будем называть её *леммой о доопределении*.

Лемма 1 о доопределении является удобным инструментом изучения инвариантных предикатов.

2. Теорема о выделении

Установим теорему о выделении, дающую необходимые и достаточные условия максимальности подклона, заданного в клоне B своим и-описанием. Эта теорема позволяет распознавать максимальность подклонов и может использоваться для выделения максимальных подклонов. Рассмотрим примеры её использования, иллюстрирующие одновременно и применение леммы о доопределнии.

Расширенные и-описания и задача выделения

Сформулируем задачу более точно. Для этого понадобятся некоторые определения.

Будем говорить, что предикат p, зависящий от $m-1$ переменной, получен из предиката q, зависящего от m переменных, *отождествлением двух переменных*, если эти предикаты для каких-то различных чисел i и j из множества $\{1, \ldots, m\}$ связаны соотношением

$$p(x_1, \ldots, \hat{x}_i, \ldots, x_m) \equiv q(x_1, \ldots, x_{i-1}, x_j, x_{i+1}, \ldots, x_m),$$

где крышечкой сверху помечена отсутствующая переменная. Будем говорить, что предикат p, зависящий от $m-t$ переменных, где $1 \leqslant t \leqslant m-1$, получен из предиката q *отождествлением переменных*, если в некоторой последовательности предикатов $p_1, \ldots, p_t$, такой, что $p_1 = q$ и $p_t = p$, каждый предикат, начиная со второго, получен из предыдущего отождествлением каких-то двух переменных.

Переменная x_i называется *фиктивной* переменной предиката $p(x_1, \ldots, x_m)$, если выполняется соотношение

$$\forall x_i p(x_1, \ldots, x_m) \equiv \exists x_i p(x_1, \ldots, x_m).$$

Пусть предикаты p и q связаны соотношением

$$p(x_1, \ldots, \hat{x}_{i_1}, \ldots, \hat{x}_{i_t}, \ldots, x_m) \equiv \exists x_{i_1} \ldots \exists x_{i_t} q(x_1, \ldots, x_m),$$

где $1 \leqslant i_1 < \ldots < i_t \leqslant m$ и крышечкой сверху помечены отсутствующие переменные. Будем говорить, что предикат p получен из предиката q *проектированием* и предикат p является *проекцией* предиката q, если $t \geqslant 1$. Будем говорить, что предикат p получен из предиката q *удалением фиктивных переменных*, если $t \geqslant 0$ и переменные $x_{i_1}, \ldots, x_{i_t}$ составляют множество всех фиктивных переменных предиката $q(x_1, \ldots, x_m)$.

Рассмотрим произвольный подклон K клона B. Пусть множество Y предикатов из $\mathrm{inv}_E(K)$ является и-описанием подклона K в клоне B, то есть выполняются равенства $\mathrm{inv}_E(K) = \mathrm{inv}_E \mathrm{pol}_E^*(Y \cup \mathrm{inv}_E(B))$ и $K^* = B^* \cap \mathrm{pol}_E^*(Y)$. И-описание Y подклона K в клоне B назовём *расширенным*, если оно состоит из предикатов без фиктивных переменных и вместе с любым своим предикатом p содержит всякий неинвариантный для клона B (то есть не принадлежащий и-классу $\mathrm{inv}_E(B)$) предикат, который можно получить удалением фиктивных переменных из предиката, возникающего в результате отождествления переменных у предиката p. Понятно, что всякий подклон K клона B обладает расширенным и-описанием, которое обычно несложно получается по произвольному и-описанию.

Будем интересоваться условиями, при которых подклон K клона B, заданный своим расширенным и-описанием Y, является максимальным.

Предельные предикаты. Понадобятся некоторые определения и замечания. В них p и q — предикаты из Π_E, зависящие от m переменных.

1. Неинвариантный для клона B предикат p из множества $\Pi_E \setminus \mathrm{inv}_E(B)$ будем называть *B-предельным по проектированию*, если всякая проекция предиката p инвариантна для клона B, то есть принадлежит множеству $\mathrm{inv}_E(B)$.

2. Неинвариантный для клона B предикат p из множества $\Pi_E \setminus \mathrm{inv}_E(B)$ будем называть *B-предельным по отождествлению*, если предикаты, полученные из предиката p отождествлением переменных, инвариантны для B.

3. Отождествляя предикат с его областью истинности (подобное отождествление предикатов и отношений принято в дискретной математике), станем использовать для предикатов теоретико-множественные отношения и операции. В частности, для m-местных предикатов p и q *включение* $p \subseteq q$ означает, что выполняется соотношение

$$p(x_1, \ldots, x_m) \Rightarrow q(x_1, \ldots, x_m),$$

а *пересечение* $p \cap q$ и *разность* $p \setminus q$ предикатов p и q определяются соотношениями

$$(p \cap q)(x_1, \ldots, x_m) \equiv p(x_1, \ldots, x_m) \wedge q(x_1, \ldots, x_m),$$

$$(p \setminus q)(x_1, \ldots, x_m) \equiv p(x_1, \ldots, x_m) \wedge \neg q(x_1, \ldots, x_m).$$

4. Для любого натурального $n \geqslant 1$ и функции $\alpha : \{1, \ldots, m\} \to \{1, \ldots, n\}$ определим n-местный предикат $p_\alpha^{(n)}$ как

$$p_\alpha^{(n)}(x_1, \ldots, x_n) \equiv p(x_{\alpha(1)}, \ldots, x_{\alpha(m)}).$$

Будем писать p_α вместо $p_\alpha^{(n)}$, если $n = m$. Если при $n = m$ функция α является подстановкой на множестве чисел $1, \ldots, m$, то будем говорить, что предикаты p_α и $p \cap p_\alpha$ получены из предиката p *перестановкой переменных* и *симметризацией* соответственно, а предикаты p и p_α будем называть *перестановочно эквивалентными*.

5. Неинвариантный для клона B предикат p из множества $\Pi_E \setminus \mathrm{inv}_E(B)$ станем называть *B-предельным по симметризации*, если любой предикат, полученный из предиката p симметризацией, совпадает с p либо инвариантен для клона B.

6. Через Bp станем обозначать m-местный предикат, которому удовлетворяют исключительно всевозможные наборы $f^{[m]}(X_1, \ldots, X_n)$, полученные последовательным вычислением значений функции f от строк матрицы $(X_1^T, \ldots, X_n^T)$, где n — произвольное натуральное число, функция f от n переменных выбирается произвольно в клоне B, и наборы $X_1, \ldots, X_n$, удовлетворяющие предикату p, также выбираются произвольно. Имеет место следующая

Лемма 2. Для любого клона B функций из P_E и любых предикатов p и q из Π_E, зависящих от m переменных, следующие условия равносильны:

1) имеют место включения $q \subseteq p$ и $(Bq) \setminus q \subseteq (Bp) \setminus p$;
2) имеет место равенство $q = (Bq) \cap p$;
3) имеет место равенство $q = p' \cap p$ для некоторого предиката p' из $\mathrm{inv}_E(B)$, зависящего от m переменных.

Если для предикатов p и q выполняются условия 1–3, то выполняется и включение $B \cap \mathrm{pol}_E(p) \subseteq B \cap \mathrm{pol}_E(q)$.

Доказательство. Если выполняется первое условие, то включение $q \subseteq (Bq) \cap p$ очевидно, а обратное включение проверяется непосредственно; таким образом, выполняется второе условие, из которого третье следует очевидным образом. Пусть выполняется третье условие. Отметим в этом случае «включения»

$$q \subseteq Bq \subseteq Bp' = p',$$

из которых первое имеет место, поскольку клон B содержит селекторы, второе выполняется в силу включения $q \subseteq p'$, имеющего место из-за третьего условия, а равенство выполняется из-за инвариантности предиката p' для клона B. В силу этого вслед за третьим условием выполняется и второе. Тогда в первом условии первое включение очевидно, а второе проверяется непосредственно. Тем самым имеет место первое условие. Первое утверждение леммы доказано.

Для доказательства второго утверждения предположим, что функция f из клона B, зависящая от n переменных, не сохраняет предикат q, удовлетворяющий вместе с предикатом p условиям 1–3. Тогда найдутся наборы $X_1, \ldots, X_n$, удовлетворяющие предикату q, в отличие от набора $X_0 = f^{[m]}(X_1, \ldots, X_n)$. Тогда в силу включений из первого условия наборы $X_1, \ldots, X_n$ удовлетворяют также и предикату p, в отличие от набора X_0. Таким образом, функция f не сохраняет предикат p. Лемма доказана. ■

Предикат q будем называть *B-сужением* предиката p, если для этих предикатов выполняются равносильные условия 1–3 из леммы 2. Неинвариантный для клона B

предикат p из множества $\Pi_E \setminus \mathrm{inv}_E(B)$ станем называть *B-предельным по сужению*, если любой предикат, полученный из предиката q B-сужением, совпадает с p или инвариантен для клона B.

7. Отметим, что для любого предиката p', полученного из предиката p проектированием, отождествлением переменных, симметризацией или B-сужением, выполняется включение $B \cap \mathrm{pol}_E(p) \subseteq B \cap \mathrm{pol}_E(p')$.

8. В соответствии с определениями, если предикат p является B-предельным по проектированию, отождествлению, симметризации или сужению, то клон $K = B \cap \mathrm{pol}_E(p)$ обладает свойством $K \subset B$.

9. *Всякий клон $K' \subset B$ можно расширить до клона $K = B \cap \mathrm{pol}_E(p)$, где $K' \subseteq \subseteq K \subset B$ и предикат p обладает любым наперёд заданным свойством B-предельности* по проектированию, отождествлению, симметризации или сужению. (Для доказательства необходимо выбрать произвольно предикат p из множества $\mathrm{inv}_E(K) \setminus \mathrm{inv}_E(B)$, непустого в силу строгого включения $K' \subset B$, а затем, сохраняя за предикатом p его обозначение, выполнять над ним соответствующие операции проектирования, отождествления переменных, симметризации или сужения, пока они не выводят предикат p из множества $\mathrm{inv}_E(K) \setminus \mathrm{inv}_E(B)$.)

10. Отсюда *критериальную систему клона B*, возможно избыточную, *составляют клоны $B \cap \mathrm{pol}_E(p)$, где предикат p обладает всеми указанными выше свойствами B-предельности* (достаточно любым зафиксированным набором этих свойств). Это даёт способ отыскания критериальной системы, требующий нахождения предикатов с фиксированным набором свойств B-предельности. Можно показать, что, найдя все предикаты, B-предельные по отождествлению, получаем критериальную систему клона B, причём конечную для конечно порождаемого клона B. Если вместо этого взять предикаты, B-предельные по проектированию, то конечность критериальной системы для конечно порождаемого клона B не гарантирована известными автору теоремами.

11. В силу сказанного, *всякий максимальный подклон клона B имеет вид $B \cap \mathrm{pol}_E(p)$, где предикат p обладает свойствами B-предельности по проектированию, отождествлению, симметризации и сужению.*

Теорема о выделении. Сделанные выше замечания позволяют теперь вернуться к задаче распознавания свойства максимальности подклона K, заданного в клоне B своим расширенным и-описанием Y. Основной является следующая

Лемма 3. Пусть K и B — клоны функций из P_E, такие, что $K \subset B$, а множество Y предикатов из $\mathrm{inv}_E(K)$ является расширенным и-описанием клона K в клоне B. Пусть также для B-предельного по проектированию и отождествлению предиката q из множества $\Pi_E \setminus \mathrm{inv}_E(B)$, зависящего от m переменных, выполняется включение $K \subseteq B \cap \mathrm{pol}_E(q)$. Тогда предикат q можно представить в виде

$$q = (Bq) \cap q_1 \cap \ldots \cap q_l$$

для некоторого целого положительного числа l, где каждый из предикатов $q_1, \ldots, q_l$ зависит от m переменных и перестановочно эквивалентен некоторому предикату из Y.

Доказательство. Нетривиальная часть леммы состоит в том, что каждый из предикатов $q_1, \ldots, q_l$ зависит от всех переменных предиката q. Для доказательства этого достаточно убедиться, что для любого набора X_0, удовлетворяющего предикату $Bq \setminus q$, найдётся предикат q^{X_0}, перестановочно эквивалентный некоторому предикату из множества Y, такой, что $q \subseteq q^{X_0}$ и набор X_0 не удовлетворяет предикату q^{X_0}.

Действительно, тогда в качестве предикатов $q_1, \ldots, q_l$ можно выбрать всевозможные предикаты q^{X_0}.

Итак, пусть набор X_0 удовлетворяет предикату $Bq \setminus q$, а наборы $X_1, \ldots, X_n$ составляют область истинности предиката q. Заметим, что в этом случае в силу B-предельности предиката q по отождествлению матрица $X = (X_1^T, \ldots, X_n^T)$ не содержит повторяющихся строк. Рассмотрим частичную n-местную функцию f, определённую соотношением

$$f^{[m]}(X_1, \ldots, X_n) = X_0$$

и принимающую неопределённое значение $*$ в остальных случаях — на наборах, не являющихся строками матрицы X. Функция f определена корректно и имеет доопределение в клоне B в силу выбора набора X_0, удовлетворяющего предикату Bq. Она не сохраняет предикат q также в силу выбора набора X_0, не удовлетворяющего q, в отличие от наборов $X_1, \ldots, X_n$. С использованием включения $K \subseteq B \cap \mathrm{pol}_E(q)$ получаем соотношения

$$B^* \cap \mathrm{pol}_E^*(Y) = K^* \subseteq (B \cap \mathrm{pol}_E(q))^* \subseteq B^* \cap \mathrm{pol}_E^*(q),$$

в силу которых частичная функция f вслед за предикатом q не сохраняет некоторый предикат из множества Y. Выберем такой не сохраняемый функцией f предикат q' в множестве Y с минимальным возможным числом m' переменных. Так как функция f не сохраняет предикат q', ему удовлетворяют некоторые наборы $Z_1, \ldots, Z_n$, в отличие от набора $Z_0 = f^{[m]}(Z_1, \ldots, Z_n)$, не удовлетворяющего ему. Поскольку набор Z_0 не содержит неопределённой компоненты, равной $*$, строки матрицы $Z = (Z_1^T, \ldots, Z_n^T)$ являются одновременно строками матрицы X. Более того, каждая строка матрицы X присутствует в матрице Z, иначе предикат q можно спроектировать по переменным, соответствующим тем строкам матрицы X, которые отсутствуют в матрице Z, и снова получить предикат, не сохраняемый функцией f и, следовательно, неинвариантный для клона B, что противоречит B-предельности предиката q по проектированию. Заметим также, что матрица Z не содержит повторяющихся строк (как и матрица X), так как в противном случае можно отождествить пару переменных у предиката q' и получить предикат с меньшим числом переменных, не сохраняемый функцией f и принадлежащий (вслед за предикатом q') и-описанию Y в силу расширенности последнего; это противоречит минимальности числа m'. Таким образом, $m' = m$ и для некоторой подстановки α на множестве чисел $1, \ldots, m$ i-я строка матрицы Z является $\alpha(i)$-й строкой матрицы X. Тогда $(Z_j)_\alpha = X_j$ для любого j, $0 \leqslant j \leqslant l$ (здесь для набора $z = (z_1, \ldots, z_m)$ через z_α обозначается набор $(z_{\alpha(1)}, \ldots, z_{\alpha(m)})$). Таким образом, для предиката q'_α, перестановочно эквивалентного предикату q' из множества Y, выполняется условие $q \subseteq q'_\alpha$, и набор $(Z_0)_\alpha = X_0$ не удовлетворяет предикату q'_α. Так что в качестве предиката q^{X_0} можно взять q'_α. Тем самым лемма доказана. ■

Следствием леммы 3 является следующая

Теорема 1. Пусть K и B — клоны функций из P_E, такие, что $K \subset B$, а множество Y предикатов из $\mathrm{inv}_E(K)$ является расширенным и-описанием клона K в клоне B. Тогда равносильны следующие условия:

1) клон K не является максимальным подклоном клона B;
2) включения $K \subset B \cap \mathrm{pol}_E(q) \subset B$ выполняются для некоторого предиката

$$q = q_0 \cap q_1 \cap \ldots \cap q_l,$$

где предикат q_0 инвариантен для клона B, а каждый из предикатов $q_0, \ldots, q_l$ перестановочно эквивалентен некоторому предикату из Y.

Доказательство. Подклон K, не являющийся максимальным в клоне B, можно расширить до клона $B \cap \mathrm{pol}_E(q)$, где q — некоторый B-предельный по проектированию и отождествлению предикат. Лемма 3 устанавливает для предиката q возможность представления и включений, указанных во втором пункте доказываемой теоремы. Таким образом из первого условия следует второе. Обратная импликация очевидна. Следствие доказано. ■

Лемма 3 и теорема 1 дают условия, при которых подклон, заданный своим расширенным и-описанием, является максимальным. Такие условия позволяют судить о максимальности подклона, исходя из свойств его инвариантных предикатов, и в некоторых случаях позволяет выделять максимальные подклоны. В связи с этим будем называть лемму 3 и теорему 1 *леммой о выделении* и *теоремой о выделении* соответственно. Проиллюстрируем возможности использования этих утверждений, а заодно и леммы о доопределении, следующим примером.

Пример 1. Предикаты

$$s_m(x_1, \ldots, x_{2m}) \equiv x_1 + \ldots + x_{2m} = 0 \,(\mathrm{mod}\, 2),\ m \geqslant 1,$$

инвариантны для клона L_2 линейных булевых функций. Такие предикаты составляют его и-описание в клоне P_2 всех булевых функций. Это следует в силу леммы 1 о доопределении из того, что всякая не полностью определённая булева функция f от n переменных, сохраняющая эти предикаты, допускает дальнейшее доопределение

$$f(x) = f(x_1) + \ldots + f(x_{2m-1}) \bmod 2$$

на любом наборе x из множества $\{0,1\}^n$, являющемся для некоторого $m \geqslant 1$ покомпонентной суммой $x = x_1 + \ldots + x_{2m-1} \bmod 2$ каких-то $2m-1$ не обязательно различных наборов $x_1, \ldots, x_{2m-1}$ из области определения функции f. Функция f допускает дальнейшее доопределение произвольным значением из множества $\{0,1\}$ на любом другом наборе x, не являющемся суммой наборов, выбранных из области определения в нечётном количестве. Непосредственно проверяется, что в результате подобного доопределения получается функция, также сохраняющая *предикаты* s_m *при* $m \geqslant 1$, которые, следовательно, *составляют и-описание клона* L_2, причём расширенное, как несложно понять.

На основании теоремы 1 о выделении теперь легко проверить максимальность подклона L_2 в клоне P_2 булевых функций. Для этого нужно рассмотреть предикат $q = q_0 \cap s_m$, где q_0 — инвариантный для P_2 предикат, то есть диагональ. В результате такого пересечения после удаления фиктивных переменных получается предикат $s_{m'}$, где $1 \leqslant m' \leqslant m$, в силу чего строгие включения $K \subset B \cap \mathrm{pol}_E(q) \subset B$ не могут выполняться одновременно. Отсюда, на основании теоремы 1, клон L_2 — максимальный в P_2.

B-простые предикаты. Отметим некоторые частные случаи, когда на основании леммы о выделении удаётся легко судить о максимальности подклонов.

B-предельный по проектированию, отождествлению, симметиризации и сужению предикат назовём *B-простым*, если он один составляет и-описание некоторого подклона в клоне B, очевидно, расширенное и-описание. Иначе, B-простой предикат можно определить как B-предельный по симметиризации и сужению, составляющий расширенное и-описание некоторого подклона в клоне B (в этом случае B-предельность по отождествлению следует из расширенности и-описания, а B-предельность по проектированию следует из леммы 3).

Лемма 4. Если предикат p — B-простой, а предикат q — B-предельный по проектированию и отождествлению, то строгое включение $B \cap \mathrm{pol}_E(p) \subset B \cap \mathrm{pol}_E(q)$ невозможно, а равенство $B \cap \mathrm{pol}_E(p) = B \cap \mathrm{pol}_E(q)$ выполняется тогда и только тогда, когда предикаты p и q перестановочно эквивалентны.

Доказательство. Пусть выполняется включение $B \cap \mathrm{pol}_E(p) \subseteq B \cap \mathrm{pol}_E(q)$. Тогда по лемме 3 предикат q можно представить в виде $q = q_0 \cap \ldots \cap q_l$, где предикат q_0 принадлежит множеству $\mathrm{inv}_E(B)$, а предикаты $q_1, \ldots, q_l$ перестановочно эквивалентны предикату p и тогда попарно перестановочно эквивалентны между собой. Пересечение любых двух предикатов q_i и q_j, $1 \leqslant i < j \leqslant l$, является тогда симметризацией каждого из них, а в силу их B-предельности по симметризации (это свойство переносится на них от перестановочно эквивалентного им предиката p) либо совпадает с некоторым из них (и тогда с каждым), либо инвариантно для клона B (что невозможно). В силу этого можно ограничиться рассмотрением случая $l = 1$. Тогда предикат $q = q_0 \cap q_1$ является B-сужением предиката q_1, B-предельного по сужению вслед за перестановочно эквивалентным ему предикатом p. Отсюда, с учётом неинвариантности предиката q для клона B, предикат q совпадает с предикатом q_1. Тогда предикаты q и p перестановочно эквивалентны, и выполняется равенство $B \cap \mathrm{pol}_E(p) = B \cap \mathrm{pol}_E(q)$. ■

Непосредственным следствием леммы 4 является

Теорема 2. Пусть предикат p является B-простым. Тогда подклон $B \cap \mathrm{pol}_E(p)$ является максимальным в B. В этом случае также для любого B-предельного предиката q равенство $B \cap \mathrm{pol}_E(p) = B\,\mathrm{pol}_E(q)$ означает перестановочную эквивалентность предикатов p и q и, в частности, B-простоту предиката q.

Часть клонов, максимальных в P_E в силу теорем Э. Поста и Розенберга, описываются P_E-простыми предикатами. Это видно из рассматриваемых ниже примеров.

Пример 2. *Предикат $\neq$ составляет и-описание клона самодвойственных булевых функций* (и тогда, очевидно, составляет расширенное и-описание), так как частичную булеву функцию, сохраняющую этот предикат, можно доопределить до самодвойственной булевой функции противоположными значениями на противоположных наборах. Непосредственно проверяются различные свойства P_2-предельности этого предиката (где P_2 — клон всех булевых функций). Таким образом, *предикат $\neq$ является P_2-простым.*

Пример 3. Пусть $\preccurlyeq$ — решёточное упорядочение множества E.

Всякую частичную фукцию f, сохраняющую предикат $\preccurlyeq$, можно доопределить до функции F из клона $\mathrm{pol}_E(\preccurlyeq)$, положив

$$F(x) = \vee f(x'),$$

где точная верхняя грань $\vee$ вычисляется в решётке E по всем наборам x' из области определения функции f, таким, что $x' \preccurlyeq x$. В силу леммы о доопределении *предикат $\preccurlyeq$ один составляет и-описание клона* $\mathrm{pol}_E(\preccurlyeq)$ *функций из P_E, монотонных относительно решёточного упорядочения $\preccurlyeq$.*

Несложно проверить свойства P_E-предельности предиката $\preccurlyeq$. Действительно, как видно, P_E-сужение предиката $\preccurlyeq$ (то есть его пересечение с диагональю) либо совпадает с ним, либо является диагональю. А в результате отождествления переменных, проектирования и симметризации с нетождественной подстановкой из предиката $\preccurlyeq$ получаются только диагонали.

В силу сказанного, *предикат $\preccurlyeq$ является P_E-простым.*

Пример 4. Пусть p — центральный вполне рефлексивный симметричный предикат из Π_E, зависящий от $m \geqslant 1$ переменных и отличающийся от полной диагонали E^m. Напомним, что *центральность* предиката p означает существование для него *центрального элемента* c в множестве E, такого, что предикату p удовлетворяет всякий набор из E^m, имеющий компоненту, равную c. *Полная рефлексивность* означает, что предикату удовлетворяет всякий набор из E^m, имеющий равные компоненты. *Симметричность* означает, что предикат совпадает с любым перестановочно эквивалентным ему предикатом.

Непосредственно проверяется, что частичную функцию из P_E^*, сохраняющую предикат p, можно доопределить до функции из клона $\mathrm{pol}_E(p)$ значением c. В силу леммы о доопределении *предикат p составляет и-описание* указанного клона. Это и-описание расширенное, поскольку в результате отождествления любых двух переменных из предиката p получается полная диагональ, инвариантная для P_E.

Далее, *предикат p является P_E-предельным по сужению и симметризации.* Действительно, P_E-сужение (то есть пересечение с диагональю) предиката p либо совпадает с ним (в случае полной диагонали), либо является диагональю (в остальных случаях — в силу полной рефлексивности предиката p) и тогда инвариантно для P_E. При симметризации предикат не изменяется (в силу симметричности).

Таким образом, *центральный, вполне рефлексивный и симметричный предикат p из Π_E является P_E-простым.*

Пример 5. Тривиальный пример P_E-простого предиката даёт для любого элемента c из множества E одноместный предикат $x_1 = c$, очевидно, центральный, симметричный и вполне рефлексивный.

Пример 6. Пусть теперь $\leqslant$ — полурешёточное упорядочение множества E. Точнее, множество E, упорядоченное отношением $\leqslant$, является верхней полурешёткой, но не решёткой [6]. Иными словами, в множестве E любые два элемента a и b обладают точной верхней гранью $a + b$, а точная нижняя грань $a \cdot b$ существует не для любых элементов a и b. Функции из P_E, сохраняющие упорядочение $\leqslant$, составляют клон $M_E = \mathrm{pol}_E(\leqslant)$ *монотонных функций на полурешётке E*. Функция f из P_E, имеющая монотонную миноранту g, принадлежащую клону M_L, такую, что $g(x) \leqslant f(x)$ для любого набора x из множества E^n, где n — число переменных функций f и g, называется *квазимонотонной на полурешётке E*. Квазимонотонные и монотонные функции на полурешётке введены в [7] для описания асинхронных дискретных управляющих систем. Основные классы квазимонотонных функций изучались также в [8]. Проблема полноты в классе квазимонотонных функций рассматривалась в [9, 10].

В соответствии с тестом квазимонотонности из [7] квазимонотонные функции составляют клон Q_E сохранения предикатов

$$\varepsilon^{(n)}(x_1, \ldots, x_n) \equiv \exists x(x \leqslant x_1 \wedge \ldots \wedge x \leqslant x_n)$$

и даже клон сохранения единственного такого предиката при $n = q(E)$, где $q(E)$ — максимальная мощность минимального по включению множества элементов из E без общей нижней грани.

В действительности, *предикаты $\varepsilon^{(n)}$, и даже единственный из них при $n = q(E)$, составляют (составляет) и-описание клона Q_E*. Это следует из леммы 1 о доопределении, так как частичную функцию из P_E^*, сохраняющую предикат $\varepsilon^{(q(L))}$, можно доопределить наибольшим значением полурешётки E до квазимонотонной функции из Q_E, сохраняющей его же. Вместе с тем *и-описание клона M_E монотонных функций*

составляют предикаты $\leqslant$ *и* $\varepsilon^{(q(E))}$, поскольку сохраняющую их частичную функцию f из P_E^* можно доопределить до монотонной функции F из M_E, положив

$$F(x) = \prod f(x'),$$

где произведение вычисляется в полурешётке E по всем наборам x' из области определения функции f, таким, что $x \leqslant x'$. Из сказанного следует также, что *предикат* $\leqslant$ *сотавляет и-описание клона* M_E *в клоне* Q_E. Это и-описание — расширенное, так как из предиката $\leqslant$ проектированием и отождествлением переменных получается только полная одноместная диагональ. Заметим, что предикат $\leqslant$ обладает свойствами Q_E-предельности по сужению и симметризации. В самом деле, в силу найденного выше и-описания клона Q_E Q_E-сужением предиката $\leqslant$ может быть только предикат

$$d(x_1, x_2) \wedge \varepsilon^{(2)}(y_1, y_2) \wedge x_1 \leqslant x_2,$$

где d — диагональ и $\{y_1, y_2\} \subseteq \{x_1, x_2\}$. Всякий такой предикат либо совпадает с предикатом $\leqslant$, либо является диагональю. В результате симметризации предиката $\leqslant$ получается либо он же, либо предикат равенства. В силу сказанного, *предикат* $\leqslant$ *является* Q_E*-простым*, а описываемый им *клон* M_E *является максимальным в* Q_E.

ЛИТЕРАТУРА

1. *Мальцев А. И.* Итеративные алгебры и многообразия Поста // Алгебра и логика. 1966. Т. 5. № 2. С. 5–24.
2. *Мальцев А. И.* Итеративные алгебры Поста. Новосибирск: Изд-во НГУ, 1976.
3. *Geiger D.* Closed systems of functions and predicates // Pacific journal of mathematics. 1968. V. 27. P. 95–100.
4. *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.
5. *Ромов Б. А.* О продолжении не всюду определённых функций // Кибернетика. 1987. № 3. С. 27–34.
6. *Курош А. Г.* Лекции по общей алгебре. СПб.: Лань, 2005.
7. *Агибалов Г. П.* Дискретные автоматы на полурешётках. Томск: Изд-во Том. ун-та, 1993.
8. *Парватов Н. Г.* Об инвариантах некоторых классов квазимонотонных функций на полурешётке // Прикладная дискретная математика. 2009. № 4. С. 21–28.
9. *Парватов Н. Г.* Функциональная полнота в замкнутых классах квазимонотонных и монотонных трёхзначных функций на полурешётке // Дискрет. анализ и исслед. операций. Сер. 1. 2003. Т. 10. № 1. С. 61–78.
10. *Парватов Н. Г.* Теорема о функциональной полноте в классе квазимонотонных функций на конечной полурешётке // Дискрет. анализ и исслед. операций. Сер. 1. 2006. Т. 13. № 3. С. 62–82.

2011 Теоретические основы прикладной дискретной математики №1(11)

DOI 10.17223/20710410/11/3

УДК 519.7

О ЧИСЛЕ СОВЕРШЕННО УРАВНОВЕШЕННЫХ БУЛЕВЫХ ФУНКЦИЙ С БАРЬЕРОМ ДЛИНЫ 3[1]

С. В. Смышляев

Московский государственный университет им. М. В. Ломоносова, г. Москва, Россия

E-mail: smyshsv@gmail.com

Рассматривается класс булевых функций с барьером длины 3, вложенный в множество совершенно уравновешенных булевых функций. Получены нижняя и верхняя оценки для мощности класса булевых функций с правым барьером длины 3, существенно зависящих от последней переменной, а также новая нижняя оценка логарифма числа совершенно уравновешенных булевых функций n переменных, существенно и нелинейно зависящих от крайних переменных: $2^{n-2}\left(1+\frac{\log_2 5}{4}-\mathrm{O}(1/\sqrt{n})\right)$.

Ключевые слова: *совершенно уравновешенные функции, барьеры булевых функций, криптография.*

Введение

Серьезные продвижения в исследовании свойств совершенно уравновешенных булевых функций были получены в работах [1, 2]. В частности, в них доказан критерий совершенной уравновешенности, связывающий это свойство со свойствами отсутствия запрета и отсутствия потери информации. Кроме того, в работе [2] впервые был приведен пример совершенно уравновешенной булевой функции, не являющейся линейной по первой (или последней) переменной. Ряд результатов о совершенно уравновешенных булевых функциях был получен в работах [3 – 7], в частности в [6] было выделено достаточное условие совершенной уравновешенности — наличие у булевой функции барьера. Свойства функций с барьером изучались позже в работах [8 – 10]; методы построения классов совершенно уравновешенных булевых функций без барьера рассматривались в [11, 12].

Одним из предложенных в работе [6] подходов к исследованию класса совершенно уравновешенных булевых функций является последовательное изучение множеств функций с барьерами длины $1, 2, 3, \ldots$ Множества функций с барьерами длины 1 и 2 описываются тривиальным образом и не представляют существенного интереса.

Настоящая работа посвящена получению мощностных оценок для класса функций с барьером длины 3. Производится модификация полученного в [6] критерия принадлежности произвольной булевой функции, существенно зависящей от последней переменной, данному классу. С помощью выделения независимого множества вершин большой мощности в графе де Брейна и выбора определенного класса разметок вершин данного независимого множества удается получить широкий класс функций с правым барьером длины 3. Кроме того, небольшая модификация этого построения приводит к получению нижней оценки мощности множества совершенно уравновешенных функций, существенно и нелинейно зависящих от крайних переменных, — множества, для

[1]Работа поддержана РФФИ (номер проекта 09-01-00653-а).

мощности которого ранее не было известно никаких оценок, кроме тривиальных. Вводится понятие правильной тройки разметок подграфа графа де Брейна, соответствующее набору необходимых условий, которым удовлетворяет всякая булева функция с правым барьером длины 3, существенно зависящая от последней переменной. С помощью построения подграфа специального вида и оценивания числа правильных троек его разметок получается требуемая верхняя мощностная оценка.

1. Основные определения и обозначения

Для множества двоичных наборов длины n будем использовать обозначение $V_n = \{0,1\}^n$. Через $\mathcal{F}_n$ будем обозначать множество булевых функций от n переменных, через Φ_n — множество функций из $\mathcal{F}_n$, существенно зависящих от первой и последней переменной.

Для всякой функции $f \in \mathcal{F}_n$ через $f_{(0)}, f_{(1)} \in \mathcal{F}_{n-1}$ будем обозначать функции, определяемые следующим равенством:

$$f(x_1, x_2, \ldots, x_n) = f_{(0)}(x_1, x_2, \ldots, x_{n-1}) \oplus x_n f_{(1)}(x_1, x_2, \ldots, x_{n-1}).$$

Аналогично,

$$\begin{aligned}
&f_{(0)}(x_1, x_2, \ldots, x_{n-1}) = f_{(00)}(x_1, x_2, \ldots, x_{n-2}) \oplus x_{n-1} f_{(01)}(x_1, x_2, \ldots, x_{n-2});\\
&f_{(1)}(x_1, x_2, \ldots, x_{n-1}) = f_{(10)}(x_1, x_2, \ldots, x_{n-2}) \oplus x_{n-1} f_{(11)}(x_1, x_2, \ldots, x_{n-2});\\
&f_{(00)}(x_1, x_2, \ldots, x_{n-2}) = f_{(000)}(x_1, x_2, \ldots, x_{n-3}) \oplus x_{n-2} f_{(001)}(x_1, x_2, \ldots, x_{n-3});\\
&f_{(01)}(x_1, x_2, \ldots, x_{n-2}) = f_{(010)}(x_1, x_2, \ldots, x_{n-3}) \oplus x_{n-2} f_{(011)}(x_1, x_2, \ldots, x_{n-3}).
\end{aligned}$$

Пусть $n, m \in \mathbb{N}$, $f \in \mathcal{F}_n$. Обозначим для $f \in \mathcal{F}_n$ через f_m следующее отображение из V_{m+n-1} в V_m:

$$f_m(x_1, x_2, \ldots, x_{m+n-1}) = (f(x_1, \ldots, x_n), f(x_2, \ldots, x_{n+1}), \ldots, f(x_m, \ldots, x_{m+n-1})). \qquad (1)$$

Определение 1 [6]. Булева функция $f \in \mathcal{F}_n$ называется совершенно уравновешенной, если соотношение

$$\left|f_m^{-1}(\mathbf{y})\right| = 2^{n-1}$$

выполняется для любого $m \in \mathbb{N}$ и любого $\mathbf{y} \in V_m$. Множество совершенно уравновешенных функций из $\mathcal{F}_n$ обозначим через $\mathcal{PB}_n$.

Понятия, эквивалентные совершенной уравновешенности булевых функций, рассматривались и широко изучались в работах [1] (сюръективные эндоморфизмы символических динамических систем) и [2] (сильно равновероятные булевы функции).

Определение 2 [6]. Булева функция $f \in \mathcal{F}_n$ называется функцией с правым барьером длины b, $b \in \mathbb{N}$, если система уравнений

$$\begin{cases} f_{b'}(x_1, x_2, \ldots, x_{b'+n-1}) = f_{b'}(z_1, z_2, \ldots, z_{b'+n-1}),\\ x_1 = z_1, \ldots, x_{n-1} = z_{n-1}, x_n = 0, z_n = 1 \end{cases}$$

имеет решение при всяком $b' \in \mathbb{N}$, таком, что $b' \leqslant b-1$, а система уравнений

$$\begin{cases} f_b(x_1, x_2, \ldots, x_{b+n-1}) = f_b(z_1, z_2, \ldots, z_{b+n-1}),\\ x_1 = z_1, \ldots, x_{n-1} = z_{n-1}, x_n = 0, z_n = 1 \end{cases}$$

решений не имеет.

Булева функция $f \in \mathcal{F}_n$ называется функцией с левым барьером длины b, если $f'(x_1, \ldots, x_n) \equiv f(x_n, \ldots, x_1)$ является функцией с правым барьером длины b.

Булева функция $f \in \mathcal{F}_n$ имеет барьер, если она имеет правый или левый барьер, или оба сразу. При этом длиной барьера функции называется соответственно длина правого барьера, левого барьера или меньшая из длин барьеров.

Замечание 1. Нетрудно заметить, что наличие правого (левого) барьера длины 1 означает линейность функции по последнему (первому) аргументу. Заметим также, что для всяких $n, b \in \mathbb{N}$, $b \leqslant n$, верно, что любая функция из $\mathcal{F}_n$, линейно зависящая от x_{n-b+1} (линейно зависящая от x_b) и не зависящая от переменных x_{n-b+2}, x_{n-b+3}, $\ldots, x_n$ (не зависящая от переменных x_1, x_2, $\ldots$, x_{b-1}), имеет правый (левый) барьер длины b.

Замечание 2. Для всех утверждений, в которых упоминается длина правого барьера некоторых функций, могут быть очевидным образом построены аналоги с использованием понятия левого барьера. Ввиду этого далее будем говорить только о правых барьерах функций.

2. Предварительные результаты

Теорема 1 [6]. Наличие барьера у булевой функции является достаточным условием совершенной уравновешенности функции.

Замечание 3. В работе [6] было установлено, что наличие барьера не является необходимым условием совершенной уравновешенности. Позже в работах [5, 8, 12] был предложен ряд методов построения совершенно уравновешенных булевых функций без барьера.

Теорема 2 [6]. Функция $f \in \mathcal{F}_n$, такая, что $f_{(1)} \not\equiv 0$, имеет правый барьер длины 3 тогда и только тогда, когда для любых $x_1, x_2, \ldots, x_{n-1}$ выполнены следующие условия:

1) $f_{(11)}(x_1, \ldots, x_{n-2}) = 0$;
2) если $f_{(10)}(x_1, \ldots, x_{n-2}) = 1$, то

$$\begin{aligned}
&f_{(10)}(x_2, \ldots, x_{n-2}, 0) = f_{(10)}(x_2, \ldots, x_{n-2}, 1) = 0,\\
&f_{(10)}(0, x_1 \ldots, x_{n-3}) = f_{(10)}(1, x_1, \ldots, x_{n-3}) = 0,\\
&f_{(011)}(x_2, \ldots, x_{n-2}) = 0,\\
&f_{(001)}(x_2, \ldots, x_{n-2}) \oplus f_{(01)}(x_1, \ldots, x_{n-2}) f_{(01)}(x_2, \ldots, x_{n-2}, 0) = 1;
\end{aligned}$$

3) если $f_{(10)}(x_1, \ldots, x_{n-2}) = f_{(10)}(x_2, \ldots, x_{n-1}) = 0$, то

$$f_{(01)}(x_2, \ldots, x_{n-1}) = 1.$$

Через GB_m будем обозначать граф де Брейна порядка m: ориентированный граф на 2^m вершинах, поставленных в соответствие элементам множества V_m и соединенных дугами так, что дуга из вершины, соответствующей набору $(a_1, a_2, \ldots, a_m) \in V_m$, в вершину, соответствующую набору $(b_1, b_2, \ldots, b_m) \in V_m$, присутствует в графе GB_m в том и только в том случае, когда $(a_2, a_3, \ldots, a_m) = (b_1, b_2, \ldots, b_{m-1})$ (см. [13]). Обозначим через GB_m^* неориентированный граф на тех же вершинах, что и граф GB_m, получаемый из него заменой всех дуг на (неориентированные) ребра и удалением петель.

Обозначим через ω отображение из V_m в множество вершин графа GB_m и графа GB_m^*, переводящее двоичные наборы в соответствующие им вершины. Через Ω будем обозначать аналогично определяемое отображение из множества всех подмножеств V_m в множество всех подмножеств вершин графов GB_m и GB_m^*.

Теорема 3 [14]. В графе GB^*_m существует независимое множество вершин (т. е. множество вершин, никакие две из которых не соединены ребром), не содержащее $\omega(0,0,\ldots,0)$ и $\omega(1,1,\ldots,1)$ и имеющее следующую мощность:

$$\begin{cases} 2^{m-1} - \left(\dfrac{1}{2}\dbinom{m}{m/2} - \dbinom{m-2}{m/2-2}\right), & \text{если } m \text{ четно;} \\ 2^{m-1} - \left(\dbinom{m-1}{(m-1)/2} - 2\dbinom{m-3}{(m-1)/2-2}\right), & \text{если } m \text{ нечетно.} \end{cases}$$

3. Основные результаты

Обозначим для всяких $n, b \in \mathbb{N}$ через $W_{b,n}$ множество функций из $\mathcal{F}_n$ с правым барьером длины b, существенно зависящих от x_n.

Из результатов работы [6] вытекает, что множество $W_{2,n}$ пусто при всяком n. С учетом этого нетрудно установить, что все функции с правым барьером длины 3, не принадлежащие $W_{3,n}$, не зависят существенно от x_{n-1} и x_n и линейны по x_{n-2}. Таким образом, учитывая замечания 1 и 2, при исследовании функций с барьером длины 3 достаточно ограничиться изучением множества $W_{3,n}$.

Перепишем условия, сформулированные в теореме 2, выделив отдельно свойства каждой из функций $f_{(11)}, f_{(10)}, f_{(01)}, f_{(00)}$. Получим: $f \in W_{3,n}$ тогда и только тогда, когда для всяких $x_1, x_2, \ldots, x_{n-1}$ выполняются следующие условия:

1) $f_{(11)}(x_1,\ldots,x_{n-2}) = 0$;
2) $f_{(10)}(x_1,\ldots,x_{n-2}) f_{(10)}(x_2,\ldots,x_{n-1}) = 0$;
3) если $f_{(10)}(x_1,\ldots,x_{n-2}) = f_{(10)}(x_2,\ldots,x_{n-1}) = 0$, то $f_{(01)}(x_2,\ldots,x_{n-1}) = 1$;
   если $f_{(10)}(x_1,\ldots,x_{n-2}) = 1$, то $f_{(01)}(x_2,\ldots,x_{n-2},0) = f_{(01)}(x_2,\ldots,x_{n-2},1)$;
   если $f_{(10)}(0,x_2,\ldots,x_{n-2}) = f_{(10)}(1,x_2,\ldots,x_{n-2}) = 1$ и $f_{(01)}(x_2,\ldots,x_{n-1}) = 1$, то $f_{(01)}(0,x_2,\ldots,x_{n-2}) = f_{(01)}(1,x_2,\ldots,x_{n-2})$;
4) если $f_{(10)}(x_1,\ldots,x_{n-2}) = 1$, то $f_{(00)}(x_2,\ldots,x_{n-2},0) \oplus f_{(00)}(x_2,\ldots,x_{n-2},1) = f_{(01)}(x_1,\ldots,x_{n-2}) f_{(01)}(x_2,\ldots,x_{n-1}) \oplus 1$.

Лемма 1. Пусть S — независимое множество вершин графа GB^*_{n-2}, не содержащее вершин $\omega(0,0,\ldots,0)$ и $\omega(1,1,\ldots,1)$. Тогда любая функция $f_{(10)}$, равная нулю на всех наборах из $V_{n-2} \setminus \Omega^{-1}(S)$, удовлетворяет условию 2.

Доказательство. Так как S является независимым множеством вершин графа GB^*_{n-2} и не содержит $\omega(0,0,\ldots,0)$ и $\omega(1,1,\ldots,1)$, то в графе GB_{n-2} никакая дуга не соединяет две вершины из S. Следовательно, при указанном выборе функции $f_{(10)}$ не существует ни одного набора $(x_1,\ldots,x_{n-1}) \in V_{n-1}$, такого, что $f_{(10)}(x_1,\ldots,x_{n-2}) = f_{(10)}(x_2,\ldots,x_{n-1}) = 1$. Таким образом, для функции $f_{(10)}$ выполняется условие 2. ■

Лемма 2. В графе GB^*_m существует не содержащее $\omega(0,0,\ldots,0)$ и $\omega(1,1,\ldots,1)$ независимое множество вершин S мощности $2^{m-1} - \mathrm{O}\left(2^m/\sqrt{m}\right)$, такое, что для любых $x_2,\ldots,x_m$ вершины $\omega(0,x_2,\ldots,x_m)$ и $\omega(1,x_2,\ldots,x_m)$ входят или не входят в S одновременно.

Доказательство. Представляя (при четном m) разность $\left(\dfrac{1}{2}\dbinom{m}{m/2} - \dbinom{m-2}{m/2-2}\right)$ в виде $\dbinom{m-2}{m/2-1}$ и применяя формулу Стирлинга, легко показать, что из теоремы 3 следует существование множества $T' \subseteq V_{m-1} \setminus \{(0,0,\ldots,0),(1,1,\ldots,1)\}$ мощности $2^{m-2} - \mathrm{O}(2^{m-1}/\sqrt{m-1})$, такого, что $S' = \Omega(T')$ — независимое множество вершин графа GB^*_{m-1}.

Положим $T = \{(x_1, x_2, \ldots, x_m) \in V_m : (x_2, x_3, \ldots, x_m) \in T'\}$, $S = \Omega(T)$. Очевидно, что $|S| = |T| = 2^{m-1} - \mathrm{O}\,(2^m/\sqrt{m})$ и $\omega(0, 0, \ldots, 0) \notin S$, $\omega(1, 1, \ldots, 1) \notin S$. Покажем, что вершины из S образуют независимое множество в графе GB^*_m. Заметим, что две вершины $\omega(x'_1, x'_2, \ldots, x'_m), \omega(x''_1, x''_2, \ldots, x''_m)$ соединены ребром в GB^*_m тогда и только тогда, когда выполнено условие $(x'_1, x'_2, \ldots, x'_{m-1}) = (x''_2, x''_3, \ldots, x''_m)$ либо условие $(x'_2, x'_3, \ldots, x'_m) = (x''_1, x''_2, \ldots, x''_{m-1})$. Таким образом, легко видеть, что если множество вершин S не является независимым в графе GB^*_m, то и множество S' не является независимым в GB^*_{m-1}. ■

Используя приведенные утверждения, докажем следующую нижнюю оценку.

Теорема 4. $\log_2 |W_{3,n}| \geqslant 2^{n-2}\left(1 + \dfrac{\log_2 5}{4} - \mathrm{O}(1/\sqrt{n})\right)$.

Доказательство. Зафиксируем произвольную функцию $f_{(10)} \in \mathcal{F}_{n-2}$, $f_{(10)} \not\equiv 0$, удовлетворяющую условию 2. Очевидно, что произвольная функция $f_{(01)} \in \mathcal{F}_{n-2}$, удовлетворяющая при всяких $x_1, x_2, \ldots, x_{n-1}$ условию

3′) если $f_{(10)}(x_1, \ldots, x_{n-2}) = f_{(10)}(x_2, \ldots, x_{n-1}) = 0$, то $f_{(01)}(x_2, \ldots, x_{n-1}) = 1$;
если $f_{(10)}(0, x_2, \ldots, x_{n-2}) \oplus f_{(10)}(1, x_2, \ldots, x_{n-2}) = 1$, то $f_{(01)}(x_2, \ldots, x_{n-2}, 0) = f_{(01)}(x_2, \ldots, x_{n-2}, 1) = 1$;
если $f_{(10)}(0, x_2, \ldots, x_{n-2}) = f_{(10)}(1, x_2, \ldots, x_{n-2}) = 1$, то $f_{(01)}(x_2, \ldots, x_{n-2}, 0) = f_{(01)}(x_2, \ldots, x_{n-2}, 1) = 0$,

удовлетворяет также и условию 3. Оценим число функций $f_{(01)}, f_{(00)}$, удовлетворяющих условиям 3′ и 4.

Нетрудно проверить, что при любой фиксированной удовлетворяющей 2 функции $f_{(10)}$ условие 3′ однозначно определяет значение $f_{(01)}$ на тех и только тех наборах, на которых $f_{(10)}$ обращается в нуль. Таким образом, удовлетворяющих 3′ функций $f_{(01)}$ в точности $2^{\mathrm{wt}(f_{(10)})}$.

При всяких удовлетворяющих условиям 2 и 3′ функциях $f_{(10)}$ и $f_{(01)}$ условие 4 не накладывает ограничений на выбор функции $f_{(000)}$ и определяет значение функции $f_{(001)}$ на наборе $(x_1, x_2, \ldots, x_{n-3})$ тогда и только тогда, когда $f_{(10)}(0, x_1, \ldots, x_{n-3}) = 1$ или $f_{(10)}(1, x_1, \ldots, x_{n-3}) = 1$. Таким образом, удовлетворяющих 4 функций $f_{(00)}$ в точности $2^{2^{n-2} - \mathrm{wt}(f_{(10)}(0,x_1,\ldots,x_{n-3}) \vee f_{(10)}(1,x_1,\ldots,x_{n-3}))}$.

С учетом равенства $\mathrm{wt}(f_{(10)}) - \mathrm{wt}(f_{(10)}(0, x_1, \ldots, x_{n-3}) \vee f_{(10)}(1, x_1, \ldots, x_{n-3})) = \mathrm{wt}(f_{(10)}(0, x_1, \ldots, x_{n-3}) f_{(10)}(1, x_1, \ldots, x_{n-3}))$ получаем: при всякой удовлетворяющей 2 функции $f_{(10)}$ не менее $2^{2^{n-2} + \mathrm{wt}(f_{(10)}(0,x_1,\ldots,x_{n-3}) f_{(10)}(1,x_1,\ldots,x_{n-3}))}$ пар функций $(f_{(01)}, f_{(00)})$ удовлетворяют условиям 3 и 4.

Пусть S — независимое множество вершин графа GB^*_{n-2}, определяемое леммой 2. Рассмотрим все возможные отличные от тождественного нуля функции $f_{(10)}$, определенные в соответствии с леммой 1. С учетом полученных выше результатов имеем

$$|W_{3,n}| \geqslant \sum_{\substack{f_{(10)} \in \mathcal{F}_{n-2} \setminus \{0\}, \\ f_{(10)}(\mathbf{x}) = 0, \omega(\mathbf{x}) \notin S}} 2^{2^{n-2} + \mathrm{wt}(f_{(10)}(0,x_1,\ldots,x_{n-3}) f_{(10)}(1,x_1,\ldots,x_{n-3}))} =$$

$$= 2^{2^{n-2}} \sum_{\substack{g\colon \Omega^{-1}(S') \mapsto V_2, \\ g \not\equiv (0,0)}} 2^{|g^{-1}(1,1)|} = 2^{2^{n-2}} \left(\sum_{i=0}^{|S'|} \left[\binom{|S'|}{i} 3^{|S'|-i} \cdot 2^i \right] - 1 \right) = \tag{2}$$

$$= 2^{2^{n-2}} \left(5^{|S'|} - 1\right) = 2^{2^{n-2} - \log_2 \left(\frac{5^{|S'|}}{5^{|S'|}-1} \right)} \cdot 5^{|S'|} = 2^{2^{n-2} - \mathrm{o}(1)} \cdot 2^{\log_2 5 \cdot \left(2^{n-4} - \mathrm{O}\left(\frac{2^{n-4}}{\sqrt{n-4}}\right)\right)} =$$

$$= 2^{2^{n-2}\left(1 + \frac{1}{4}\log_2 5 - \mathrm{O}\left(\frac{1}{4\sqrt{n-4}}\right)\right) - \mathrm{o}(1)}.$$

Логарифмируя получившееся неравенство, получаем

$$\log_2 |W_{3,n}| \geqslant 2^{n-2}\left(1+\frac{\log_2 5}{4}-\mathrm{O}(1/\sqrt{n})\right). \blacksquare$$

Через $\mathcal{L}_n^{\mathcal{L}}$ ($\mathcal{L}_n^{\mathcal{R}}$) обозначим множество функций n переменных, линейно зависящих от первой (последней) переменной. Как следует из результатов работ [3, 4, 6], наибольший интерес среди элементов $\mathcal{PB}_n$ представляют функции из $(\mathcal{PB}_n \cap \Phi_n) \setminus (\mathcal{L}_n^{\mathcal{L}} \cup \mathcal{L}_n^{\mathcal{R}})$.

Для получения нижней оценки мощности данного множества вернемся к доказательству теоремы 4. Требуя дополнительно от функции $f_{(000)}$ нелинейной существенной зависимости от x_1, получим, что при этом при всяких удовлетворяющих условиям 2 и 3′ функциях $f_{(10)}$ и $f_{(01)}$ число удовлетворяющих условию 4 функций $f_{(00)}$ в точности равно $\left(2^{2^{n-2}}-2^{2^{n-3}+2^{n-4}+1}\right)2^{-\mathrm{wt}(f_{(10)}(0,x_1,\ldots,x_{n-3})\vee f_{(10)}(1,x_1,\ldots,x_{n-3}))}$. Пользуясь цепочкой неравенств, аналогичной (2), и логарифмируя, приходим к следующему утверждению.

Теорема 5. $\log_2 \left|(\mathcal{PB}_n \cap \Phi_n) \setminus (\mathcal{L}_n^{\mathcal{L}} \cup \mathcal{L}_n^{\mathcal{R}})\right| \geqslant 2^{n-2}\left(1+\frac{\log_2 5}{4}-\mathrm{O}(1/\sqrt{n})\right)$.

Для получения верхней оценки мощности $W_{3,n}$ введем следующее понятие.

Определение 3. Пусть H — некоторый подграф графа GB_m. Будем называть тройку $(\varphi_{(10)}, \varphi_{(01)}, \varphi_{(00)})$ разметок вершин графа H элементами множества $\{0,1\}$ правильной, если для любых трех вершин v_1, v_2, v_3 графа H выполняются следующие условия:

1) если в H есть дуга из v_1 в v_2, то $\varphi_{(10)}(v_1) = 0$ или $\varphi_{(10)}(v_2) = 0$, причем если $\varphi_{(10)}(v_1) = \varphi_{(10)}(v_2) = 0$, то $\varphi_{(01)}(v_2) = 1$;
2) если в H из v_1 в вершины v_2 и v_3 ведут дуги и $v_2 \neq v_3$, то если $\varphi_{(10)}(v_1) = 1$, то $\varphi_{(01)}(v_2) = \varphi_{(01)}(v_3)$ и $\varphi_{(00)}(v_2) \oplus \varphi_{(00)}(v_3) = \varphi_{(01)}(v_1)\varphi_{(01)}(v_2) \oplus 1$.

Непосредственно из определения 3 и теоремы 2 вытекает следующее утверждение.

Утверждение 1. Если $f \in W_{3,n}$, то тройка $(f_{(10)} * \omega^{-1}, f_{(01)} * \omega^{-1}, f_{(00)} * \omega^{-1})$ разметок вершин GB_{n-2} является правильной относительно любого подграфа GB_{n-2}, содержащего все вершины GB_{n-2}.

Опишем для всех m подграфы H_m графа де Брейна GB_m, для которых далее получим верхние оценки числа правильных троек разметок. Удалим из графа GB_m вершину $\omega(0,0,\ldots,0)$, затем выделим в получившемся подграфе остовное дерево, представляющее собой полное двоичное дерево высоты $m-1$, на i-м уровне которого $(i = 0,1,\ldots,m-1)$ находятся все вершины множества

$$\Omega\left(\{(x_1,x_2,\ldots,x_m) \in V_m : (x_1,x_2,\ldots,x_{m-1-i}) = (0,0,\ldots,0), x_{m-i} = 1\}\right).$$

Обозначим получившийся подграф через H'_m. Добавим к H'_m вершину $\omega(0,0,\ldots,0)$ и обе исходящие из нее в графе GB_m дуги; получившийся подграф обозначим H_m. На рис. 1 приведен пример такого графа для $m = 3$.

Обозначим через c_m число правильных троек разметок H_m. Пусть среди правильных троек разметок графа H'_m есть ровно $4a_m$ таких, что $\varphi_{(10)}(\omega(0,0,\ldots,0,1)) = 0$, и $4b_m$ таких, что $\varphi_{(10)}(\omega(0,0,\ldots,0,1)) = 1$.

Учитывая структуру графов H'_m и H_m, приходим к следующему утверждению.

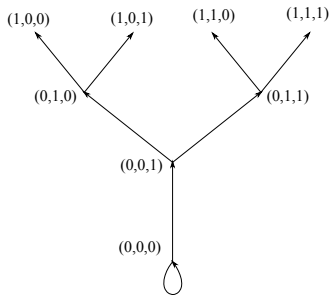

Рис. 1. Граф H_3

Утверждение 2. Значения a_m, b_m, c_m, $m = 1, 2, \ldots$, удовлетворяют следующим равенствам:

$$a_1 = b_1 = 1;$$
$$a_{m+1} = 4(a_m + 2b_m)^2,\ m = 1, 2, \ldots;$$
$$b_{m+1} = 4a_m^2,\ m = 1, 2, \ldots;$$
$$c_m = 4a_m + 8b_m,\ m = 1, 2, \ldots$$

С учетом полученных результатов докажем верхнюю оценку.

Теорема 6. Для всякого $n \geqslant 3$ верно

$$\log_2 |W_{3,n}| < 2^{n-2} \cdot 2{,}100641.$$

Доказательство. Из утверждения 1, определений графа H_m и величины c_m следует, что $|W_{3,n}| \leqslant c_{n-2}$.

Обозначим для всякого $m = 1, 2, \ldots$ отношение a_m/b_m через d_m. Получим выражение c_{m+1} через c_m и d_m:

$$c_{m+1} = 4a_{m+1} + 8b_{m+1} = 16a_m^2 + 64a_m b_m + 64b_m^2 + 32a_m^2 =$$
$$= c_m^2 \frac{48a_m^2 + 64a_m b_m + 64b_m^2}{16a_m^2 + 64a_m b_m + 64b_m^2} = c_m^2 \frac{3d_m^2 + 4d_m + 4}{d_m^2 + 4d_m + 4}.$$

Выражая d_{m+1} через d_m, получим $d_{m+1} = 1 + 4\dfrac{d_m + 1}{d_m^2}$. Найдем отрезок числовой оси, которому принадлежат все значения d_m, начиная с некоторого номера m^*. Покажем, что в качестве такого отрезка можно выбрать отрезок от 2,867 до 2,882. Для этого заметим, что если для некоторого m^* верно $2{,}867 \leqslant d_{m^*} \leqslant 2{,}882$, то и для всех $m \geqslant m^*$ верно $2{,}867 \leqslant d_m \leqslant 2{,}882$. Учитывая, что $d_1 = 1$ и вычисляя явно все d_m вплоть до d_{32}, получим, что указанные неравенства выполняются при $m^* = 32$. Таким образом, для любого $m \geqslant 32$ верно неравенство $c_{m+1} \leqslant 1{,}697 \cdot c_m^2$ и, следовательно, $c_m \leqslant 1{,}697^{-1}(1{,}697 \cdot c_{32})^{2^{m-32}}$, $\log_2 c_m \leqslant 2^m(\log_2 1{,}697 + \log_2 c_{32})/2^{32} - \log_2 1{,}697$. Вычисляя $\log_2 c_{32}$, получаем $\log_2 c_m < 2^m \cdot 2{,}100641$ для всех $m \geqslant 32$. Проверяя явным образом выполнение данного соотношения для всех $m = 1, 2, \ldots, 31$, приходим к требуемому утверждению. ■

Из теорем 4 и 6 окончательно получаем следующие оценки мощности $W_{3,n}$:

$$2^{n-2}\left(C_1 - \mathrm{O}(1/\sqrt{n})\right) \leqslant \log_2 |W_{3,n}| < 2^{n-2} \cdot C_2,$$

где $C_1 = 1 + (\log_2 5)/4 \approx 1{,}58048$; $C_2 = 2{,}100641$.

ЛИТЕРАТУРА

1. *Hedlund G. A.* Endomorphisms and automorphisms of the shift dynamical system // Math. Sys. Theory. 1969. No. 3. P. 320–375.
2. *Сумароков С. Н.* Запреты двоичных функций и обратимость для одного класса кодирующих устройств // Обозрение прикладной и промышленной математики. 1994. Т. 1. Вып. 1. С. 33–55.
3. *Anderson R. J.* Searching for the Optimum Correlation Attack // LNCS. 1995. V. 1008. P. 137–143.
4. *Golic Dj. J.* On the Security of Nonlinear Filter Generators // LNCS. 1996. V. 1039. P. 173–188.
5. *Смышляев С. В.* О некоторых свойствах совершенно уравновешенных булевых функций // Материалы Четвертой Междунар. науч. конф. по проблемам безопасности и противодействия терроризму (МГУ им. М. В. Ломоносова, Москва, 30–31 октября 2008). М.: МЦНМО, 2009. С. 57–64.
6. *Логачев О. А., Смышляев С. В., Ященко В. В.* Новые методы изучения совершенно уравновешенных булевых функций // Дискретная математика. 2009. Т. 21. Вып. 2. С. 51–74.
7. *Логачев О. А.* Об одном классе совершенно уравновешенных булевых функций // Материалы Третьей Междунар. науч. конф. по проблемам безопасности и противодействия терроризму (МГУ им. М. В. Ломоносова, Москва, 25–27 октября 2007). М.: МЦНМО, 2008. С. 137–141.
8. *Смышляев С. В.* Барьеры совершенно уравновешенных булевых функций // Дискретная математика. 2010. Т. 22. Вып. 2. С. 66–79.
9. *Смышляев С. В.* О преобразовании двоичных последовательностей с помощью совершенно уравновешенных булевых функций // Материалы Пятой Междунар. науч. конф. по проблемам безопасности и противодействия терроризму (МГУ им. М. В. Ломоносова, Москва, 29–30 октября 2009). М.: МЦНМО, 2010. С. 31–41.
10. *Смышляев С. В.* О криптографических слабостях некоторых классов преобразований двоичных последовательностей // Прикладная дискретная математика. 2010. № 1(7). С. 5–15.
11. *Смышляев С. В.* О совершенно уравновешенных булевых функциях без барьера // Материалы Восьмой Междунар. науч. конф. «Дискретные модели в теории управляющих систем» (МГУ им. М. В. Ломоносова, Москва, 6–9 апреля 2009). М.: МАКС Пресс, 2009. С. 278–284.
12. *Смышляев С. В.* Построение классов совершенно уравновешенных булевых функций без барьера // Прикладная дискретная математика. 2010. № 3(9). С. 41–50.
13. *Холл М.* Комбинаторика. М.: Мир, 1970.
14. *Lichiardopol N.* Independence number of de Bruijn graphs // Dicrete Mathematics. 2006. V. 306(12). P. 1145–1160.

2011 Теоретические основы прикладной дискретной математики №1(11)

DOI 10.17223/20710410/11/4

УДК 519.1, 519.7

ОЦЕНКА НЕЛИНЕЙНОСТИ КОРРЕЛЯЦИОННО-ИММУННЫХ БУЛЕВЫХ ФУНКЦИЙ[1]

А. В. Халявин

Московский государственный университет им. М. В. Ломоносова, г. Москва, Россия

E-mail: halyavin@gmail.com

Исследуется точность оценки нелинейности булевых функций от n переменных, корреляционно-иммунных порядка m: $\mathrm{nl}(f) \leqslant 2^{n-1} - 2^m$. Показывается, что для всех пар значений $n \geqslant 512$ и $0 < m < n-1$, кроме двух серий $m = 2^s$, $n = 2^{s+1}+1$ и $m = 2^s+1$, $n = 2^{s+1}+2$ при $s \geqslant 0$, эту оценку можно улучшить до $\mathrm{nl}(f) \leqslant 2^{n-1} - 2^{m+1}$. Справедливость результата для $n < 512$, $0 < m < n-1$ проверена на компьютере.

Ключевые слова: *булевы функции, нелинейность, корреляционная иммунность.*

Введение

Булевы функции активно применяются при построении быстрых блочных и поточных шифров. Стойкость получающихся шифров напрямую зависит от характеристик этих булевых функций. Наиболее важными из них являются корреляционная иммунность, нелинейность, устойчивость, алгебраическая иммунность. Отсюда вытекает сложная задача построения функций с наилучшими значениями этих параметров. Большинство вопросов в этой области по-прежнему остаются открытыми.

В частности, не известна точная граница нелинейности для корреляционно-иммунных порядка m функций от n переменных. В работах [1–3] независимо друг от друга была доказана оценка $\mathrm{nl}(f) \leqslant 2^{n-1} - 2^m$. При $0 < m \leqslant n/2 - 1$ лучшую оценку дает равенство Парсеваля, из которого следует $\mathrm{nl}(f) \leqslant 2^{n-1} - 2^{n/2-1}$. При этом равенство $\mathrm{nl}(f) = 2^{n-1} - 2^{n/2-1}$ достигается лишь для бент-функций, которые не бывают корреляционно-иммунными. Установлено, что при $m > n/2 - 1$ из равенства $\mathrm{nl}(f) = 2^{n-1} - 2^m$ следует комбинаторное соотношение для n и m, которое будет дано далее. Единственными известными значениями параметров n и $\max(0, n/2-1) < m < n-1$, при которых оно выполняется, являются серии $m = 2^s$, $n = 2^{s+1}+1$ и $m = 2^s+1$, $n = 2^{s+1}+2$ при $s \geqslant 0$. Следует отметить, что для малых значений s функции с этими параметрами действительно существуют. В частности, легко построить функции для $n = 3, 5, 6$ (примеры можно найти в статье [4]), а в работе [5] построены примеры функций для $n = 9, 10$. Однако доказать отсутствие других подходящих пар значений n и m долгое время не удавалось. Ближе всего к этому вопросу удалось подобраться в работе [6], где было доказано соотношение

$$\frac{n}{2} + \frac{1}{2}\log_2 n + \frac{1}{2}\log_2\left(\frac{\pi}{2}e^{8/9}\right) - 1 > m \geqslant \frac{n-1}{2} \quad \text{для} \quad n \geqslant 12.$$

[1]Работа поддержана грантом РФФИ (проект 08-01-00863), грантом ведущих научных школ РФ (проект НШ-4437.2010.1) и программой фундаментальных исследований ОМН РАН «Алгебраические и комбинаторные методы математической кибернетики и информационные системы нового поколения».

В данной работе докажем, что при $n \geqslant 512$ никаких других подходящих пар, кроме указанных двух серий, не существует. Значения $n < 512$ легко проверить на компьютере.

1. Основные определения

Пусть $F_2 = \{0, 1\}$ — поле из двух элементов. Под *булевыми функциями* будем понимать отображения $f : F_2^n \mapsto F_2$. *Весом* $\mathrm{wt}(u)$ набора $u \in F_2^n$ назовём количество единиц в нём. *Весом* $\mathrm{wt}(f)$ булевой функции f будем называть число её единичных значений. *Расстояние* между булевыми функциями — это число наборов, значения функций на которых различаются. *Нелинейностью* $\mathrm{nl}(f)$ булевой функции f назовём расстояние до ближайшей к ней аффинной функции. Будем говорить, что набор $x \in F_2^n$ *мажорируется* набором $y \in F_2^n$, и обозначим это $x \preccurlyeq y$, если $x_i \leqslant y_i$ для всех $i = 1, \ldots, n$. *Подфункцией* булевой функции f назовем функцию, полученную из неё подстановкой констант вместо некоторых аргументов. Булева функция f от n переменных называется *корреляционно-иммунной порядка* m, $1 \leqslant m \leqslant n$, если для всех её подфункций f' от $n - m$ переменных выполняется соотношение $\mathrm{wt}(f') = \mathrm{wt}(f)/2^m$. Для изучения корреляционно-иммунных функций активно используется *преобразование Уолша*

$$W_f(u) = \sum_{x \in F_2^n} (-1)^{f(x)+(x,u)},$$

где (x, u) — скалярное произведение векторов x и u. Числа $W_f(u)$ называются коэффициентами Уолша и обладают множеством замечательных свойств. Сформулируем некоторые из них.

Равенство Парсеваля:

$$\sum_{u \in F_2^n} W_f^2(u) = 2^{2n}.$$

Равенство Саркара:

$$\sum_{u \in F_2^n, u \preccurlyeq w} W_f(u) = 2^n - 2^{\mathrm{wt}(w)+1}\mathrm{wt}(f_w),$$

где f_w — подфункция f, полученная подстановкой нулей вместо аргументов x_i для всех таких i, что $w_i = 1$.

Многие свойства булевой функции f могут быть выражены в терминах коэффициентов Уолша. В частности, уравновешенность равносильна $W_f(0) = 0$, а корреляционная иммунность порядка m — условию $W_f(u) = 0$ при $1 \leqslant \mathrm{wt}(u) \leqslant m$ [7]. Кроме того, нелинейность булевой функции выражается равенством $\mathrm{nl}(f) = 2^{n-1} - \dfrac{1}{2} \max\limits_{u \in F_2^n} |W_f(u)|$.

Корреляционно-иммунную порядка $m > 0$ функцию f от n переменных будем называть *экстремальной*, если $\mathrm{nl}(f) = 2^{n-1} - 2^m$.

2. Строение спектра экстремальных функций

Рассмотрим корреляционно-иммунную порядка m, $0 < m < n - 1$, булеву функцию f от n переменных. Известно [1, 4], что все её коэффициенты Уолша $W_f(u)$ делятся на 2^{m+1}.

Отсюда следует, что найдется коэффициент Уолша, по модулю не меньший 2^{m+1}, а значит, $\mathrm{nl}(f) = 2^{n-1} - \dfrac{1}{2} \max\limits_{u \in F_2^n} |W_f(u)| \leqslant 2^{n-1} - 2^m$. Кроме того, если неравенство строгое, то есть коэффициент Уолша, который по модулю не меньше чем 2^{m+2}, а значит, выполнено $\mathrm{nl}(f) \leqslant 2^{n-1} - 2^{m+1}$.

Если $m < (n-1)/2$, то, с учётом равенства Парсеваля, имеем $\mathrm{nl}(f) \leqslant 2^{n-1} - 2^{n/2-1} \leqslant 2^{n-1} - 2^m$. Равенство $\mathrm{nl}(f) = 2^{n-1} - 2^{n/2-1}$ возможно только на бент-функциях, которые не обладают свойством корреляционной иммунности. Значит, экстремальные функции могут существовать только при $m \geqslant (n-1)/2$. Изучим строение спектра таких функций более подробно.

Теорема 1. Если корреляционно-иммунная порядка m булева функция f от n переменных, $m \leqslant n-2$, имеет нелинейность $2^{n-1} - 2^m$, то для всех $u \in F_2^n$ выполнено условие

$$W_f(u) \equiv \pi_{\mathrm{wt}(u)} 2^{m+1} \pmod{2^{m+2}},$$

где $\pi_0 = 1$, $\pi_1 = \pi_2 = \cdots = \pi_m = 0$, $\pi_i = \sum\limits_{j=0}^{i-1} \pi_j \binom{i}{j} \bmod 2$ при $i > m$.

Доказательство. В силу корреляционной иммунности функции f имеет место $W_f(u) = 0$ при $0 < \mathrm{wt}(u) \leqslant m$.

Как уже отмечено, $W_f(0)$ делится на 2^{m+1}. Предположим, что $W_f(0)$ делится на 2^{m+2}. Докажем индукцией по $\mathrm{wt}(u)$, что в таком случае все коэффициенты $W_f(u)$ также делятся на 2^{m+2}. База для $\mathrm{wt}(u) \leqslant m$ очевидна. Докажем переход. Пусть $\mathrm{wt}(u) > m$. По равенству Саркара $\sum\limits_{v \preccurlyeq u} W_f(v) = 2^n - 2^{\mathrm{wt}(u)+1}\mathrm{wt}(f_u)$. Правая часть делится на 2^{m+2}. Кроме того, все слагаемые в сумме, кроме $W_f(u)$, делятся на 2^{m+2} по предположению индукции. Значит, и $W_f(u)$ делится на 2^{m+2}. Переход доказан. Из того, что все коэффициенты Уолша делятся на 2^{m+2}, получаем противоречие: $\mathrm{nl}(f) = 2^{n-1} - \dfrac{1}{2}\max\limits_u |W_f(u)| \leqslant 2^{n-1} - 2^{m+1}$. Следовательно, $W_f(0) \equiv 2^{m+1} \pmod{2^{m+2}}$.

Докажем теперь утверждение теоремы при $\mathrm{wt}(u) > m$ индукцией по $\mathrm{wt}(u)$. Как и раньше, получаем, что $\sum\limits_{v \preccurlyeq u} W_f(v) \equiv 0 \pmod{2^{m+2}}$. С другой стороны, по предположению индукции

$$\sum_{v \preccurlyeq u} W_f(v) \equiv \sum_{j=0}^{\mathrm{wt}(u)-1} \binom{\mathrm{wt}(u)}{j} \pi_j 2^{m+1} + W_f(u) \equiv \pi_{\mathrm{wt}(u)} 2^{m+1} + W_f(u) \pmod{2^{m+2}},$$

откуда ввиду $2^{m+1} \equiv -2^{m+1} \pmod{2^{m+2}}$ получаем требуемое. Переход доказан. ■

Заметим теперь, что из $\mathrm{nl}(f) = 2^{n-1} - \dfrac{1}{2}\max\limits_u |W_f(u)| = 2^{n-1} - 2^m$ следует $|W_f(u)| \leqslant 2^{m+1}$, а значит, $|W_f(u)| = \pi_{\mathrm{wt}(u)} 2^{m+1}$. Применяя равенство Парсеваля, получаем следующую теорему.

Теорема 2. Если существует корреляционно-иммунная порядка m булева функция от n переменных, $m \leqslant n-2$, с нелинейностью $2^{n-1} - 2^m$, то выполнено

$$\sum_{j=0}^{n} \binom{n}{j} \pi_j = 2^{2n-2m-2}. \tag{1}$$

Далее покажем в лемме 7, что равенство (1) выполнено при $n = 2^{s+1} + 1$, $m = 2^s$ и $n = 2^{s+1} + 2$, $m = 2^s + 1$, где $s \geqslant 0$. Однако ранее оставался открытым вопрос о возможности выполнения равенства для других значений n и m. Последующие разделы посвящены доказательству того, что других подходящих пар при $n-2 \geqslant m \geqslant (n-1)/2$, $n \geqslant 512$ не существует.

3. Остатки биномиальных коэффициентов по модулю 2^k

Известно, что двоичная запись чисел n и m тесно связана с остатком $\binom{n}{m}$ по модулю 2^k. Покажем, как именно устроена эта связь. Обозначим $F(1)=1$, $F(x)=\prod\limits_{i=1}^{x-1}(2i+1)$ при $x>1$, $F(x)=1/\prod\limits_{i=x}^{0}(2i+1)$ при $x\leqslant 0$. Поскольку все множители нечётны, то можно говорить о значении $F(x)$ по модулю 2^k для всех целых x. Кроме того, легко видеть, что $F(x)=(2x)!/x!$ при $x\geqslant 0$.

Лемма 1. *$F(x+2^{k-1})\equiv cF(x) \pmod{2^k}$ для некоторого c, зависящего только от k.*

Доказательство. $F(x+2^{k-1})/F(x)\equiv\prod\limits_{i=x}^{x+2^{k-1}-1}(2i+1)\equiv\prod\limits_{i=0}^{2^{k-1}-1}(2i+1) \pmod{2^k}$. Последнее равенство выполнено потому, что в обеих частях стоит произведение всех нечётных остатков по модулю 2^k. Таким образом, утверждение теоремы будет выполнено, если в качестве c взять $\prod\limits_{i=0}^{2^{k-1}-1}(2i+1)$. ■

Введем еще одну функцию $G(x,y)=\dfrac{F(x)}{F(y)F(x-y)}$.

Лемма 2. *$G(x,y) \pmod{2^k}$ периодична с периодом 2^{k-1} по обоим аргументам.*

Доказательство.

$G(x,y+2^{k-1})\equiv\dfrac{F(x)}{F(y+2^{k-1})F(x-y-2^{k-1})}\equiv\dfrac{F(x)}{cF(y)c^{-1}F(x-y)}\equiv\dfrac{F(x)}{F(y)F(x-y)}\equiv$
$\equiv G(x,y) \pmod{2^k}$. Отсюда следует периодичность по второму аргументу. Для первого аргумента выкладки аналогичны:

$G(x+2^{k-1},y)\equiv\dfrac{F(x+2^{k-1})}{F(y)F(x-y+2^{k-1})}\equiv\dfrac{cF(x)}{F(y)cF(x-y)}\equiv\dfrac{F(x)}{F(y)F(x-y)}\equiv$
$\equiv G(x,y) \pmod{2^k}$. ■

Лемма 3. $\binom{2n}{2m}=\binom{n}{m}G(n,m)$.

Доказательство. Если $m>n$, то обе части равны нулю. В противном случае $\binom{2n}{2m}=\dfrac{2n!}{2m!(2n-2m)!}=\dfrac{(2n)!m!(n-m)!}{n!(2m)!(2n-2m)!}\cdot\dfrac{n!}{m!(n-m)!}=\dfrac{F(n)}{F(m)F(n-m)}\binom{n}{m}=$
$=\binom{n}{m}G(n,m)$. ■

Заметим, что $\binom{2n+1}{2m}=\binom{2n}{2m}\dfrac{2n+1}{2n-2m+1}$, $\binom{2n}{2m+1}=\binom{2n}{2m}\dfrac{2n-2m}{2m+1}$, $\binom{2n+1}{2m+1}=\binom{2n}{2m}\dfrac{2n+1}{2m+1}$. Рассмотрев эти равенства по модулю 2^k, получаем следующую теорему.

Теорема 3. $\binom{2n+a}{2m+b}\equiv\binom{n}{m}H((2n+a) \bmod 2^k,(2m+b) \bmod 2^k) \pmod{2^k}$, где $a,b\in\{0,1\}$, $H(2x,2y)=G(x,y)$, $H(2x+1,2y)=G(x,y)\dfrac{2x+1}{2x-2y+1}$, $H(2x,2y+1)=$
$=G(x,y)\dfrac{2x-2y}{2y+1}$, $H(2x+1,2y+1)=G(x,y)\dfrac{2x+1}{2y+1}$.

Обозначим M_k матрицу значений функции H по модулю 2^k:

$$M_k = \begin{bmatrix} H(0,0) \bmod 2^k & \cdots & H(0,2^k-1) \bmod 2^k \\ \vdots & \ddots & \vdots \\ H(2^k-1,0) \bmod 2^k & \cdots & H(2^k-1,2^k-1) \bmod 2^k \end{bmatrix}.$$

Элемент этой матрицы в $(i+1)$-й строке и $(j+1)$-м столбце будем записывать как $M_k(i,j) = H(i,j) \bmod 2^k$. Матрицы с маленькими индексами имеют следующие значения:

$$M_1 = \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}, \quad M_2 = \begin{bmatrix} 1 & 0 & 3 & 2 \\ 1 & 1 & 1 & 1 \\ 1 & 2 & 1 & 0 \\ 1 & 3 & 3 & 1 \end{bmatrix}, \quad M_3 = \begin{bmatrix} 1 & 0 & 7 & 6 & 1 & 4 & 7 & 2 \\ 1 & 1 & 1 & 5 & 5 & 5 & 5 & 1 \\ 1 & 2 & 1 & 0 & 5 & 6 & 5 & 4 \\ 1 & 3 & 3 & 1 & 1 & 3 & 3 & 1 \\ 1 & 4 & 3 & 2 & 1 & 0 & 3 & 6 \\ 1 & 5 & 5 & 5 & 5 & 1 & 1 & 1 \\ 1 & 6 & 5 & 4 & 5 & 2 & 1 & 0 \\ 1 & 7 & 7 & 1 & 1 & 7 & 7 & 1 \end{bmatrix}.$$

Пусть $wa = a_{s-1}\ldots a_1a_0$ и $wb = b_{s-1}\ldots b_1b_0$ — два двоичных слова. Обозначим $\Pi_k(wa, wb) = \prod\limits_{i=0}^{s-k} M_k\left(\sum\limits_{j=0}^{k-1} a_{i+j}2^j, \sum\limits_{j=0}^{k-1} b_{i+j}2^j\right)$.

Для чисел $a = \sum\limits_{j=0}^{s-1} a_j2^j$ и $b = \sum\limits_{j=0}^{s-1} b_j2^j$, последовательно применяя теорему 3 вместе с определениями M_k и Π_k, получим следующее утверждение.

Лемма 4. $\binom{b}{a} \equiv \Pi_k(wb, wa) \begin{pmatrix} \sum\limits_{j=0}^{k-2} b_{s-k+1+j}2^j \\ \sum\limits_{j=0}^{k-2} a_{s-k+1+j}2^j \end{pmatrix} \pmod{2^k}$.

Для каждой матрицы $M = \{m_{ij}\}$ размера $2^k \times 2^k$ введём функцию $M(wa, wb) = m_{1+[a/2^{s-k}],1+[b/2^{s-k}]}$. Аналогично для строки или столбца $\{m_i\}$ размера 2^k введём функцию $M(wa) = m_{1+[a/2^{s-k}]}$. Определим семейство матриц

$$M_k^* = \begin{bmatrix} \binom{0}{0} & \cdots & \binom{0}{2^k-1} \\ \vdots & \ddots & \vdots \\ \binom{2^k-1}{0} & \cdots & \binom{2^k-1}{2^k-1} \end{bmatrix}.$$

Для маленьких k получаем $M_0^* = [1]$, $M_1^* = \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}$, $M_2^* = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 2 & 1 & 0 \\ 1 & 3 & 3 & 1 \end{bmatrix}$. Тогда биномиальные коэффициенты можно полностью выразить через функции от их двоичных записей:

$$\binom{b}{a} \equiv \Pi_k(wb, wa) M_{k-1}^*(wb, wa) \pmod{2^k}. \tag{2}$$

4. Делимость сумм биномиальных коэффициентов

Для начала выведем более простую формулу для π_i.

Лемма 5. При $i > m$ выполнено

$$\sum_{j=m}^{i-1} \binom{j}{m}\binom{i}{j} \equiv \binom{i}{m} \pmod 2.$$

Доказательство. Из строения матрицы M_1^* видно, что биномиальный коэффициент $\binom{a}{b}$ отличен от 0 по модулю 2 тогда и только тогда, когда двоичная запись a мажорирует двоичную запись b. Поэтому если двоичная запись i не мажорирует двоичную запись m, то правая часть чётна; левая часть также чётна, поскольку в силу транзитивности мажорирования хотя бы один из множителей в каждом слагаемом будет чётным. Если же двоичная запись i мажорирует двоичную запись m и в ней на k единичных бит больше, то в левой части будет $2^k - 1$ нечётных слагаемых. Поскольку $i \neq m$, то $k > 0$; значит, левая часть будет нечётной. ■

Лемма 6. При $i > m$ выполнено

$$\pi_i \equiv \binom{i-1}{m} \pmod 2.$$

Доказательство. Докажем это утверждение индукцией по i. База $i = m + 1$: $\pi_{m+1} = 1 = \binom{m}{m}$. Докажем переход:

$$\pi_i = 1 + \sum_{j=m+1}^{i-1} \pi_j \binom{i}{j} \equiv 1 + \sum_{j=m+1}^{i-1} \binom{j-1}{m}\binom{i}{j} = 1 + \sum_{j=m+1}^{i-1} \binom{j-1}{m}\binom{i-1}{j} +$$
$$+ \sum_{j=m+1}^{i-1} \binom{j-1}{m}\binom{i-1}{j-1} \equiv 1 + \sum_{j=m+1}^{i-2} \pi_j \binom{i-1}{j} + \binom{i-2}{m}\binom{i-1}{i-1} +$$
$$+ \sum_{j=m}^{i-2} \binom{j}{m}\binom{i-1}{j} = \pi_{i-1} + \binom{i-2}{m} + \sum_{j=m}^{i-2} \binom{j}{m}\binom{i-1}{j} \equiv \binom{i-2}{m} + \binom{i-2}{m} +$$
$$+ \sum_{j=m}^{i-2} \binom{j}{m}\binom{i-1}{j} \equiv \sum_{j=m}^{i-2} \binom{j}{m}\binom{i-1}{j} \equiv \binom{i-1}{m} \pmod 2.$$

Здесь последнее равенство следует из леммы 5 ввиду $i > m + 1$. ■

Используя лемму 6, условие (1) можно преобразовать к виду

$$1 + \sum_{\substack{i, \\ \binom{i}{m} \equiv 1 \pmod 2}} \binom{n}{i+1} = 2^{2n-2m-2}. \tag{3}$$

Лемма 7. Равенства (1) и (3) верны при $n = 2^{s+1} + 1$, $m = 2^s$ и $n = 2^{s+1} + 2$, $m = 2^s + 1$, где $s \geqslant 0$.

Доказательство. Воспользуемся тем, что $\binom{i}{m} \equiv 1 \pmod 2$ тогда и только тогда, когда двоичная запись i мажорирует двоичную запись m. В случае $n = 2^{s+1}+1$, $m = 2^s$

сумма равна

$$1+\sum_{\substack{i,\\ \binom{i}{m}\equiv 1 \pmod 2}}\binom{n}{i+1}=1+\sum_{i=2^s+1}^{2^{s+1}}\binom{2^{s+1}+1}{i}=$$
$$=\sum_{i=2^s+1}^{2^{s+1}+1}\binom{2^{s+1}+1}{i}=2^{2^{s+1}}=2^{2n-2m-2}.$$

В случае же $n=2^{s+1}+2$, $m=2^s+1$ сумма равна

$$1+\sum_{\substack{i,\\ \binom{i}{m}\equiv 1 \pmod 2}}\binom{n}{i+1}=1+\sum_{i=2^s+2,\ 2|i}^{2^{s+1}}\binom{2^{s+1}+2}{i}=$$
$$=\sum_{i=2^s+2,\ 2|i}^{2^{s+1}+2}\binom{2^{s+1}+2}{i}=\frac{1}{2}\sum_{i=0,\ 2|i}^{2^{s+1}+2}\binom{2^{s+1}+2}{i}=\frac{1}{2}2^{2^{s+1}+1}=2^{2^{s+1}}=2^{2n-2m-2}.$$

Лемма доказана. ■

Выразим теперь условие суммирования в (3) в терминах слов. Пусть $s=$ $=[\log_2\max(n,m)]+1$, wn и wm — двоичные записи длины s чисел n и m соответственно. Если $i=2^s-1$, то $i+1=2^s>n$, а значит, $\binom{n}{i+1}=0$. Поэтому значение $i=2^s-1$ можно исключить из суммирования. Используя теперь (2), получаем, что

$$1+\sum_{\substack{i,\\ \binom{i}{m}\equiv 1 \pmod 2}}\binom{n}{i+1}\equiv$$
$$\equiv 1+\sum_{\substack{wi\neq 11\ldots1,\\ wm\preccurlyeq wi}}\Pi_k(wn,wi+1)M^*_{k-1}(wn,wi+1)\pmod{2^k},$$

где под $wi+1$ понимается слово той же длины s, представляющее двоичную запись числа $i+1 \pmod{2^s}$. Поскольку $i<2^s-1$, это слово всегда представляет число $i+1$.

Обозначим $\overline{w}$ — натуральное число, соответствующее слову w; $|w|$ — длина слова w.

Лемма 8. Пусть $\overline{wm}<\overline{wn}$. Тогда

$$\sum_{\substack{wi\neq 11\ldots1,\\ wm\preccurlyeq wi}}\Pi_1(wn,wi+1)\equiv 1\pmod 2.$$

Доказательство. Будем доказывать это утверждение индукцией по длине слов. Если $|wn|=1$, то $wn=1$, $wm=0$. В сумме оказывается только одно слагаемое, соответствующее $wi=0$: $\Pi_1(1,1)=1$. База индукции доказана. Докажем переход. Пусть утверждение верно для $|wm|=|wn|\leqslant k$; докажем его для $|wm|=|wn|=k+1$.

Пусть $wm=1w_1$, $wn=1w_2$, $\overline{w_1}<\overline{w_2}$. Получаем

$$\sum_{\substack{wi\neq 11\ldots1,\\ 1w_1\preccurlyeq wi}}\Pi_1(1w_2,wi+1)=\sum_{\substack{wi\neq 11\ldots1,\\ w_1\preccurlyeq wi}}\Pi_1(1w_2,1(wi+1))=\sum_{\substack{wi\neq 11\ldots1,\\ w_1\preccurlyeq wi}}\Pi_1(w_2,wi+1).$$

Последнее выражение равно 1 по модулю 2 в силу предположения индукции.

Пусть $wm = 0w_1$, $wn = 1w_2$. Получаем

$$\sum_{\substack{wi\neq 11\ldots 1,\\ 0w_1\preccurlyeq wi}} \Pi_1(1w_2, wi+1) = \Pi_1(1w_2, 100\ldots 0) + \sum_{\substack{wi\neq 11\ldots 1,\\ w_1\preccurlyeq wi}} \Pi_1(1w_2, 0(wi+1)) +$$
$$+ \sum_{\substack{wi\neq 11\ldots 1,\\ w_1\preccurlyeq wi}} \Pi_1(1w_2, 1(wi+1)) = 1 + \sum_{\substack{wi\neq 11\ldots 1,\\ w_1\preccurlyeq wi}} (\Pi_1(w_2, wi+1) + \Pi_1(w_2, wi+1)) \equiv 1 \pmod 2.$$

Пусть $wm = 0w_1$, $wn = 0w_2$, $\overline{w_1} < \overline{w_2}$. Получаем

$$\sum_{\substack{wi\neq 11\ldots 1,\\ 0w_1\preccurlyeq wi}} \Pi_1(0w_2, wi+1) = \Pi_1(0w_2, 100\ldots 0) + \sum_{\substack{wi\neq 11\ldots 1,\\ w_1\preccurlyeq wi}} \Pi_1(0w_2, 0(wi+1)) +$$
$$+ \sum_{\substack{wi\neq 11\ldots 1,\\ w_1\preccurlyeq wi}} \Pi_1(0w_2, 1(wi+1)) = 0 + \sum_{\substack{wi\neq 11\ldots 1,\\ w_1\preccurlyeq wi}} (\Pi_1(w_2, wi+1) + \Pi_1(w_2, wi+1)\cdot 0) =$$
$$= \sum_{\substack{wi\neq 11\ldots 1,\\ w_1\preccurlyeq wi}} \Pi_1(w_2, wi+1) \equiv 1 \pmod 2.$$

Последнее сравнение верно в силу предположения индукции. Переход доказан. ■

Лемма 9. Пусть $\overline{wm} < \overline{wn}$. Тогда

$$\begin{bmatrix} 1 \\ 0 \end{bmatrix} (wn) + \sum_{\substack{wi\neq 11\ldots 1,\\ wm\preccurlyeq wi}} \Pi_2(wn, wi+1) \begin{bmatrix} 0 & 1 \\ 1 & 1 \end{bmatrix} (wn, wi+1) \equiv 1 \pmod 2.$$

Доказательство. Если длина слов wm и wn равна 1, то $wn = 1$, $wm = 0$. Первое слагаемое получается равным нулю, а в сумме оказывается только одно слагаемое, соответствующее $wi = 0$: $\Pi_2(1,1) \begin{bmatrix} 0 & 1 \\ 1 & 1 \end{bmatrix} (1,1) = 1 \cdot 1 = 1$. Равенство доказано.

Пусть теперь длина слов больше 1. Пусть $wm = 1w_1$, $wn = 1w_2$, $\overline{w_1} < \overline{w_2}$. Получаем

$$\begin{bmatrix} 1 \\ 0 \end{bmatrix} (1w_2) + \sum_{\substack{wi\neq 11\ldots 1,\\ 1w_1\preccurlyeq wi}} \Pi_2(1w_2, wi+1) \begin{bmatrix} 0 & 1 \\ 1 & 1 \end{bmatrix} (1w_2, wi+1) \equiv$$
$$\equiv \sum_{\substack{wi\neq 11\ldots 1,\\ w_1\preccurlyeq wi}} \Pi_1(w_2, wi+1) \pmod 2.$$

К последнему выражению применяем лемму 8 и получаем требуемое.

Пусть $wm = 0w_1$, $wn = 1w_2$. Получаем

$$\begin{bmatrix} 1 \\ 0 \end{bmatrix} (1w_2) + \sum_{\substack{wi\neq 11\ldots 1,\\ 0w_1\preccurlyeq wi}} \Pi_2(1w_2, wi+1) \begin{bmatrix} 0 & 1 \\ 1 & 1 \end{bmatrix} (1w_2, wi+1) =$$
$$= \Pi_2(1w_2, 100\ldots 0) \begin{bmatrix} 0 & 1 \\ 1 & 1 \end{bmatrix} (1w_2, 100\ldots 0) + \sum_{\substack{wi\neq 11\ldots 1,\\ w_1\preccurlyeq wi}} \Pi_2(1w_2, 0(wi+1)) +$$
$$+ \sum_{\substack{wi\neq 11\ldots 1,\\ w_1\preccurlyeq wi}} \Pi_2(1w_2, 1(wi+1)) \equiv \Pi_1(w_2, 00\ldots 0) + \sum_{\substack{wi\neq 11\ldots 1,\\ w_1\preccurlyeq wi}} (\Pi_1(w_2, wi+1) + \Pi_1(w_2, wi+1)) \equiv$$
$$\equiv 1 \pmod 2.$$

Утверждение доказано.

Пусть, наконец, $wm = 0w_1$, $wn = 0w_2$, $\overline{w_1} < \overline{w_2}$. Получаем

$$\begin{bmatrix} 1 \\ 0 \end{bmatrix}(0w_2) + \sum_{\substack{wi \neq 11\ldots1,\\ 0w_1 \preccurlyeq wi}} \Pi_2(0w_2, wi+1) \begin{bmatrix} 0 & 1 \\ 1 & 1 \end{bmatrix}(0w_2, wi+1) =$$

$$= 1 + \Pi_2(0w_2, 100\ldots0) \begin{bmatrix} 0 & 1 \\ 1 & 1 \end{bmatrix}(0w_2, 100\ldots0) + \sum_{\substack{wi \neq 11\ldots1,\\ w_1 \preccurlyeq wi}} \Pi_2(0w_2, 0(wi+1)) \cdot 0 +$$

$$+ \sum_{\substack{wi \neq 11\ldots1,\\ w_1 \preccurlyeq wi}} \Pi_2(0w_2, 1(wi+1)) \cdot 1 \equiv 1 + \Pi_1(w_2, 00\ldots0) \cdot 1 + \sum_{\substack{wi \neq 11\ldots1,\\ w_1 \preccurlyeq wi}} \Pi_1(w_2, wi+1) \equiv$$

$$\equiv \sum_{\substack{wi \neq 11\ldots1,\\ w_1 \preccurlyeq wi}} \Pi_1(w_2, wi+1) \pmod 2.$$

К последнему выражению применяем лемму 8 и получаем требуемое. ■

Лемма 10. Пусть $\overline{wn} \leqslant \overline{wm}$. Тогда

$$\sum_{\substack{wi \neq 11\ldots1,\\ wm \preccurlyeq wi}} \Pi_1(wn, wi+1) \equiv 0 \pmod 2.$$

Доказательство. Докажем это утверждение индукцией по длине слов wm и wn. Для слов длины 1 в сумме есть слагаемые лишь при $wm = 0$, а значит, и $wn = 0$. В этом случае единственное слагаемое равно $\Pi_1(0, 1) = 0$. База индукции доказана.

Докажем переход. Пусть утвеждение доказано для слов длины не больше k, докажем его для слов длины $k+1$.

Пусть $wm = 1w_1$, $wn = 1w_2$. Тогда $\overline{w_2} \leqslant \overline{w_1}$. Получаем

$$\sum_{\substack{wi \neq 11\ldots1,\\ 1w_1 \preccurlyeq wi}} \Pi_1(1w_2, wi+1) = \sum_{\substack{wi \neq 11\ldots1,\\ w_1 \preccurlyeq wi}} \Pi_1(1w_2, 1(wi+1)) = \sum_{\substack{wi \neq 11\ldots1,\\ w_1 \preccurlyeq wi}} \Pi_1(w_2, wi+1).$$

По предположению индукции это выражение сравнимо с 0 по модулю 2.

Пусть $wm = 1w_1$, $wn = 0w_2$. Получаем

$$\sum_{\substack{wi \neq 11\ldots1,\\ 1w_1 \preccurlyeq wi}} \Pi_1(0w_2, wi+1) = \sum_{\substack{wi \neq 11\ldots1,\\ w_1 \preccurlyeq wi}} \Pi_1(0w_2, 1(wi+1)) = 0.$$

Утверждение доказано.

Пусть $wm = 0w_1$, $wn = 0w_2$. Тогда $\overline{w_2} \leqslant \overline{w_1}$. Получаем

$$\sum_{\substack{wi \neq 11\ldots1,\\ 0w_1 \preccurlyeq wi}} \Pi_1(0w_2, wi+1) = \Pi_1(0w_2, 100\ldots0) + \sum_{\substack{wi \neq 11\ldots1,\\ w_1 \preccurlyeq wi}} \Pi_1(0w_2, 0(wi+1)) +$$

$$+ \sum_{\substack{wi \neq 11\ldots1,\\ w_1 \preccurlyeq wi}} \Pi_1(0w_2, 1(wi+1)) = 0 + \sum_{\substack{wi \neq 11\ldots1,\\ w_1 \preccurlyeq wi}} (\Pi_1(w_2, wi+1) + \Pi_1(w_2, wi+1) \cdot 0) =$$

$$= \sum_{\substack{wi \neq 11\ldots1,\\ w_1 \preccurlyeq wi}} \Pi_1(w_2, wi+1).$$

По предположению индукции это выражение сравнимо с 0 по модулю 2. Переход доказан. ■

Теорема 4. Пусть для некоторых непустых слов w_1 и w_2 выполнено $wm = 10w_1$, $wn = 11w_2$ и $\overline{w_2} > \overline{w_1}$; или $wm = 01w_1$, $wn = 10w_2$ и $\overline{w_2} \leqslant \overline{w_1}$. Тогда

$$1 + \sum_{\substack{wi \neq 11\ldots1,\\ wm \preccurlyeq wi}} \Pi_2(wn, wi+1) M_1^*(wn, wi+1) \equiv 2 \pmod 4.$$

Доказательство. Рассмотрим первый случай: $wm = 10w_1$, $wn = 11w_2$ и $\overline{w_2} > \overline{w_1}$. Поскольку wm начинается с 1, то и wi начинается с 1. Получаем

$$1 + \sum_{\substack{wi \neq 11\ldots1,\\ wm \preccurlyeq wi}} \Pi_2(wn, wi+1) M_1^*(wn, wi+1) =$$

$$= 1 + \sum_{\substack{wi \neq 11\ldots1,\\ 0w_1 \preccurlyeq wi}} \Pi_2(11w_2, 1(wi+1)) \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix} (11w_2, 1(wi+1)) =$$

$$= 1 + \sum_{\substack{wi \neq 11\ldots1,\\ 0w_1 \preccurlyeq wi}} \Pi_2(1w_2, wi+1) \begin{bmatrix} 3 & 1 \end{bmatrix} (wi+1) \cdot 1.$$

Выделяем слагаемое $wi = 011\ldots1$, а остальные слагаемые разбиваем на две суммы:

$$1 + \sum_{\substack{wi \neq 11\ldots1,\\ 0w_1 \preccurlyeq wi}} \Pi_2(1w_2, wi+1) \begin{bmatrix} 3 & 1 \end{bmatrix} (wi+1) =$$

$$= 1 + \Pi_2(1w_2, 100\ldots0) \cdot 1 + \sum_{\substack{wi \neq 11\ldots1,\\ w_1 \preccurlyeq wi}} \Pi_2(1w_2, 0(wi+1)) \cdot 3 + \sum_{\substack{wi \neq 11\ldots1,\\ w_1 \preccurlyeq wi}} \Pi_2(1w_2, 1(wi+1)) \cdot 1 =$$

$$= 1 + \begin{bmatrix} 1 \\ 3 \end{bmatrix} (w_2) + \sum_{\substack{wi \neq 11\ldots1,\\ w_1 \preccurlyeq wi}} \Pi_2(w_2, wi+1) \cdot 3 \begin{bmatrix} 1 & 2 \\ 1 & 3 \end{bmatrix} (w_2, wi+1) +$$

$$+ \sum_{\substack{wi \neq 11\ldots1,\\ w_1 \preccurlyeq wi}} \Pi_2(w_2, wi+1) \begin{bmatrix} 1 & 0 \\ 3 & 1 \end{bmatrix} (w_2, wi+1) \equiv$$

$$\equiv \begin{bmatrix} 2 \\ 0 \end{bmatrix} (w_2) + \sum_{\substack{wi \neq 11\ldots1,\\ w_1 \preccurlyeq wi}} \Pi_2(w_2, wi+1) \begin{bmatrix} 0 & 2 \\ 2 & 2 \end{bmatrix} (w_2, wi+1) \pmod 4.$$

Сокращая на 2, получаем, что утверждение теоремы преобразуется к виду

$$\begin{bmatrix} 1 \\ 0 \end{bmatrix} (w_2) + \sum_{\substack{wi \neq 11\ldots1,\\ w_1 \preccurlyeq wi}} \Pi_2(w_2, wi+1) \begin{bmatrix} 0 & 1 \\ 1 & 1 \end{bmatrix} (w_2, wi+1) \equiv 1 \pmod 2,$$

который в точности совпадает с леммой 9.

Рассмотрим теперь случай $wm = 01w_1$, $wn = 10w_2$ и $\overline{w_2} \leqslant \overline{w_1}$:

$$1 + \sum_{\substack{wi \neq 11\ldots1,\\ wm \preccurlyeq wi}} \Pi_2(wn, wi+1) M_1^*(wn, wi+1) =$$

$$= 1 + \sum_{\substack{wi \neq 11\ldots1,\\ 01w_1 \preccurlyeq wi}} \Pi_2(10w_2, wi+1) \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix} (10w_2, wi+1) =$$

$$= 1 + \sum_{\substack{wi \neq 11\ldots1,\\ 01w_1 \preccurlyeq wi}} \Pi_2(10w_2, wi+1) \cdot 1 = 1 + \Pi_2(10w_2, 100\ldots0) +$$

$$+ \sum_{\substack{wi \neq 11\ldots1,\\ 1w_1 \preccurlyeq wi}} \Pi_2(10w_2, 0(wi+1)) + \sum_{\substack{wi \neq 11\ldots1,\\ 1w_1 \preccurlyeq wi}} \Pi_2(10w_2, 1(wi+1)) =$$

$$= 1 + 1 + \sum_{\substack{wi \neq 11\ldots1,\\ 1w_1 \preccurlyeq wi}} (\Pi_2(0w_2, wi+1) \begin{bmatrix} 1 & 2 \end{bmatrix} (wi+1) + \Pi_2(0w_2, wi+1) \begin{bmatrix} 1 & 0 \end{bmatrix} (wi+1)) =$$

$$= 2 + \sum_{\substack{wi \neq 11\ldots1,\\ 1w_1 \preccurlyeq wi}} \Pi_2(0w_2, wi+1) \begin{bmatrix} 2 & 2 \end{bmatrix} (wi+1) = 2 + 2 \sum_{\substack{wi \neq 11\ldots1,\\ 1w_1 \preccurlyeq wi}} \Pi_2(0w_2, wi+1).$$

Сокращая на 2, получаем, что утверждение преобразуется к виду

$$\sum_{\substack{wi\neq 11\ldots 1,\\ 1w_1\preccurlyeq wi}} \Pi_2(0w_2, wi+1) \equiv 0 \pmod 2.$$

Продолжая преобразования, получаем

$$\sum_{\substack{wi\neq 11\ldots 1,\\ 1w_1\preccurlyeq wi}} \Pi_2(0w_2, wi+1) = \sum_{\substack{wi\neq 11\ldots 1,\\ w_1\preccurlyeq wi}} \Pi_2(0w_2, 1(wi+1)) \equiv \sum_{\substack{wi\neq 11\ldots 1,\\ w_1\preccurlyeq wi}} \Pi_1(w_2, wi+1) \pmod 2.$$

Утверждение теперь непосредственно следует из леммы 10. ■

Лемма 11. При $\overline{wm} - 2^{|wm|-1} < \overline{wn}$ выполнено

$$\begin{bmatrix} 0 \\ 1 \end{bmatrix}(wn) + \sum_{\substack{wi\neq 11\ldots 1,\\ wm\preccurlyeq wi}} \Pi_2(wn, wi+1) \begin{bmatrix} 1 \\ 0 \end{bmatrix}(wn) \equiv 1 \pmod 2.$$

Доказательство. Если wn начинается с 1, то $\begin{bmatrix} 1 \\ 0 \end{bmatrix}(wn) = 0$, а значит, сумма равна нулю. Остается лишь $\begin{bmatrix} 0 \\ 1 \end{bmatrix}(wn) = 1$, что и требуется доказать.

Пусть теперь $wn = 0w_2$. Если $wm = 1w_1$, то получаем $\overline{w_1} < \overline{w_2}$, а выражение преобразуется к виду

$$\begin{bmatrix} 0 \\ 1 \end{bmatrix}(0w_2) + \sum_{\substack{wi\neq 11\ldots 1,\\ w_1\preccurlyeq wi}} \Pi_2(0w_2, 1(wi+1)) \begin{bmatrix} 1 \\ 0 \end{bmatrix}(0w_2) \equiv 0 + \sum_{\substack{wi\neq 11\ldots 1,\\ w_1\preccurlyeq wi}} \Pi_1(w_2, wi+1)\cdot 1 \pmod 2.$$

Применяя лемму 8, получаем требуемое.

Если $wm = 0w_1$, то выражение преобразуется к виду

$$\begin{bmatrix} 0 \\ 1 \end{bmatrix}(0w_2) + \sum_{\substack{wi\neq 11\ldots 1\\ 0w_1\preccurlyeq wi}} \Pi_2(0w_2, wi+1) \begin{bmatrix} 1 \\ 0 \end{bmatrix}(0w_2) =$$

$$= 0 + \Pi_2(0w_2, 100\ldots 0) + \sum_{\substack{wi\neq 11\ldots 1,\\ w_1\preccurlyeq wi}} \Pi_2(0w_2, 0(wi+1))\cdot 1 + \sum_{\substack{wi\neq 11\ldots 1,\\ w_1\preccurlyeq wi}} \Pi_2(0w_2, 1(wi+1))\cdot 1 \equiv$$

$$\equiv \Pi_1(w_2, 00\ldots 0) + \sum_{\substack{wi\neq 11\ldots 1,\\ w_1\preccurlyeq wi}} \Pi_1(w_2, wi+1) + \sum_{\substack{wi\neq 11\ldots 1,\\ w_1\preccurlyeq wi}} \Pi_1(w_2, wi+1) \equiv 1 \pmod 2.$$

Утверждение доказано. ■

Лемма 12. Обозначим 0^t последовательность нулей длины t. При $wm = 0^t 0w_1$, $wn = 0^t 1w_2$, $\overline{w_1} - 2^{|w_1|-1} < \overline{w_2}$, $t \geqslant 0$ имеет место

$$1 + \sum_{\substack{wi\neq 11\ldots 1,\\ wm\preccurlyeq wi}} \Pi_2(wn, wi+1) \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}(wn, wi+1) \equiv 0 \pmod 4.$$

Доказательство. Докажем утверждение индукцией по t. Пусть $t = 0$, откуда $wm = 0w_1$, $wn = 1w_2$. Преобразуем выражение:

$$1 + \sum_{\substack{wi\neq 11\ldots 1,\\ 0w_1\preccurlyeq wi}} \Pi_2(1w_2, wi+1) \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}(1w_2, wi+1) =$$

$$= 1 + \Pi_2(1w_2, 100\ldots 0) + \sum_{\substack{wi\neq 11\ldots 1,\\ w_1\preccurlyeq wi}} \Pi_2(1w_2, 0(wi+1))\cdot 1 + \sum_{\substack{wi\neq 11\ldots 1,\\ w_1\preccurlyeq wi}} \Pi_2(1w_2, 1(wi+1))\cdot 1 =$$

$$= 1 + \begin{bmatrix} 1 \\ 3 \end{bmatrix}(w_2) + \sum_{\substack{wi \neq 11\ldots1, \\ w_1 \preccurlyeq wi}} \Pi_2(w_2, wi+1) \begin{bmatrix} 1 & 2 \\ 1 & 3 \end{bmatrix}(w_2, wi+1) +$$

$$+ \sum_{\substack{wi \neq 11\ldots1, \\ w_1 \preccurlyeq wi}} \Pi_2(w_2, wi+1) \begin{bmatrix} 1 & 0 \\ 3 & 1 \end{bmatrix} \equiv \begin{bmatrix} 2 \\ 0 \end{bmatrix}(w_2) + \sum_{\substack{wi \neq 11\ldots1, \\ w_1 \preccurlyeq wi}} \Pi_2(w_2, wi+1) \begin{bmatrix} 2 & 2 \\ 0 & 0 \end{bmatrix}(w_2, wi+1).$$

Сокращая на 2 и прибавляя к обеим частям 1, преобразуем утверждение к виду

$$\begin{bmatrix} 0 \\ 1 \end{bmatrix}(w_2) + \sum_{\substack{wi \neq 11\ldots1, \\ w_1 \preccurlyeq wi}} \Pi_2(w_2, wi+1) \begin{bmatrix} 1 \\ 0 \end{bmatrix}(w_2) \equiv 1 \pmod 2,$$

который совпадает с леммой 11. База индукции доказана.

Пусть утверждение доказано для $t \leqslant k$, докажем его для $t = k+1$. Тогда для слов wm и wn получаем представление $wm = 00^k0w_1$, $n = 00^k1w_2$. Преобразуем выражение:

$$1 + \sum_{\substack{wi \neq 11\ldots1, \\ 00^k0w_1 \preccurlyeq wi}} \Pi_2(00^k1w_2, wi+1) \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}(00^k1w_2, wi+1) =$$

$$= 1 + \Pi_2(00^k1w_2, 100\ldots0) \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}(00^k1w_2, 100\ldots0) +$$

$$+ \sum_{\substack{wi \neq 11\ldots1, \\ 0^k0w_1 \preccurlyeq wi}} \Pi_2(00^k1w_2, 0(wi+1)) \cdot 1 + \sum_{\substack{wi \neq 11\ldots1, \\ 0^k0w_1 \preccurlyeq wi}} \Pi_2(00^k1w_2, 1(wi+1)) \cdot 0 =$$

$$= 1 + \Pi_2(00^k1w_2, 100\ldots0) \cdot 0 + \sum_{\substack{wi \neq 11\ldots1, \\ 0^k0w_1 \preccurlyeq wi}} \Pi_2(0^k1w_2, wi+1) \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}(0^t1w_2, wi+1).$$

Последнее выражение сравнимо с 2 по модулю 4 по предположению индукции. ■

Лемма 13. При $wm = 0^t0w_1$, $wn = 0^t1w_2$, $\overline{w_1} - 2^{|w_1|-1} < \overline{w_2}$, $t \geqslant 0$ выполнено

$$\begin{bmatrix} 1 \\ 3 \end{bmatrix}(wn) + \sum_{\substack{wi \neq 11\ldots1, \\ wm \preccurlyeq wi}} \Pi_2(wn, wi+1) \begin{bmatrix} 3 & 2 \\ 1 & 1 \end{bmatrix}(wn, wi+1) \equiv 2 \pmod 4. \qquad (4)$$

Доказательство. Пусть $t = 0$, откуда $wm = 0w_1$, $wn = 1w_2$. Преобразуем выражение:

$$\begin{bmatrix} 1 \\ 3 \end{bmatrix}(1w_2) + \sum_{\substack{wi \neq 11\ldots1, \\ 0w_1 \preccurlyeq wi}} \Pi_2(1w_2, wi+1) \begin{bmatrix} 3 & 2 \\ 1 & 1 \end{bmatrix}(1w_2, wi+1) =$$

$$= 3 + \Pi_2(1w_2, 100\ldots0) \cdot 1 + \sum_{\substack{wi \neq 11\ldots1, \\ w_1 \preccurlyeq wi}} \Pi_2(1w_2, 0(wi+1)) \cdot 1 + \sum_{\substack{wi \neq 11\ldots1, \\ w_1 \preccurlyeq wi}} \Pi_2(1w_2, 1(wi+1)) \cdot 1 =$$

$$= 3 + \begin{bmatrix} 1 \\ 3 \end{bmatrix}(w_2) + \sum_{\substack{wi \neq 11\ldots1, \\ w_1 \preccurlyeq wi}} (\Pi_2(w_2, wi+1) \begin{bmatrix} 1 & 2 \\ 1 & 3 \end{bmatrix}(w_2, wi+1) +$$

$$+ \Pi_2(w_2, wi+1) \begin{bmatrix} 1 & 0 \\ 3 & 1 \end{bmatrix}(w_2, wi+1)) \equiv$$

$$\equiv \begin{bmatrix} 0 \\ 2 \end{bmatrix}(w_2) + \sum_{\substack{wi \neq 11\ldots1, \\ w_1 \preccurlyeq wi}} \Pi_2(w_2, wi+1) \begin{bmatrix} 2 & 2 \\ 0 & 0 \end{bmatrix}(w_2, wi+1) \pmod 4.$$

Сокращая на 2, преобразуем утверждение к виду

$$\left[\begin{array}{c}0\\1\end{array}\right](w_2)+\sum_{\substack{wi\neq 11\ldots1,\\ w_1\preccurlyeq wi}}\Pi_2(w_2,wi+1)\left[\begin{array}{c}1\\0\end{array}\right](w_2)\equiv 1 \pmod 2,$$

который совпадает с леммой 11.

Пусть теперь $t>0$. Тогда, уменьшая t на единицу, получаем $wm=00^t0w_1$, $n=00^t1w_2$. Преобразуем левую часть (4):

$$\left[\begin{array}{c}1\\3\end{array}\right](00^t1w_2)+\sum_{\substack{wi\neq 11\ldots1,\\ 00^t0w_1\preccurlyeq wi}}\Pi_2(00^t1w_2,wi+1)\left[\begin{array}{cc}3&2\\1&1\end{array}\right](00^t1w_2,wi+1)=$$

$$=1+\Pi_2(00^t1w_2,100\ldots0)\left[\begin{array}{cc}3&2\\1&1\end{array}\right](00^t1w_2,100\ldots0)+$$

$$+\sum_{\substack{wi\neq 11\ldots1,\\ 0^t0w_1\preccurlyeq wi}}\Pi_2(00^t1w_2,0(wi+1))\left[\begin{array}{cc}3&2\\1&1\end{array}\right](00^t1w_2,0(wi+1))+$$

$$+\sum_{\substack{wi\neq 11\ldots1,\\ 0^t0w_1\preccurlyeq wi}}\Pi_2(00^t1w_2,1(wi+1))\left[\begin{array}{cc}3&2\\1&1\end{array}\right](00^t1w_2,1(wi+1))=$$

$$=1+\left[\begin{array}{c}3\\1\end{array}\right](0^t1w_2)\cdot 2+\sum_{\substack{wi\neq 11\ldots1,\\ 0^t0w_1\preccurlyeq wi}}\Pi_2(0^t1w_2,wi+1)\left[\begin{array}{cc}1&0\\1&1\end{array}\right](0^t1w_2,wi+1)\cdot 3+$$

$$+\sum_{\substack{wi\neq 11\ldots1,\\ 0^t0w_1\preccurlyeq wi}}\Pi_2(0^t1w_2,wi+1)\left[\begin{array}{cc}3&2\\1&1\end{array}\right](0^t1w_2,wi+1)\cdot 2\equiv$$

$$\equiv 1+2+\sum_{\substack{wi\neq 11\ldots1,\\ 0^t0w_1\preccurlyeq wi}}\Pi_2(0^t1w_2,wi+1)\left[\begin{array}{cc}1&0\\1&1\end{array}\right](0^t1w_2,wi+1).$$

В результате утверждение преобразуется к виду

$$1+\sum_{\substack{wi\neq 11\ldots1,\\ 0^t0w_1\preccurlyeq wi}}\Pi_2(0^t1w_2,wi+1)\left[\begin{array}{cc}1&0\\1&1\end{array}\right](0^t1w_2,wi+1)\equiv 0 \pmod 4,$$

который совпадает с леммой 12. ■

Лемма 14. Пусть $\overline{wm}-2^{|wm|-1}<\overline{wn}\leqslant\overline{wm}$. Тогда

$$\left[\begin{array}{c}0\\1\end{array}\right](wn)+\sum_{\substack{wi\neq 11\ldots1,\\ wm\preccurlyeq wi}}\Pi_2(wn,wi+1)\left[\begin{array}{cc}0&1\\1&1\end{array}\right](wn,wi+1)\equiv 1 \pmod 2.$$

Доказательство. Пусть $wm=1w_1$, $wn=1w_2$, тогда $\overline{w_2}\leqslant\overline{w_1}$. Преобразуем выражение:

$$\left[\begin{array}{c}0\\1\end{array}\right](1w_2)+\sum_{\substack{wi\neq 11\ldots1,\\ 1w_1\preccurlyeq wi}}\Pi_2(1w_2,wi+1)\left[\begin{array}{cc}0&1\\1&1\end{array}\right](1w_2,wi+1)=$$

$$=1+\sum_{\substack{wi\neq 11\ldots1,\\ w_1\preccurlyeq wi}}\Pi_2(1w_2,1(wi+1))\left[\begin{array}{cc}0&1\\1&1\end{array}\right](1w_2,1(wi+1))\equiv 1+\sum_{\substack{wi\neq 11\ldots1,\\ w_1\preccurlyeq wi}}\Pi_1(w_2,wi+1)\cdot 1.$$

Сумма равна 0 по модулю 2 в силу леммы 10, а значит, все выражение сравнимо с 1 по модулю 2, что и требовалось доказать.

Пусть $wm = 1w_1$, $wn = 0w_2$, тогда $\overline{w_1} < \overline{w_2}$. Преобразуем выражение:

$$\begin{bmatrix} 0 \\ 1 \end{bmatrix}(0w_2) + \sum_{\substack{wi \neq 11\ldots1, \\ 1w_1 \preccurlyeq wi}} \Pi_2(0w_2, wi+1) \begin{bmatrix} 0 & 1 \\ 1 & 1 \end{bmatrix}(0w_2, wi+1) =$$

$$= 0 + \sum_{\substack{wi \neq 11\ldots1, \\ w_1 \preccurlyeq wi}} \Pi_2(0w_2, 1(wi+1)) \begin{bmatrix} 0 & 1 \\ 1 & 1 \end{bmatrix}(0w_2, 1(wi+1)) \equiv \sum_{\substack{wi \neq 11\ldots1, \\ w_1 \preccurlyeq wi}} \Pi_1(w_2, wi+1) \cdot 1.$$

По лемме 8 это выражение сравнимо с 1 по модулю 2, что и требовалось доказать.

Пусть $wm = 0w_1$, $wn = 0w_2$, тогда $\overline{w_2} \leqslant \overline{w_1}$. Преобразуем выражение:

$$\begin{bmatrix} 0 \\ 1 \end{bmatrix}(0w_2) + \sum_{\substack{wi \neq 11\ldots1, \\ 0w_1 \preccurlyeq wi}} \Pi_2(0w_2, wi+1) \begin{bmatrix} 0 & 1 \\ 1 & 1 \end{bmatrix}(0w_2, wi+1) =$$

$$= 0 + \Pi_2(0w_2, 100\ldots0) \begin{bmatrix} 0 & 1 \\ 1 & 1 \end{bmatrix}(0w_2, 100\ldots0) +$$

$$+ \sum_{\substack{wi \neq 11\ldots1, \\ w_1 \preccurlyeq wi}} \Pi_2(0w_2, 0(wi+1)) \begin{bmatrix} 0 & 1 \\ 1 & 1 \end{bmatrix}(0w_2, 0(wi+1)) +$$

$$+ \sum_{\substack{wi \neq 11\ldots1, \\ w_1 \preccurlyeq wi}} \Pi_2(0w_2, 1(wi+1)) \begin{bmatrix} 0 & 1 \\ 1 & 1 \end{bmatrix}(0w_2, 1(wi+1)) \equiv$$

$$\equiv 0 + \Pi_1(w_2, 00\ldots0) \cdot 1 + \sum_{\substack{wi \neq 11\ldots1, \\ w_1 \preccurlyeq wi}} \Pi_1(w_2, wi+1) \cdot 0 + \sum_{\substack{wi \neq 11\ldots1, \\ w_1 \preccurlyeq wi}} \Pi_1(w_2, wi+1) \cdot 1 =$$

$$= 1 + \sum_{\substack{wi \neq 11\ldots1, \\ w_1 \preccurlyeq wi}} \Pi_1(w_2, wi+1).$$

Рассуждая так же, как и в первом случае, получаем требуемое. ■

Теорема 5. Пусть для некоторых непустых слов w_1 и w_2 выполнено $wm = 010^t0w_1$, $wn = 100^t1w_2$, $\overline{w_1} - 2^{|w_1|-1} < \overline{w_2}$, $t \geqslant 0$; или $wm = 01w_1$, $wn = 11w_2$, $\overline{w_1} - 2^{|w_1|-1} < \overline{w_2} \leqslant \overline{w_1}$. Тогда

$$1 + \sum_{\substack{wi \neq 11\ldots1, \\ wm \preccurlyeq wi}} \Pi_3(wn, wi+1) M_2^*(wn, wi+1) \equiv 4 \pmod 8.$$

Доказательство. Рассмотрим первый случай:

$$1 + \sum_{\substack{wi \neq 11\ldots1, \\ wm \preccurlyeq wi}} \Pi_3(wn, wi+1) M_2^*(wn, wi+1) =$$

$$= 1 + \sum_{\substack{wi \neq 11\ldots1, \\ 010^t0w_1 \preccurlyeq wi}} \Pi_3(100^t1w_2, wi+1) \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 2 & 1 & 0 \\ 1 & 3 & 3 & 1 \end{bmatrix}(100^t1w_2, wi+1) =$$

$$= 1 + \sum_{\substack{wi \neq 11\ldots1, \\ 010^t0w_1 \preccurlyeq wi}} \Pi_3(100^t1w_2, wi+1) \begin{bmatrix} 1 & 2 & 1 & 0 \end{bmatrix}(wi+1) = 1 + \Pi_3(100^t1w_2, 100\ldots0) \cdot 1 +$$

$$+ \sum_{\substack{wi \neq 11\ldots1, \\ 10^t0w_1 \preccurlyeq wi}} \Pi_3(100^t1w_2, 0(wi+1)) \begin{bmatrix} 1 & 2 \end{bmatrix}(wi+1) +$$

$$+\sum_{\substack{wi\neq 11\ldots1,\\ 10^t0w_1\preccurlyeq wi}}\Pi_3(100^t1w_2,1(wi+1))\,[\,1\;0\,]\,(wi+1)=1+\begin{bmatrix}1\\5\end{bmatrix}(0^t1w_2)+$$

$$+\sum_{\substack{wi\neq 11\ldots1,\\ 10^t0w_1\preccurlyeq wi}}\Pi_3(00^t1w_2,wi+1)\begin{bmatrix}1&4&3&2\\1&5&5&5\\1&6&5&4\\1&7&7&1\end{bmatrix}(00^t1w_2,wi+1)\,[\,1\;2\,]\,(wi+1)+$$

$$+\sum_{\substack{wi\neq 11\ldots1,\\ 10^t0w_1\preccurlyeq wi}}\Pi_3(00^t1w_2,wi+1)\begin{bmatrix}1&0&3&6\\5&1&1&1\\5&2&1&0\\1&7&7&1\end{bmatrix}(00^t1w_2,wi+1)\,[\,1\;0\,]\,(wi+1)\equiv$$

$$\equiv\begin{bmatrix}2\\6\end{bmatrix}(0^t1w_2)+\sum_{\substack{wi\neq 11\ldots1,\\ 10^t0w_1\preccurlyeq wi}}\Pi_3(00^t1w_2,wi+1)\begin{bmatrix}1&4&6&4\\1&5&2&2\\1&6&2&0\\1&7&6&2\end{bmatrix}(00^t1w_2,wi+1)+$$

$$+\sum_{\substack{wi\neq 11\ldots1,\\ 10^t0w_1\preccurlyeq wi}}\Pi_3(00^t1w_2,wi+1)\begin{bmatrix}1&0&0&0\\5&1&0&0\\5&2&0&0\\1&7&0&0\end{bmatrix}(00^t1w_2,wi+1)\equiv$$

$$\equiv\begin{bmatrix}2\\6\end{bmatrix}(0^t1w_2)+\sum_{\substack{wi\neq 11\ldots1,\\ 10^t0w_1\preccurlyeq wi}}\Pi_3(00^t1w_2,wi+1)\begin{bmatrix}2&4&6&4\\6&6&2&2\\6&0&2&0\\0&0&6&2\end{bmatrix}(00^t1w_2,wi+1)\pmod 8.$$

Сокращая на 2, получаем, что утверждение леммы сводится к виду

$$\begin{bmatrix}1\\3\end{bmatrix}(0^t1w_2)+\sum_{\substack{wi\neq 11\ldots1,\\ 10^t0w_1\preccurlyeq wi}}\Pi_3(00^t1w_2,wi+1)\begin{bmatrix}1&2&3&2\\3&3&1&1\\3&0&1&0\\0&0&3&1\end{bmatrix}(00^t1w_2,wi+1)\equiv 2\pmod 4.$$

Преобразуем дальше:

$$\begin{bmatrix}1\\3\end{bmatrix}(0^t1w_2)+\sum_{\substack{wi\neq 11\ldots1,\\ 10^t0w_1\preccurlyeq wi}}\Pi_3(00^t1w_2,wi+1)\begin{bmatrix}1&2&3&2\\3&3&1&1\\3&0&1&0\\0&0&3&1\end{bmatrix}(00^t1w_2,wi+1)=$$

$$=\begin{bmatrix}1\\3\end{bmatrix}(0^t1w_2)+\sum_{\substack{wi\neq 11\ldots1,\\ 0^t0w_1\preccurlyeq wi}}\Pi_3(00^t1w_2,1(wi+1))\begin{bmatrix}1&2&3&2\\3&3&1&1\\3&0&1&0\\0&0&3&1\end{bmatrix}(00^t1w_2,1(wi+1))\equiv$$

$$\equiv\begin{bmatrix}1\\3\end{bmatrix}(0^t1w_2)+\sum_{\substack{wi\neq 11\ldots1,\\ 0^t0w_1\preccurlyeq wi}}\Pi_2(0^t1w_2,wi+1)\begin{bmatrix}3&2\\1&1\end{bmatrix}(0^t1w_2,wi+1).$$

Последнее выражение сравнимо с 2 по модулю 4 по лемме 13.

Рассмотрим второй случай:

$$1+\sum_{\substack{wi\neq 11\ldots1,\\ wm\preccurlyeq wi}}\Pi_3(wn,wi+1)M_2^*(wn,wi+1)=$$

$$= 1 + \sum_{\substack{wi \neq 11\ldots1,\\ 01w_1 \preccurlyeq wi}} \Pi_3(11w_2, wi+1) \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 2 & 1 & 0 \\ 1 & 3 & 3 & 1 \end{bmatrix} (11w_2, wi+1) =$$

$$= 1 + \Pi_3(11w_2, 100\ldots0) \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 2 & 1 & 0 \\ 1 & 3 & 3 & 1 \end{bmatrix} (11w_2, 100\ldots0) +$$

$$+ \sum_{\substack{wi \neq 11\ldots1,\\ 1w_1 \preccurlyeq wi}} \Pi_3(11w_2, 0(wi+1)) \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 2 & 1 & 0 \\ 1 & 3 & 3 & 1 \end{bmatrix} (11w_2, 0(wi+1)) +$$

$$+ \sum_{\substack{wi \neq 11\ldots1,\\ 1w_1 \preccurlyeq wi}} \Pi_3(11w_2, 1(wi+1)) \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 2 & 1 & 0 \\ 1 & 3 & 3 & 1 \end{bmatrix} (11w_2, 1(wi+1)) = 1 + \begin{bmatrix} 5 \\ 1 \end{bmatrix} (w_2) \cdot 3 +$$

$$+ \sum_{\substack{wi \neq 11\ldots1,\\ 1w_1 \preccurlyeq wi}} \Pi_3(1w_2, wi+1) \begin{bmatrix} 1 & 4 & 3 & 2 \\ 1 & 5 & 5 & 5 \\ 1 & 6 & 5 & 4 \\ 1 & 7 & 7 & 1 \end{bmatrix} (1w_2, wi+1) \begin{bmatrix} 1 & 2 \\ 1 & 3 \end{bmatrix} (1w_2, wi+1) +$$

$$+ \sum_{\substack{wi \neq 11\ldots1,\\ 1w_1 \preccurlyeq wi}} \Pi_3(1w_2, wi+1) \begin{bmatrix} 1 & 0 & 3 & 6 \\ 5 & 1 & 1 & 1 \\ 5 & 2 & 1 & 0 \\ 1 & 7 & 7 & 1 \end{bmatrix} (1w_2, wi+1) \begin{bmatrix} 1 & 0 \\ 3 & 1 \end{bmatrix} (1w_2, wi+1) \equiv$$

$$\equiv \begin{bmatrix} 0 \\ 4 \end{bmatrix} (w_2) + \sum_{\substack{wi \neq 11\ldots1,\\ 1w_1 \preccurlyeq wi}} \Pi_3(1w_2, wi+1) \begin{bmatrix} 1 & 4 & 6 & 4 \\ 1 & 5 & 2 & 2 \\ 1 & 6 & 7 & 4 \\ 1 & 7 & 5 & 3 \end{bmatrix} (1w_2, wi+1) +$$

$$+ \sum_{\substack{wi \neq 11\ldots1,\\ 1w_1 \preccurlyeq wi}} \Pi_3(1w_2, wi+1) \begin{bmatrix} 1 & 0 & 0 & 0 \\ 5 & 1 & 0 & 0 \\ 7 & 6 & 1 & 0 \\ 3 & 5 & 7 & 1 \end{bmatrix} (1w_2, wi+1) \equiv$$

$$\equiv \begin{bmatrix} 0 \\ 4 \end{bmatrix} (w_2) + \sum_{\substack{wi \neq 11\ldots1,\\ 1w_1 \preccurlyeq wi}} \Pi_3(1w_2, wi+1) \begin{bmatrix} 2 & 4 & 6 & 4 \\ 6 & 6 & 2 & 2 \\ 0 & 4 & 0 & 4 \\ 4 & 4 & 4 & 4 \end{bmatrix} (1w_2, wi+1) \pmod 8.$$

Сокращая на 2, получаем, что утверждение леммы сводится к виду

$$\begin{bmatrix} 0 \\ 2 \end{bmatrix} (w_2) + \sum_{\substack{wi \neq 11\ldots1,\\ 1w_1 \preccurlyeq wi}} \Pi_3(1w_2, wi+1) \begin{bmatrix} 1 & 2 & 3 & 2 \\ 3 & 3 & 1 & 1 \\ 0 & 2 & 0 & 2 \\ 2 & 2 & 2 & 2 \end{bmatrix} (1w_2, wi+1) \equiv 2 \pmod 4.$$

Преобразуем дальше:

$$\left[\begin{array}{c}0\\2\end{array}\right](w_2)+\sum_{\substack{wi\neq 11\ldots1,\\ 1w_1\preccurlyeq wi}}\Pi_3(1w_2,wi+1)\left[\begin{array}{cccc}1&2&3&2\\3&3&1&1\\0&2&0&2\\2&2&2&2\end{array}\right](1w_2,wi+1)=$$

$$=\left[\begin{array}{c}0\\2\end{array}\right](w_2)+\sum_{\substack{wi\neq 11\ldots1,\\ w_1\preccurlyeq wi}}\Pi_3(1w_2,1(wi+1))\left[\begin{array}{cccc}1&2&3&2\\3&3&1&1\\0&2&0&2\\2&2&2&2\end{array}\right](1w_2,1(wi+1))\equiv$$

$$\equiv\left[\begin{array}{c}0\\2\end{array}\right](w_2)+\sum_{\substack{wi\neq 11\ldots1,\\ w_1\preccurlyeq wi}}\Pi_2(w_2,wi+1)\left[\begin{array}{cc}0&2\\2&2\end{array}\right](w_2,wi+1).$$

Сокращая на 2, получаем, что утверждение леммы сводится к виду

$$\left[\begin{array}{c}0\\1\end{array}\right](w_2)+\sum_{\substack{wi\neq 11\ldots1,\\ w_1\preccurlyeq wi}}\Pi_2(w_2,wi+1)\left[\begin{array}{cc}0&1\\1&1\end{array}\right](w_2,wi+1)\equiv 1 \pmod 2.$$

Это выражение в точности совпадает с леммой 14. ■

Лемма 15. Пусть для непустых слов $\overline{wn}\leqslant\overline{wm}$. Тогда

$$\sum_{\substack{wi\neq 11\ldots1,\\ wm\preccurlyeq wi}}\Pi_2(wn,wi+1)\left[\begin{array}{cc}1&0\\*&1\end{array}\right](wn,wi+1)\equiv 0 \pmod 4.$$

Здесь $*$ обозначает произвольное фиксированное число.

Доказательство. Докажем утвреждение индукцией по длине слов wm и wn. Если wm и wn имеют длину 1, то в сумме есть слагаемые только при $wm=0$, а значит, и $wn=0$. В этом случае единственное слагаемое равно

$$\Pi_2(0,1)\left[\begin{array}{cc}1&0\\*&1\end{array}\right](0,1)=1\cdot 0=0.$$

База индукции доказана. Докажем переход. Пусть утверждение доказано для длин wm и wn не больше k. Докажем его для $|wm|=|wn|=k+1$.

Пусть $wm=1w_1$, $wn=1w_2$, $\overline{w_2}\leqslant\overline{w_1}$. Преобразуем выражение:

$$\sum_{\substack{wi\neq 11\ldots1,\\ 1w_1\preccurlyeq wi}}\Pi_2(1w_2,wi+1)\left[\begin{array}{cc}1&0\\*&1\end{array}\right](1w_2,wi+1)=$$

$$=\sum_{\substack{wi\neq 11\ldots1,\\ w_1\preccurlyeq wi}}\Pi_2(1w_2,1(wi+1))\left[\begin{array}{cc}1&0\\*&1\end{array}\right](1w_2,1(wi+1))=$$

$$=\sum_{\substack{wi\neq 11\ldots1,\\ w_1\preccurlyeq wi}}\Pi_2(w_2,wi+1)\left[\begin{array}{cc}1&0\\3&1\end{array}\right](w_2,wi+1)\cdot 1\equiv 0 \pmod 4.$$

Здесь последнее сравнение следует из предположения индукции.

Пусть $wm = 1w_1$, $wn = 0w_2$. Получаем

$$\sum_{\substack{wi\neq 11\ldots 1,\\ 1w_1\preccurlyeq wi}} \Pi_2(0w_2, wi+1)\begin{bmatrix} 1 & 0 \\ * & 1 \end{bmatrix}(0w_2, wi+1) =$$

$$= \sum_{\substack{wi\neq 11\ldots 1,\\ w_1\preccurlyeq wi}} \Pi_2(0w_2, 1(wi+1))\begin{bmatrix} 1 & 0 \\ * & 1 \end{bmatrix}(0w_2, 1(wi+1)) = \sum_{\substack{wi\neq 11\ldots 1,\\ w_1\preccurlyeq wi}} \Pi_2(0w_2, 1(wi+1))\cdot 0 = 0.$$

Пусть $wm = 0w_1$, $wn = 0w_2$, $\overline{w_2} \leqslant \overline{w_1}$. Имеем

$$\sum_{\substack{wi\neq 11\ldots 1,\\ 0w_1\preccurlyeq wi}} \Pi_2(0w_2, wi+1)\begin{bmatrix} 1 & 0 \\ * & 1 \end{bmatrix}(0w_2, wi+1) = \Pi_2(0w_2, 100\ldots 0)\begin{bmatrix} 1 & 0 \\ * & 1 \end{bmatrix}(0w_2, 100\ldots 0)+$$

$$+ \sum_{\substack{wi\neq 11\ldots 1,\\ w_1\preccurlyeq wi}} \Pi_2(0w_2, 0(wi+1))\begin{bmatrix} 1 & 0 \\ * & 1 \end{bmatrix}(0w_2, 0(wi+1))+$$

$$+ \sum_{\substack{wi\neq 11\ldots 1,\\ w_1\preccurlyeq wi}} \Pi_2(0w_2, 1(wi+1))\begin{bmatrix} 1 & 0 \\ * & 1 \end{bmatrix}(0w_2, 1(wi+1)) =$$

$$= \Pi_2(0w_2, 100\ldots 0)\cdot 0 + \sum_{\substack{wi\neq 11\ldots 1,\\ w_1\preccurlyeq wi}} \Pi_2(0w_2, 0(wi+1))\cdot 1 + \sum_{\substack{wi\neq 11\ldots 1,\\ w_1\preccurlyeq wi}} \Pi_2(0w_2, 1(wi+1))\cdot 0 =$$

$$= \sum_{\substack{wi\neq 11\ldots 1,\\ w_1\preccurlyeq wi}} \Pi_2(w_2, wi+1)\begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}(w_2, wi+1) \equiv 0 \pmod 4,$$

где последнее сравнение следует из предположения индукции. Переход доказан. ■

Лемма 16. Пусть для непустых слов $\overline{wm} - 2^{|wm|-1} < \overline{wn} \leqslant \overline{wm}$. Тогда

$$\begin{bmatrix} 0 \\ 2 \end{bmatrix}(wn) + \sum_{\substack{wi\neq 11\ldots 1,\\ wm\preccurlyeq wi}} \Pi_2(wn, wi+1)\begin{bmatrix} 3 & 2 \\ 1 & 1 \end{bmatrix}(wn, wi+1) \equiv 2 \pmod 4.$$

Доказательство. Пусть $wm = 1w_1$, $wn = 1w_2$. Тогда $\overline{w_2} \leqslant \overline{w_1}$. Преобразуем выражение:

$$\begin{bmatrix} 0 \\ 2 \end{bmatrix}(1w_2) + \sum_{\substack{wi\neq 11\ldots 1,\\ 1w_1\preccurlyeq wi}} \Pi_2(1w_2, wi+1)\begin{bmatrix} 3 & 2 \\ 1 & 1 \end{bmatrix}(1w_2, wi+1) =$$

$$= 2 + \sum_{\substack{wi\neq 11\ldots 1,\\ w_1\preccurlyeq wi}} \Pi_2(1w_2, 1(wi+1))\begin{bmatrix} 3 & 2 \\ 1 & 1 \end{bmatrix}(1w_2, 1(wi+1)) =$$

$$= 2 + \sum_{\substack{wi\neq 11\ldots 1,\\ w_1\preccurlyeq wi}} \Pi_2(w_2, wi+1)\begin{bmatrix} 1 & 0 \\ 3 & 1 \end{bmatrix}(w_2, wi+1)\cdot 1.$$

Применяя лемму 15, получаем требуемое.

Пусть $wm = 1w_1$, $wn = 0w_2$. Тогда $\overline{w_2} > \overline{w_1}$. Имеем

$$\begin{bmatrix} 0 \\ 2 \end{bmatrix}(0w_2) + \sum_{\substack{wi\neq 11\ldots 1,\\ 1w_1\preccurlyeq wi}} \Pi_2(0w_2, wi+1)\begin{bmatrix} 3 & 2 \\ 1 & 1 \end{bmatrix}(0w_2, wi+1) =$$

$$= 0 + \sum_{\substack{wi\neq 11\ldots 1,\\ w_1\preccurlyeq wi}} \Pi_2(0w_2, 1(wi+1))\begin{bmatrix} 3 & 2 \\ 1 & 1 \end{bmatrix}(0w_2, 1(wi+1)) = \sum_{\substack{wi\neq 11\ldots 1,\\ w_1\preccurlyeq wi}} \Pi_2(0w_2, 1(wi+1))\cdot 2.$$

Сокращая на 2, получаем, что утверждение преобразуется к виду

$$\sum_{\substack{wi\neq 11\ldots 1,\\ w_1\preccurlyeq wi}} \Pi_2(0w_2, 1(wi+1)) \equiv 1 \pmod 2,$$

но $$\sum_{\substack{wi\neq 11\ldots 1,\\ w_1\preccurlyeq wi}} \Pi_2(0w_2, 1(wi+1)) \equiv \sum_{\substack{wi\neq 11\ldots 1,\\ w_1\preccurlyeq wi}} \Pi_1(w_2, wi+1) \pmod 2.$$

Утверждение теперь следует из леммы 8.

Пусть $wm = 0w_1$, $wn = 0w_2$. Тогда $\overline{w_2} \leqslant \overline{w_1}$. Преобразуем выражение:

$$\left[\begin{array}{c}0\\2\end{array}\right](0w_2) + \sum_{\substack{wi\neq 11\ldots 1,\\ 0w_1\preccurlyeq wi}} \Pi_2(0w_2, wi+1) \left[\begin{array}{cc}3&2\\1&1\end{array}\right](0w_2, wi+1) =$$

$$= 0 + \Pi_2(0w_2, 100\ldots 0) \left[\begin{array}{cc}3&2\\1&1\end{array}\right](0w_2, 100\ldots 0) +$$

$$+ \sum_{\substack{wi\neq 11\ldots 1,\\ w_1\preccurlyeq wi}} \Pi_2(0w_2, 0(wi+1)) \left[\begin{array}{cc}3&2\\1&1\end{array}\right](0w_2, 0(wi+1)) +$$

$$+ \sum_{\substack{wi\neq 11\ldots 1,\\ w_1\preccurlyeq wi}} \Pi_2(0w_2, 1(wi+1)) \left[\begin{array}{cc}3&2\\1&1\end{array}\right](0w_2, 1(wi+1)) = \left[\begin{array}{c}3\\1\end{array}\right](w_2)\cdot 2 +$$

$$+ \sum_{\substack{wi\neq 11\ldots 1,\\ w_1\preccurlyeq wi}} \Pi_2(w_2, wi+1) \left[\begin{array}{cc}1&0\\1&1\end{array}\right](w_2, wi+1)\cdot 3 + \sum_{\substack{wi\neq 11\ldots 1,\\ w_1\preccurlyeq wi}} \Pi_2(0w_2, 1(wi+1))\cdot 2.$$

Заметим, что вторая сумма равна 0 по модулю 4 по лемме 15. Сокращая оставшиеся члены на 2, получаем

$$\left[\begin{array}{c}3\\1\end{array}\right](w_2) + \sum_{\substack{wi\neq 11\ldots 1,\\ w_1\preccurlyeq wi}} \Pi_2(0w_2, 1(wi+1)) \equiv 1 \pmod 2,$$

но $$\left[\begin{array}{c}3\\1\end{array}\right](w_2) + \sum_{\substack{wi\neq 11\ldots 1,\\ w_1\preccurlyeq wi}} \Pi_2(0w_2, 1(wi+1)) \equiv 1 + \sum_{\substack{wi\neq 11\ldots 1,\\ w_1\preccurlyeq wi}} \Pi_1(w_2, wi+1) \pmod 2.$$

По лемме 10 получаем требуемое. ■

Лемма 17. Пусть $\overline{wn} > \overline{wm}$. Тогда

$$\left[\begin{array}{c}1\\0\end{array}\right](wn) + \sum_{\substack{wi\neq 11\ldots 1,\\ wm\preccurlyeq wi}} \Pi_2(wn, wi+1) \left[\begin{array}{cc}0&1\\1&1\end{array}\right](wn, wi+1) \equiv 1 \pmod 2.$$

Доказательство. Если длина wn и wm равна 1, то $wm = 0$, $wn = 1$. В этом случае в сумме одно слагаемое, соответствующее $wi = 0$:

$$\Pi_2(1,1) \left[\begin{array}{cc}0&1\\1&1\end{array}\right](1,1) = 1\cdot 1 = 1.$$

Пусть теперь $|wn| = |wm| > 1$. Рассмотрим случай $wn = 1w_1$, $wm = 1w_2$, откуда $\overline{w_1} > \overline{w_2}$. Имеем

$$\left[\begin{array}{c}1\\0\end{array}\right](1w_1) + \sum_{\substack{wi\neq 11\ldots 1,\\ 1w_2\preccurlyeq wi}} \Pi_2(1w_1, wi+1) \left[\begin{array}{cc}0&1\\1&1\end{array}\right](1w_1, wi+1) =$$

$$= \sum_{\substack{wi\neq 11\ldots 1,\\ w_2\preccurlyeq wi}} \Pi_2(1w_1, 1(wi+1)) \left[\begin{array}{cc}0&1\\1&1\end{array}\right](1w_1, 1(wi+1)) \equiv \sum_{\substack{wi\neq 11\ldots 1,\\ w_2\preccurlyeq wi}} \Pi_1(w_1, wi+1)\cdot 1.$$

Применяя лемму 8, получаем требуемое.

Пусть $wn = 1w_1$, $wm = 0w_2$. Имеем

$$\left[\begin{array}{c} 1 \\ 0 \end{array}\right](1w_1) + \sum_{\substack{wi \neq 11\ldots1,\\ 0w_2 \preccurlyeq wi}} \Pi_2(1w_1, wi+1) \left[\begin{array}{cc} 0 & 1 \\ 1 & 1 \end{array}\right](1w_1, wi+1) =$$

$$= \sum_{\substack{wi \neq 11\ldots1,\\ 0w_2 \preccurlyeq wi}} \Pi_2(1w_1, wi+1)\,[\,1\;1\,](wi+1) = \sum_{\substack{wi \neq 11\ldots1,\\ 0w_2 \preccurlyeq wi}} \Pi_2(1w_1, wi+1) =$$

$$= \Pi_2(1w_1, 100\ldots0) + \sum_{\substack{wi \neq 11\ldots1,\\ w_2 \preccurlyeq wi}} \Pi_2(1w_1, 0(wi+1)) + \sum_{\substack{wi \neq 11\ldots1,\\ w_2 \preccurlyeq wi}} \Pi_2(1w_1, 1(wi+1)) \equiv$$

$$\equiv \Pi_1(w_1, 00\ldots0) + \sum_{\substack{wi \neq 11\ldots1,\\ w_2 \preccurlyeq wi}} \Pi_1(w_1, wi+1) + \sum_{\substack{wi \neq 11\ldots1,\\ w_2 \preccurlyeq wi}} \Pi_1(w_1, wi+1) \equiv 1 \pmod 2.$$

Пусть $wn = 0w_1$, $wm = 0w_2$, откуда $\overline{w_1} > \overline{w_2}$. Имеем

$$\left[\begin{array}{c} 1 \\ 0 \end{array}\right](0w_1) + \sum_{\substack{wi \neq 11\ldots1,\\ 0w_2 \preccurlyeq wi}} \Pi_2(0w_1, wi+1) \left[\begin{array}{cc} 0 & 1 \\ 1 & 1 \end{array}\right](0w_1, wi+1) =$$

$$= 1 + \sum_{\substack{wi \neq 11\ldots1,\\ 0w_2 \preccurlyeq wi}} \Pi_2(0w_1, wi+1)\,[\,0\;1\,](wi+1) =$$

$$= 1 + \Pi_2(0w_1, 100\ldots0)\cdot 1 + \sum_{\substack{wi \neq 11\ldots1,\\ w_2 \preccurlyeq wi}} \Pi_2(0w_1, 1(wi+1))\cdot 1 \equiv$$

$$\equiv 1 + \Pi_1(w_1, 00\ldots0) + \sum_{\substack{wi \neq 11\ldots1,\\ w_2 \preccurlyeq wi}} \Pi_1(w_1, wi+1) \equiv \sum_{\substack{wi \neq 11\ldots1,\\ w_2 \preccurlyeq wi}} \Pi_1(w_1, wi+1).$$

Применяя лемму 8, получаем требуемое. ■

Лемма 18. Пусть для $t \geqslant 0$ выполнено $wm = 0^t01w_1$, $wn = 0^t10w_2$, $\overline{w_1} - 2^{|w_1|-1} < \overline{w_2} \leqslant \overline{w_1}$, $|w_2| = |w_1| > 0$; или $wm = 0^t100w_1$, $wn = 0^t111w_2$, $\overline{w_2} > \overline{w_1}$. Тогда

$$7 + \textstyle\sum_{\substack{wi \neq 11\ldots1,\\ wm \preccurlyeq wi}} \Pi_3(wn, wi+1) M_3^*(wn, wi+1) \equiv 4 \pmod 8.$$

Доказательство. Докажем утверждение индукцией по t. Пусть $t = 0$ и $wm = 01w_1$, $wn = 10w_2$, $\overline{w_1} - 2^{|w_1|-1} < \overline{w_2} \leqslant \overline{w_1}$. Имеем

$$7 + \sum_{\substack{wi \neq 11\ldots1,\\ 01w_1 \preccurlyeq wi}} \Pi_3(10w_2, wi+1) \left[\begin{array}{cccc} 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 2 & 1 & 0 \\ 1 & 3 & 3 & 1 \end{array}\right](10w_2, wi+1) = 7 +$$

$$+ \sum_{\substack{wi \neq 11\ldots1,\\ 01w_1 \preccurlyeq wi}} \Pi_3(10w_2, wi+1)[\,1\;2\;1\;0\,](wi+1) = 7 + \Pi_3(10w_2, 100\ldots0)[\,1\;2\;1\;0\,](100\ldots0) +$$

$$+ \sum_{\substack{wi \neq 11\ldots1,\\ w_1 \preccurlyeq wi}} \Pi_3(10w_2, 01(wi+1))[\,1\;2\;1\;0\,](01(wi+1)) +$$

$$+ \sum_{\substack{wi \neq 11\ldots1,\\ w_1 \preccurlyeq wi}} \Pi_3(10w_2, 11(wi+1))[\,1\;2\;1\;0\,](11(wi+1)) =$$

$$= 7 + \left[\begin{array}{c} 1 \\ 5 \end{array}\right](w_2)\cdot 1 + \sum_{\substack{wi \neq 11\ldots1,\\ w_1 \preccurlyeq wi}} \Pi_3(10w_2, 01(wi+1))\cdot 2 + \sum_{\substack{wi \neq 11\ldots1,\\ w_1 \preccurlyeq wi}} \Pi_3(10w_2, 11(wi+1))\cdot 0 \equiv$$

$$\equiv \left[\begin{array}{c} 0 \\ 4 \end{array}\right](w_2) + \sum_{\substack{wi \neq 11\ldots1,\\ w_1 \preccurlyeq wi}} \Pi_2(0w_2, 1(wi+1))\cdot 2 \pmod 8.$$

Последний переход от Π_3 к Π_2 возможен благодаря тому, что все слагаемые в сумме умножены на 2. Сокращая на 2, сводим утверждение леммы к следующему:

$$\begin{bmatrix} 0 \\ 2 \end{bmatrix}(w_2) + \sum_{\substack{wi\neq 11\ldots1,\\ w_1\preccurlyeq wi}} \Pi_2(0w_2, 1(wi+1)) \equiv 2 \pmod 4.$$

Преобразуя левую часть, получим

$$\begin{bmatrix} 0 \\ 2 \end{bmatrix}(w_2) + \sum_{\substack{wi\neq 11\ldots1,\\ w_1\preccurlyeq wi}} \Pi_2(0w_2, 1(wi+1)) =$$

$$= \begin{bmatrix} 0 \\ 2 \end{bmatrix}(w_2) + \sum_{\substack{wi\neq 11\ldots1,\\ w_1\preccurlyeq wi}} \Pi_2(w_2, wi+1) \begin{bmatrix} 3 & 2 \\ 1 & 1 \end{bmatrix}(w_2, wi+1) \equiv 2 \pmod 4,$$

где последнее сравнение следует из леммы 16.

Пусть $t = 0$ и $wm = 100w_1$, $wn = 111w_2$, $\overline{w_2} > \overline{w_1}$. Имеем

$$7 + \sum_{\substack{wi\neq 11\ldots1,\\ 100w_1\preccurlyeq wi}} \Pi_3(111w_2, wi+1) M_3^*(111w_2, wi+1) =$$

$$= 7 + \sum_{\substack{wi\neq 11\ldots1,\\ 00w_1\preccurlyeq wi}} \Pi_3(111w_2, 1(wi+1)) \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 2 & 1 & 0 \\ 1 & 3 & 3 & 1 \end{bmatrix}(111w_2, 1(wi+1)) =$$

$$= 7 + \sum_{\substack{wi\neq 11\ldots1,\\ 00w_1\preccurlyeq wi}} \Pi_3(11w_2, wi+1) \begin{bmatrix} 1 & 0 & 3 & 6 \\ 5 & 1 & 1 & 1 \\ 5 & 2 & 1 & 0 \\ 1 & 7 & 7 & 1 \end{bmatrix}(11w_2, wi+1) \begin{bmatrix} 1 & 0 \\ 3 & 1 \end{bmatrix}(11w_2, wi+1) =$$

$$= 7 + \sum_{\substack{wi\neq 11\ldots1,\\ 00w_1\preccurlyeq wi}} \Pi_3(11w_2, wi+1) \begin{bmatrix} 1 & 0 & 0 & 0 \\ 5 & 1 & 0 & 0 \\ 7 & 6 & 1 & 0 \\ 3 & 5 & 7 & 1 \end{bmatrix}(11w_2, wi+1) =$$

$$= 7 + \Pi_3(11w_2, 100\ldots0) \begin{bmatrix} 1 & 0 & 0 & 0 \\ 5 & 1 & 0 & 0 \\ 7 & 6 & 1 & 0 \\ 3 & 5 & 7 & 1 \end{bmatrix}(11w_2, 100\ldots0) +$$

$$+ \sum_{\substack{wi\neq 11\ldots1,\\ 0w_1\preccurlyeq wi}} \Pi_3(11w_2, 0(wi+1)) \begin{bmatrix} 1 & 0 & 0 & 0 \\ 5 & 1 & 0 & 0 \\ 7 & 6 & 1 & 0 \\ 3 & 5 & 7 & 1 \end{bmatrix}(11w_2, 0(wi+1)) +$$

$$+ \sum_{\substack{wi\neq 11\ldots1,\\ 0w_1\preccurlyeq wi}} \Pi_3(11w_2, 1(wi+1)) \begin{bmatrix} 1 & 0 & 0 & 0 \\ 5 & 1 & 0 & 0 \\ 7 & 6 & 1 & 0 \\ 3 & 5 & 7 & 1 \end{bmatrix}(11w_2, 1(wi+1)) = 7 + \begin{bmatrix} 5 \\ 1 \end{bmatrix}(w_2) \cdot 7 +$$

$$+ \sum_{\substack{wi\neq 11\ldots1,\\ 0w_1\preccurlyeq wi}} \Pi_3(1w_2, wi+1) \begin{bmatrix} 1 & 4 & 3 & 2 \\ 1 & 5 & 5 & 5 \\ 1 & 6 & 5 & 4 \\ 1 & 7 & 7 & 1 \end{bmatrix}(1w_2, wi+1) \begin{bmatrix} 7 & 6 \\ 3 & 5 \end{bmatrix}(1w_2, wi+1) +$$

$$+\sum_{\substack{wi\neq 11\ldots 1,\\ 0w_1\preccurlyeq wi}}\Pi_3(1w_2,wi+1)\begin{bmatrix}1&0&3&6\\5&1&1&1\\5&2&1&0\\1&7&7&1\end{bmatrix}(1w_2,wi+1)\begin{bmatrix}1&0\\7&1\end{bmatrix}(1w_2,wi+1)\equiv$$

$$\equiv\begin{bmatrix}2\\6\end{bmatrix}(w_2)+\sum_{\substack{wi\neq 11\ldots 1,\\ 0w_1\preccurlyeq wi}}\Pi_3(1w_2,wi+1)\begin{bmatrix}7&4&2&4\\7&3&6&6\\3&2&1&4\\3&5&3&5\end{bmatrix}(1w_2,wi+1)+$$

$$+\sum_{\substack{wi\neq 11\ldots 1,\\ 0w_1\preccurlyeq wi}}\Pi_3(1w_2,wi+1)\begin{bmatrix}1&0&0&0\\5&1&0&0\\3&6&1&0\\7&1&7&1\end{bmatrix}(1w_2,wi+1)\equiv$$

$$\equiv\begin{bmatrix}2\\6\end{bmatrix}(w_2)+\sum_{\substack{wi\neq 11\ldots 1,\\ 0w_1\preccurlyeq wi}}\Pi_3(1w_2,wi+1)\begin{bmatrix}0&4&2&4\\4&4&6&6\\6&0&2&4\\2&6&2&6\end{bmatrix}(1w_2,wi+1)\pmod 8.$$

Сокращая на 2, сводим утверждение к виду

$$\begin{bmatrix}1\\3\end{bmatrix}(w_2)+\sum_{\substack{wi\neq 11\ldots 1,\\ 0w_1\preccurlyeq wi}}\Pi_3(1w_2,wi+1)\begin{bmatrix}0&2&1&2\\2&2&3&3\\3&0&1&2\\1&3&1&3\end{bmatrix}(1w_2,wi+1)\equiv 2\pmod 4.$$

Продолжая вычисления, получаем

$$\begin{bmatrix}1\\3\end{bmatrix}(w_2)+\sum_{\substack{wi\neq 11\ldots 1,\\ 0w_1\preccurlyeq wi}}\Pi_3(1w_2,wi+1)\begin{bmatrix}0&2&1&2\\2&2&3&3\\3&0&1&2\\1&3&1&3\end{bmatrix}(1w_2,wi+1)=$$

$$=\begin{bmatrix}1\\3\end{bmatrix}(w_2)+\Pi_3(1w_2,100\ldots 0)\begin{bmatrix}0&2&1&2\\2&2&3&3\\3&0&1&2\\1&3&1&3\end{bmatrix}(1w_2,100\ldots 0)+$$

$$+\sum_{\substack{wi\neq 11\ldots 1,\\ w_1\preccurlyeq wi}}\Pi_3(1w_2,0(wi+1))\begin{bmatrix}0&2&1&2\\2&2&3&3\\3&0&1&2\\1&3&1&3\end{bmatrix}(1w_2,0(wi+1))+$$

$$+\sum_{\substack{wi\neq 11\ldots 1,\\ w_1\preccurlyeq wi}}\Pi_3(1w_2,1(wi+1))\begin{bmatrix}0&2&1&2\\2&2&3&3\\3&0&1&2\\1&3&1&3\end{bmatrix}(1w_2,1(wi+1))\equiv$$

$$\equiv\begin{bmatrix}1\\3\end{bmatrix}(w_2)+\Pi_2(w_2,00\ldots 0)\cdot 1+\sum_{\substack{wi\neq 11\ldots 1,\\ w_1\preccurlyeq wi}}\Pi_2(w_2,wi+1)\begin{bmatrix}3&0\\1&3\end{bmatrix}(w_2,wi+1)+$$

$$+\sum_{\substack{wi\neq 11\ldots 1,\\ w_1\preccurlyeq wi}}\Pi_2(w_2,wi+1)\begin{bmatrix}1&2\\1&3\end{bmatrix}(w_2,wi+1)\equiv$$

$$\equiv \begin{bmatrix} 1 \\ 3 \end{bmatrix} (w_2) + 1 + \sum_{\substack{wi \neq 11\ldots1, \\ w_1 \preccurlyeq wi}} \Pi_2(w_2, wi+1) \begin{bmatrix} 0 & 2 \\ 2 & 2 \end{bmatrix} (w_2, wi+1) \equiv$$

$$\equiv \begin{bmatrix} 2 \\ 0 \end{bmatrix} (w_2) + \sum_{\substack{wi \neq 11\ldots1, \\ w_1 \preccurlyeq wi}} \Pi_2(w_2, wi+1) \begin{bmatrix} 0 & 2 \\ 2 & 2 \end{bmatrix} (w_2, wi+1) \pmod 4.$$

Сократив на 2, сводим утверждение к виду

$$\begin{bmatrix} 1 \\ 0 \end{bmatrix} (w_2) + \sum_{\substack{wi \neq 11\ldots1, \\ w_1 \preccurlyeq wi}} \Pi_2(w_2, wi+1) \begin{bmatrix} 0 & 1 \\ 1 & 1 \end{bmatrix} (w_2, wi+1) \equiv 1 \pmod 2,$$

который совпадает с леммой 17. База индукции доказана.

Пусть утверждение доказано для $t \leqslant k$, докажем его для $t = k+1 > 0$. Заметим, что $w_1 = 0v_1$, $w_2 = 0v_2$, где v_1 и v_2 удовлетворяют условиям леммы. Имеем

$$7 + \sum_{\substack{wi \neq 11\ldots1, \\ 0v_1 \preccurlyeq wi}} \Pi_3(0v_2, wi+1) \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 2 & 1 & 0 \\ 1 & 3 & 3 & 1 \end{bmatrix} (0v_2, wi+1) =$$

$$= 7 + \Pi_3(0v_2, 100\ldots0) \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 2 & 1 & 0 \\ 1 & 3 & 3 & 1 \end{bmatrix} (0v_2, 100\ldots0) +$$

$$+ \sum_{\substack{wi \neq 11\ldots1, \\ v_1 \preccurlyeq wi}} \Pi_3(0v_2, 0(wi+1)) \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 2 & 1 & 0 \\ 1 & 3 & 3 & 1 \end{bmatrix} (0v_2, 0(wi+1)) +$$

$$+ \sum_{\substack{wi \neq 11\ldots1, \\ v_1 \preccurlyeq wi}} \Pi_3(0v_2, 1(wi+1)) \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 2 & 1 & 0 \\ 1 & 3 & 3 & 1 \end{bmatrix} (0v_2, 1(wi+1)) = 7 + \Pi_3(0v_2, 100\ldots0) \cdot 0 +$$

$$+ \sum_{\substack{wi \neq 11\ldots1, \\ v_1 \preccurlyeq wi}} \Pi_3(v_2, wi+1) \begin{bmatrix} 1 & 0 & 7 & 6 \\ 1 & 1 & 1 & 5 \\ 1 & 2 & 1 & 0 \\ 1 & 3 & 3 & 1 \end{bmatrix} (v_2, wi+1) \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix} (v_2, wi+1) +$$

$$+ \sum_{\substack{wi \neq 11\ldots1, \\ v_1 \preccurlyeq wi}} \Pi_3(0v_2, 1(wi+1)) \cdot 0 =$$

$$= 7 + \sum_{\substack{wi \neq 11\ldots1, \\ v_1 \preccurlyeq wi}} \Pi_3(v_2, wi+1) \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 2 & 1 & 0 \\ 1 & 3 & 3 & 1 \end{bmatrix} (v_2, wi+1) \equiv 4 \pmod 8.$$

Последнее сравнение следует из предположения индукции. Переход доказан. ■

Лемма 19. Пусть для $t \geqslant 0$ выполнено $wm = 0^t 01 w_1$, $wn = 0^t 10 w_2$, $\overline{w_1} - 2^{|w_1|-1} < \overline{w_2} \leqslant \overline{w_1}$; или $wm = 0^t 100 w_1$, $wn = 0^t 111 w_2$, $\overline{w_2} > \overline{w_1}$. Тогда

$$\begin{bmatrix} 1 \\ 5 \\ 7 \\ 3 \end{bmatrix} (wn) + \sum_{\substack{wi \neq 11\ldots1, \\ wm \preccurlyeq wi}} \Pi_3(wn, wi+1) \begin{bmatrix} 3 & 4 & 6 & 4 \\ 7 & 7 & 2 & 2 \\ 1 & 6 & 1 & 4 \\ 5 & 7 & 7 & 5 \end{bmatrix} (wn, wi+1) \equiv 4 \pmod 8.$$

Доказательство. Пусть $t = 0$ и $wm = 01 w_1$, $wn = 10 w_2$, $\overline{w_1} - 2^{|w_1|-1} < \overline{w_2} \leqslant \overline{w_1}$. Имеем

$$\begin{bmatrix} 1 \\ 5 \\ 7 \\ 3 \end{bmatrix} (10w_2) + \sum_{\substack{wi \neq 11\ldots1, \\ 01w_1 \preccurlyeq wi}} \Pi_3(10w_2, wi+1) \begin{bmatrix} 3 & 4 & 6 & 4 \\ 7 & 7 & 2 & 2 \\ 1 & 6 & 1 & 4 \\ 5 & 7 & 7 & 5 \end{bmatrix} (10w_2, wi+1) = 7 +$$

$$+ \sum_{\substack{wi \neq 11\ldots1, \\ 01w_1 \preccurlyeq wi}} \Pi_3(10w_2, wi+1) \,[\, 1\ 6\ 1\ 4 \,]\, (wi+1) = 7 + \Pi_3(10w_2, 100\ldots0) \,[\, 1\ 6\ 1\ 4 \,]\, (100\ldots0) +$$

$$+ \sum_{\substack{wi \neq 11\ldots1, \\ w_1 \preccurlyeq wi}} \Pi_3(10w_2, 01(wi+1)) \,[\, 1\ 6\ 1\ 4 \,]\, (01(wi+1)) +$$

$$+ \sum_{\substack{wi \neq 11\ldots1, \\ w_1 \preccurlyeq wi}} \Pi_3(10w_2, 11(wi+1)) \,[\, 1\ 6\ 1\ 4 \,]\, (11(wi+1)) =$$

$$= 7 + \begin{bmatrix} 1 \\ 5 \end{bmatrix} (w_2) \cdot 1 + \sum_{\substack{wi \neq 11\ldots1, \\ w_1 \preccurlyeq wi}} \Pi_3(10w_2, 01(wi+1)) \cdot 6 + \sum_{\substack{wi \neq 11\ldots1, \\ w_1 \preccurlyeq wi}} \Pi_3(10w_2, 11(wi+1)) \cdot 4 \equiv$$

$$\equiv \begin{bmatrix} 0 \\ 4 \end{bmatrix} (w_2) + \sum_{\substack{wi \neq 11\ldots1, \\ w_1 \preccurlyeq wi}} \Pi_2(0w_2, 1(wi+1)) \cdot 6 + \sum_{\substack{wi \neq 11\ldots1, \\ w_1 \preccurlyeq wi}} \Pi_2(0w_2, 1(wi+1) \cdot 4 =$$

$$= \begin{bmatrix} 0 \\ 4 \end{bmatrix} (w_2) + 2 \sum_{\substack{wi \neq 11\ldots1, \\ w_1 \preccurlyeq wi}} \Pi_2(0w_2, 1(wi+1)) \pmod 8.$$

Переход от Π_3 к Π_2 выполнен благодаря тому, что все слагаемые в суммах домножены на чётные числа 6 и 4. Сокращая на 2, сводим утверждение к следующему:

$$\begin{bmatrix} 0 \\ 2 \end{bmatrix} (w_2) + \sum_{\substack{wi \neq 11\ldots1, \\ w_1 \preccurlyeq wi}} \Pi_2(0w_2, 1(wi+1)) \equiv 2 \pmod 4.$$

Преобразуя левую часть, получим

$$\begin{bmatrix} 0 \\ 2 \end{bmatrix} (w_2) + \sum_{\substack{wi \neq 11\ldots1, \\ w_1 \preccurlyeq wi}} \Pi_2(0w_2, 1(wi+1)) =$$

$$= \begin{bmatrix} 0 \\ 2 \end{bmatrix} (w_2) + \sum_{\substack{wi \neq 11\ldots1, \\ w_1 \preccurlyeq wi}} \Pi_2(w_2, wi+1) \begin{bmatrix} 3 & 2 \\ 1 & 1 \end{bmatrix} (w_2, wi+1) \equiv 2 \pmod 4,$$

где последнее сравнение следует из леммы 16.

Пусть $t=0$ и $wm=100w_1$, $wn=111w_2$, $\overline{w_2}>\overline{w_1}$. Имеем

$$\begin{bmatrix}1\\5\\7\\3\end{bmatrix}(111w_2)+\sum_{\substack{wi\neq 11\ldots1,\\ 100w_1\preccurlyeq wi}}\Pi_3(111w_2,wi+1)\begin{bmatrix}3&4&6&4\\7&7&2&2\\1&6&1&4\\5&7&7&5\end{bmatrix}(111w_2,wi+1)=$$

$$=3+\sum_{\substack{wi\neq 11\ldots1,\\ 00w_1\preccurlyeq wi}}\Pi_3(111w_2,1(wi+1))\,[\,5\;7\;7\;5\,]\,(1(wi+1))=$$

$$=3+\sum_{\substack{wi\neq 11\ldots1,\\ 00w_1\preccurlyeq wi}}\Pi_3(11w_2,wi+1)\begin{bmatrix}1&0&3&6\\5&1&1&1\\5&2&1&0\\1&7&7&1\end{bmatrix}(11w_2,wi+1)\,[\,7\;5\,]\,(wi+1)\equiv$$

$$\equiv 3+\sum_{\substack{wi\neq 11\ldots1,\\ 00w_1\preccurlyeq wi}}\Pi_3(11w_2,wi+1)\begin{bmatrix}7&0&7&6\\3&7&5&5\\3&6&5&0\\7&1&3&5\end{bmatrix}(11w_2,wi+1)=$$

$$=3+\Pi_3(11w_2,100\ldots0)\begin{bmatrix}7&0&7&6\\3&7&5&5\\3&6&5&0\\7&1&3&5\end{bmatrix}(11w_2,100\ldots0)+$$

$$+\sum_{\substack{wi\neq 11\ldots1,\\ 0w_1\preccurlyeq wi}}\Pi_3(11w_2,0(wi+1))\begin{bmatrix}7&0&7&6\\3&7&5&5\\3&6&5&0\\7&1&3&5\end{bmatrix}(11w_2,0(wi+1))+$$

$$+\sum_{\substack{wi\neq 11\ldots1,\\ 0w_1\preccurlyeq wi}}\Pi_3(11w_2,1(wi+1))\begin{bmatrix}7&0&7&6\\3&7&5&5\\3&6&5&0\\7&1&3&5\end{bmatrix}(11w_2,1(wi+1))=3+\begin{bmatrix}5\\1\end{bmatrix}(w_2)\cdot 3+$$

$$+\sum_{\substack{wi\neq 11\ldots1,\\ 0w_1\preccurlyeq wi}}\Pi_3(1w_2,wi+1)\begin{bmatrix}1&4&3&2\\1&5&5&5\\1&6&5&4\\1&7&7&1\end{bmatrix}(1w_2,wi+1)\begin{bmatrix}3&6\\7&1\end{bmatrix}(1w_2,wi+1)+$$

$$+\sum_{\substack{wi\neq 11\ldots1,\\ 0w_1\preccurlyeq wi}}\Pi_3(1w_2,wi+1)\begin{bmatrix}1&0&3&6\\5&1&1&1\\5&2&1&0\\1&7&7&1\end{bmatrix}(1w_2,wi+1)\begin{bmatrix}5&0\\3&5\end{bmatrix}(1w_2,wi+1)\equiv$$

$$\equiv\begin{bmatrix}2\\6\end{bmatrix}(w_2)+\sum_{\substack{wi\neq 11\ldots1,\\ 0w_1\preccurlyeq wi}}\Pi_3(1w_2,wi+1)\begin{bmatrix}3&4&2&4\\3&7&6&6\\7&2&5&4\\7&1&7&1\end{bmatrix}(1w_2,wi+1)+$$

$$+\sum_{\substack{wi\neq 11\ldots1,\\ 0w_1\preccurlyeq wi}}\Pi_3(1w_2,wi+1)\begin{bmatrix}5&0&0&0\\1&5&0&0\\7&6&5&0\\3&5&3&5\end{bmatrix}(1w_2,wi+1)\equiv$$

$$\equiv\begin{bmatrix}2\\6\end{bmatrix}(w_2)+\sum_{\substack{wi\neq 11\ldots1,\\ 0w_1\preccurlyeq wi}}\Pi_3(1w_2,wi+1)\begin{bmatrix}0&4&2&4\\4&4&6&6\\6&0&2&4\\2&6&2&6\end{bmatrix}\pmod 8.$$

Это выражение уже получалось в лемме 18, где доказывалось, что оно сравнимо с 4 по модулю 8, как и требуется.

Пусть теперь $t = k + 1 > 0$. Обозначим $wm = 0v_1$, $wn = 0v_2$, где v_1 и v_2 удовлетворяют условиям леммы. Имеем

$$\begin{bmatrix} 1 \\ 5 \\ 7 \\ 3 \end{bmatrix}(0v_2) + \sum_{\substack{wi \neq 11\ldots1, \\ 0v_1 \preccurlyeq wi}} \Pi_3(0v_2, wi+1) \begin{bmatrix} 3 & 4 & 6 & 4 \\ 7 & 7 & 2 & 2 \\ 1 & 6 & 1 & 4 \\ 5 & 7 & 7 & 5 \end{bmatrix}(0v_2, wi+1) =$$

$$= \begin{bmatrix} 1 \\ 5 \end{bmatrix}(v_2) + \Pi_3(0v_2, 100\ldots0) \begin{bmatrix} 3 & 4 & 6 & 4 \\ 7 & 7 & 2 & 2 \\ 1 & 6 & 1 & 4 \\ 5 & 7 & 7 & 5 \end{bmatrix}(0v_2, 100\ldots0) +$$

$$+ \sum_{\substack{wi \neq 11\ldots1, \\ v_1 \preccurlyeq wi}} \Pi_3(0v_2, 0(wi+1)) \begin{bmatrix} 3 & 4 & 6 & 4 \\ 7 & 7 & 2 & 2 \\ 1 & 6 & 1 & 4 \\ 5 & 7 & 7 & 5 \end{bmatrix}(0v_2, 0(wi+1)) +$$

$$+ \sum_{\substack{wi \neq 11\ldots1, \\ v_1 \preccurlyeq wi}} \Pi_3(0v_2, 1(wi+1)) \begin{bmatrix} 3 & 4 & 6 & 4 \\ 7 & 7 & 2 & 2 \\ 1 & 6 & 1 & 4 \\ 5 & 7 & 7 & 5 \end{bmatrix}(0v_2, 1(wi+1)) = \begin{bmatrix} 1 \\ 5 \end{bmatrix}(v_2) +$$

$$+ \begin{bmatrix} 1 \\ 5 \\ 5 \\ 1 \end{bmatrix}(v_2) \begin{bmatrix} 6 \\ 2 \end{bmatrix}(v_2) + \sum_{\substack{wi \neq 11\ldots1, \\ v_1 \preccurlyeq wi}} \Pi_3(v_2, wi+1) \begin{bmatrix} 1 & 0 & 7 & 6 \\ 1 & 1 & 1 & 5 \\ 1 & 2 & 1 & 0 \\ 1 & 3 & 3 & 1 \end{bmatrix}(v_2, wi+1) \begin{bmatrix} 3 & 4 \\ 7 & 7 \end{bmatrix}(v_2, wi+1) +$$

$$+ \sum_{\substack{wi \neq 11\ldots1, \\ v_1 \preccurlyeq wi}} \Pi_3(v_2, wi+1) \begin{bmatrix} 1 & 4 & 7 & 2 \\ 5 & 5 & 5 & 1 \\ 5 & 6 & 5 & 4 \\ 1 & 3 & 3 & 1 \end{bmatrix}(v_2, wi+1) \begin{bmatrix} 6 & 4 \\ 2 & 2 \end{bmatrix}(v_2, wi+1) \equiv$$

$$\equiv \begin{bmatrix} 1 \\ 5 \end{bmatrix}(v_2) + \begin{bmatrix} 6 \\ 2 \end{bmatrix}(v_2) + \sum_{\substack{wi \neq 11\ldots1, \\ v_1 \preccurlyeq wi}} \Pi_3(v_2, wi+1) \begin{bmatrix} 3 & 0 & 4 & 0 \\ 3 & 3 & 4 & 4 \\ 7 & 6 & 7 & 0 \\ 7 & 5 & 5 & 7 \end{bmatrix}(v_2, wi+1) +$$

$$+ \sum_{\substack{wi \neq 11\ldots1, \\ v_1 \preccurlyeq wi}} \Pi_3(v_2, wi+1) \begin{bmatrix} 6 & 0 & 4 & 0 \\ 6 & 6 & 4 & 4 \\ 2 & 4 & 2 & 0 \\ 2 & 6 & 6 & 2 \end{bmatrix}(v_2, wi+1) \equiv$$

$$\equiv 7 + \sum_{\substack{wi \neq 11\ldots1, \\ v_1 \preccurlyeq wi}} \Pi_3(v_2, wi+1) \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 2 & 1 & 0 \\ 1 & 3 & 3 & 1 \end{bmatrix}(v_2, wi+1) \pmod 8.$$

Применяя лемму 18, получаем требуемое. ■

Теорема 6. Пусть для некоторых непустых слов w_1 и w_2 выполнено $wm = 011w_1$, $wn = 110w_2$, $\overline{w_1} - 2^{|w_1|-1} < \overline{w_2} \leqslant \overline{w_1}$; или $wm = 010^t01w_1$, $wn = 100^t10w_2$, $\overline{w_1} - 2^{|w_1|-1} <$

$< \overline{w_2} \leqslant \overline{w_1}$, $t \geqslant 0$; или $wm = 010^t100w_1$, $wn = 100^t111w_2$, $\overline{w_2} > \overline{w_1}$, $t \geqslant 0$. Тогда

$$1 + \sum_{\substack{wi \neq 11\ldots1,\\ wm \preccurlyeq wi}} \Pi_4(wn, wi+1) M_3^*(wn, wi+1) \equiv 8 \pmod{16}. \tag{5}$$

Доказательство. Вычисления показывают, что

$$M_4 = \begin{bmatrix}
1 & 0 & 15 & 6 & 1 & 12 & 15 & 10 & 1 & 8 & 15 & 15 & 1 & 4 & 15 & 2 \\
1 & 1 & 1 & 5 & 5 & 13 & 13 & 9 & 9 & 9 & 9 & 13 & 13 & 5 & 5 & 1 \\
1 & 2 & 1 & 0 & 5 & 14 & 13 & 4 & 9 & 10 & 9 & 8 & 13 & 6 & 5 & 12 \\
1 & 3 & 3 & 1 & 1 & 3 & 3 & 1 & 1 & 3 & 3 & 1 & 1 & 3 & 3 & 1 \\
1 & 4 & 3 & 2 & 1 & 0 & 3 & 6 & 1 & 12 & 3 & 10 & 1 & 8 & 3 & 14 \\
1 & 5 & 5 & 5 & 5 & 1 & 1 & 9 & 9 & 13 & 13 & 13 & 13 & 9 & 9 & 1 \\
1 & 6 & 5 & 12 & 5 & 2 & 1 & 0 & 9 & 14 & 13 & 4 & 13 & 10 & 9 & 8 \\
1 & 7 & 7 & 1 & 1 & 7 & 7 & 1 & 1 & 7 & 7 & 1 & 1 & 7 & 7 & 1 \\
1 & 8 & 7 & 14 & 1 & 4 & 7 & 2 & 1 & 0 & 7 & 6 & 1 & 12 & 7 & 10 \\
1 & 9 & 9 & 5 & 5 & 5 & 5 & 9 & 9 & 1 & 1 & 13 & 13 & 13 & 13 & 1 \\
1 & 10 & 9 & 8 & 5 & 6 & 5 & 12 & 9 & 2 & 1 & 0 & 13 & 14 & 13 & 4 \\
1 & 11 & 11 & 1 & 1 & 11 & 11 & 1 & 1 & 11 & 11 & 1 & 1 & 11 & 11 & 1 \\
1 & 12 & 11 & 10 & 1 & 8 & 11 & 14 & 1 & 4 & 11 & 2 & 1 & 0 & 11 & 6 \\
1 & 13 & 13 & 5 & 5 & 9 & 9 & 9 & 9 & 5 & 5 & 13 & 13 & 1 & 1 & 1 \\
1 & 14 & 13 & 4 & 5 & 10 & 9 & 8 & 9 & 6 & 5 & 12 & 13 & 2 & 1 & 0 \\
1 & 15 & 15 & 1 & 1 & 15 & 15 & 1 & 1 & 15 & 15 & 1 & 1 & 15 & 15 & 1
\end{bmatrix},$$

$$M_3^* \equiv \begin{bmatrix}
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 2 & 1 & 0 & 0 & 0 & 0 & 0 \\
1 & 3 & 3 & 1 & 0 & 0 & 0 & 0 \\
1 & 4 & 6 & 4 & 1 & 0 & 0 & 0 \\
1 & 5 & 10 & 10 & 5 & 1 & 0 & 0 \\
1 & 6 & 15 & 4 & 15 & 6 & 1 & 0 \\
1 & 7 & 5 & 3 & 3 & 5 & 7 & 1
\end{bmatrix} \pmod{16}.$$

Рассмотрим первый случай: $wm = 011w_1$, $wn = 110w_2$, $\overline{w_1} - 2^{|w_1|-1} < \overline{w_2} \leqslant \overline{w_1}$. Преобразуем левую часть (5):

$$1 + \sum_{\substack{wi \neq 11\ldots1,\\ wm \preccurlyeq wi}} \Pi_4(wn, wi+1) M_3^*(wn, wi+1) =$$

$$= 1 + \sum_{\substack{wi \neq 11\ldots1,\\ 011w_1 \preccurlyeq wi}} \Pi_4(110w_2, wi+1) M_3^*(110w_2, wi+1) \equiv$$

$$\equiv 1 + \sum_{\substack{wi \neq 11\ldots1,\\ 011w_1 \preccurlyeq wi}} \Pi_4(110w_2, wi+1) \left[\begin{smallmatrix} 1 & 6 & 15 & 4 & 15 & 6 & 1 & 0 \end{smallmatrix}\right] (wi+1) =$$

$$= 1 + \Pi_4(110w_2, 100\ldots0) \left[\begin{smallmatrix} 1 & 6 & 15 & 4 & 15 & 6 & 1 & 0 \end{smallmatrix}\right] (100\ldots0) +$$

$$+ \sum_{\substack{wi \neq 11\ldots1,\\ w_1 \preccurlyeq wi}} \Pi_4(110w_2, 011(wi+1)) \left[\begin{smallmatrix} 1 & 6 & 15 & 4 & 15 & 6 & 1 & 0 \end{smallmatrix}\right] (011(wi+1)) +$$

$$+ \sum_{\substack{wi \neq 11\ldots1,\\ w_1 \preccurlyeq wi}} \Pi_4(110w_2, 111(wi+1)) \left[\begin{smallmatrix} 1 & 6 & 15 & 4 & 15 & 6 & 1 & 0 \end{smallmatrix}\right] (111(wi+1)) = 1 + \begin{bmatrix} 1 \\ 9 \end{bmatrix} (w_2) \cdot 15 +$$

$$+\sum_{\substack{wi\neq 11\ldots 1,\\ w_1\preccurlyeq wi}}\Pi_4(110w_2,011(wi+1))\cdot 4+\sum_{\substack{wi\neq 11\ldots 1,\\ w_1\preccurlyeq wi}}\Pi_4(110w_2,111(wi+1))\cdot 0\equiv$$

$$\equiv\begin{bmatrix}0\\8\end{bmatrix}(w_2)+\sum_{\substack{wi\neq 11\ldots 1,\\ w_1\preccurlyeq wi}}\Pi_4(110w_2,011(wi+1))\cdot 4\pmod{16}.$$

Сокращая на 4, получаем, что утверждение теоремы сводится к виду

$$\begin{bmatrix}0\\2\end{bmatrix}(w_2)+\sum_{\substack{wi\neq 11\ldots 1,\\ w_1\preccurlyeq wi}}\Pi_4(110w_2,011(wi+1))\equiv 2\pmod 4.$$

Избавляясь от Π_4, получаем

$$\begin{bmatrix}0\\2\end{bmatrix}(w_2)+\sum_{\substack{wi\neq 11\ldots 1,\\ w_1\preccurlyeq wi}}\Pi_4(110w_2,011(wi+1))\equiv\begin{bmatrix}0\\2\end{bmatrix}(w_2)+$$

$$+\sum_{\substack{wi\neq 11\ldots 1,\\ w_1\preccurlyeq wi}}\Pi_2(0w_2,1(wi+1))=\begin{bmatrix}0\\2\end{bmatrix}(w_2)+\sum_{\substack{wi\neq 11\ldots 1,\\ w_1\preccurlyeq wi}}\Pi_2(w_2,wi+1)\begin{bmatrix}3&2\\1&1\end{bmatrix}(w_2,wi+1).$$

Применяя лемму 16, получаем требуемое.

Рассмотрим второй и третий случаи: $wm=01v_1$, $wn=10v_2$, где v_1 и v_2 удовлетворяют условию леммы 19. Имеем

$$1+\sum_{\substack{wi\neq 11\ldots 1,\\ wm\preccurlyeq wi}}\Pi_4(wn,wi+1)M_3^*(wn,wi+1)=1+\sum_{\substack{wi\neq 11\ldots 1,\\ 01v_1\preccurlyeq wi}}\Pi_4(10v_2,wi+1)M_3^*(10v_2,wi+1)=$$

$$=1+\Pi_4(10v_2,100\ldots 0)M_3^*(10v_2,100\ldots 0)+$$

$$+\sum_{\substack{wi\neq 11\ldots 1,\\ v_1\preccurlyeq wi}}\Pi_4(10v_2,01(wi+1))M_3^*(10v_2,01(wi+1))+$$

$$+\sum_{\substack{wi\neq 11\ldots 1,\\ v_1\preccurlyeq wi}}\Pi_4(10v_2,11(wi+1))M_3^*(10v_2,11(wi+1))=1+\begin{bmatrix}1\\9\\9\\1\end{bmatrix}(v_2)\begin{bmatrix}1\\5\end{bmatrix}(v_2)+$$

$$+\sum_{\substack{wi\neq 11\ldots 1,\\ v_1\preccurlyeq wi}}\Pi_4(0v_2,1(wi+1))M_4(10v_2,01(wi+1))\begin{bmatrix}6&4\\10&10\end{bmatrix}(v_2,wi+1)+$$

$$+\sum_{\substack{wi\neq 11\ldots 1,\\ v_1\preccurlyeq wi}}\Pi_4(0v_2,1(i+1))M_4(10v_2,11(wi+1))\begin{bmatrix}0&0\\0&0\end{bmatrix}(v_2,wi+1)\equiv 1+\begin{bmatrix}1\\9\\13\\5\end{bmatrix}(v_2)+$$

$$+2\sum_{\substack{wi\neq 11\ldots 1,\\ v_1\preccurlyeq wi}}\Pi_3(v_2,wi+1)\begin{bmatrix}1&4&7&2\\5&5&5&9\\5&6&5&12\\1&11&11&1\end{bmatrix}(v_2,wi+1)\begin{bmatrix}3&2\\5&5\end{bmatrix}(v_2,wi+1)\equiv$$

$$\equiv 2\begin{bmatrix}1\\5\\7\\3\end{bmatrix}(v_2)+2\sum_{\substack{wi\neq 11\ldots 1,\\ v_1\preccurlyeq wi}}\Pi_3(v_2,wi+1)\begin{bmatrix}3&4&6&4\\7&7&2&2\\1&6&1&4\\5&7&7&5\end{bmatrix}(v_2,wi+1)\pmod{16}.$$

Сокращая на 2, получаем, что нужно доказать

$$\begin{bmatrix} 1 \\ 5 \\ 7 \\ 3 \end{bmatrix} (v_2) + \sum_{\substack{wi \neq 11\ldots1, \\ v_1 \preccurlyeq wi}} \Pi_3(v_2, wi+1) \begin{bmatrix} 3 & 4 & 6 & 4 \\ 7 & 7 & 2 & 2 \\ 1 & 6 & 1 & 4 \\ 5 & 7 & 7 & 5 \end{bmatrix} (v_2, wi+1) \equiv 4 \pmod 8.$$

Используя лемму 19, получаем требуемое. ■

5. Оценки сумм биномиальных коэффициентов

Приступим теперь к оценке области возможных значений m и n. В работе [6] доказана

Теорема 7. При $n \geqslant 12$ выполнение (3) влечет

$$\frac{n}{2} + \frac{1}{2}\log_2 n + \frac{1}{2}\log_2\left(\frac{\pi}{2}e^{8/9}\right) - 1 > m \geqslant \frac{n-1}{2}.$$

Далее будем предполагать, что экстремальная функция с параметрами n и m существует, а значит, при $n \geqslant 12$ можно использовать неравенство, указанное в теореме. Вычисления показывают, что $\frac{1}{2}\log_2\left(\frac{\pi}{2}e^{8/9}\right) = 0{,}9669\cdots < 1$, поэтому будем использовать более удобную оценку $\frac{n}{2} + \frac{1}{2}\log_2 n > m$.

Пусть $n \geqslant 32$. Заметим, что $2n - 2m - 2 > 2n - n - \log_2 n - 2 = n - \log_2 n - 2 \geqslant$ $\geqslant 32 - 5 - 2 = 25$, поскольку $(n+1) - \log_2(n+1) = n - \log_2 \frac{n+1}{2} > n - \log_2 n$ при $n > 1$. Отсюда следует, что левая часть в формуле (3) делится на 2^{25}, а значит, все значения $m = \overline{wm}$ и $n = \overline{wn} \geqslant 32$ из условий теорем 4, 5 и 6 не подходят.

Лемма 20. Если $n \geqslant 32$, то двоичная запись n не начинается с 11.

Доказательство. Предположим противное, тогда n имеет вид $n = \overline{11w_2} = 2^{s+1} + 2^s + \overline{w_2}$, $|w_2| = s \geqslant 4$. Из теоремы 4 следует, что $m \notin I_1 = [2^{s+1}; 2^{s+1} + \overline{w_2})$; из теоремы 5 — $m \notin I_2 = [2^s + \overline{w_2}; \min(2^s + 2^{s-1} + \overline{w_2}, 2^{s+1}))$; ввиду теоремы 6 для $\overline{w_2} < 2^{s-1}$ выполнено $m \notin I_3 = [2^s + 2^{s-1} + \overline{w_2}; \min(2^s + 2^{s-1} + 2^{s-2} + \overline{w_2}, 2^{s+1}))$. Заметим, что $m \geqslant \frac{n-1}{2} = 2^s + 2^{s-1} + \frac{\overline{w_2} - 1}{2} \geqslant 2^s + 2^{s-1} + \frac{\overline{w_2} - 1 + (\overline{w_2} - 2^s + 1)}{2} = 2^s + 2^{s-1} + \overline{w_2} - 2^{s-1} =$ $= 2^s + \overline{w_2}$. Поскольку $m \notin I_2$, то $m \geqslant \min(2^s + 2^{s-1} + \overline{w_2}, 2^{s+1})$. Если $\overline{w_2} \geqslant 2^{s-1}$, то $m \geqslant 2^{s+1}$, а поскольку $m \notin I_1$, то $m \geqslant 2^{s+1} + \overline{w_2}$. Если же $\overline{w_2} < 2^{s-1}$, то $m \geqslant 2^s + 2^{s-1} + \overline{w_2}$, а поскольку $m \notin I_3$ для таких $\overline{w_2}$, то $m \geqslant \min(2^s + 2^{s-1} + 2^{s-2} + \overline{w_2}, 2^{s+1})$. Если $\overline{w_2} < 2^{s-2}$, то $m \geqslant 2^s + 2^{s-1} + 2^{s-2} + \overline{w_2}$, а если $2^{s-2} \leqslant \overline{w_2} < 2^{s-1}$, то $m \geqslant 2^{s+1}$, откуда из $m \notin I_1$ следует $m \geqslant 2^{s+1} + \overline{w_2}$. Получаем, что во всех случаях выполнено по крайней мере $m \geqslant 2^s + 2^{s-1} + 2^{s-2} + \overline{w_2}$. Отсюда $m - \frac{n}{2} \geqslant 2^s + 2^{s-1} + 2^{s-2} + \overline{w_2} - 2^s - 2^{s-1} - \frac{\overline{w_2}}{2} =$ $= 2^{s-2} + \frac{\overline{w_2}}{2} \geqslant 2^{s-2}$. Но $m < \frac{n}{2} + \frac{1}{2}\log_2 n$, значит, $\frac{1}{2}\log_2 n > 2^{s-2}$. С другой стороны, $\frac{1}{2}\log_2 n < \frac{1}{2}\log_2 2^{s+2} = \frac{s+2}{2}$. Легко видеть, что функция $f(s) = 2^{s-1} - s - 2$, $s \in \mathbb{Z}$, не убывает при $s \geqslant 1$, а $f(4) = 8 - 4 - 2 = 2 > 0$. Поскольку у нас $s \geqslant 4$, то $\frac{s+2}{2} < \frac{2^{s-1}}{2} = 2^{s-2}$. Получили противоречие. ■

Для удобства далее будем считать, что $2^{p-1} \leqslant n < 2^p$ $(p \geqslant 6)$.

Лемма 21. Пусть $n = \overline{10^k 1 w_2} = 2^{s+k+1} + 2^s + \overline{w_2}$, $|w_2| = s \geqslant 3$, $k \geqslant 1$. Тогда $m \geqslant \frac{n}{2} + 2^{s-3}$.

Доказательство. В предположениях леммы $p = s + k + 2$. Из теоремы 4 известно, что $m \notin I_1 = [2^{p-2} + 2^s + \overline{w_2}; 2^{p-1})$; по теореме 5 имеет место $m \notin I_2 = [2^{p-2}; \min(2^{p-2} + 2^{s-1} + \overline{w_2}, 2^{p-2} + 2^s))$; по теореме 6 для $\overline{w_2} < 2^{s-1}$ выполнено $m \notin I_3 = [2^{p-2} + 2^{s-1} + \overline{w_2}; \min(2^{p-2} + 2^{s-1} + 2^{s-2} + \overline{w_2}, 2^{p-2} + 2^s))$, а для $\overline{w_2} \geqslant 2^{s-1} + 2^{s-2}$ — $m \notin I_4 = [2^{p-2} + 2^s; 2^{p-2} + 2^{s-2} + \overline{w_2})$. Замечаем, что $m \geqslant \dfrac{n-1}{2} = 2^{p-2} + 2^{s-1} + \dfrac{\overline{w_2} - 1}{2} \geqslant$ $\geqslant 2^{p-2} + 2^{s-1} + \dfrac{\overline{w_2} - 1 + (\overline{w_2} - 2^s + 1)}{2} = 2^{p-2} + 2^{s-1} + \overline{w_2} - 2^{s-1} = 2^{p-2} + \overline{w_2}$. Поскольку $m \notin I_2$, то $m \geqslant \min(2^{p-2} + 2^{s-1} + \overline{w_2}, 2^{p-2} + 2^s)$. Рассмотрим несколько случаев. Если $\overline{w_2} \geqslant 3 \cdot 2^{s-2}$, то $m \geqslant 2^{p-2} + 2^s$, а поскольку $m \notin I_4$, то $m \geqslant 2^{p-2} + 2^{s-2} + \overline{w_2} \geqslant 2^{p-2} + 2^{s-2} + 3 \cdot 2^{s-3} + \dfrac{\overline{w_2}}{2} = \dfrac{n}{2} + 2^{s-3}$. Если $3 \cdot 2^{s-2} > \overline{w_2} \geqslant 2^{s-1}$, то $m \geqslant 2^{p-2} + 2^s \geqslant 2^{p-2} + 2^s - 3 \cdot 2^{s-3} + \dfrac{\overline{w_2}}{2} = \dfrac{n}{2} + 2^{s-3}$. Если $2^{s-1} > \overline{w_2} \geqslant 2^{s-2}$, то $m \geqslant 2^{p-2} + 2^{s-1} + \overline{w_2} \geqslant 2^{p-2} + 2^{s-1} + 2^{s-3} + \dfrac{\overline{w_2}}{2} = \dfrac{n}{2} + 2^{s-3}$. Если $2^{s-2} > \overline{w_2}$, то $m \geqslant 2^{p-2} + 2^{s-1} + \overline{w_2}$, а поскольку $m \notin I_3$, то $m \geqslant \min(2^{p-2} + 2^{s-1} + 2^{s-2} + \overline{w_2}, 2^{p-2} + 2^s) =$ $= 2^{p-2} + 2^{s-1} + 2^{s-2} + \overline{w_2} \geqslant 2^{p-2} + 2^{s-1} + 2^{s-2} + \dfrac{\overline{w_2}}{2} = \dfrac{n}{2} + 2^{s-2}$. Получаем, что во всех случаях $m \geqslant \dfrac{n}{2} + 2^{s-3}$. ■

Лемма 22. Выполнены неравенства

$$2^{p-1} \leqslant n < 2^{p-1} + 8p, \qquad 2^{p-2} \leqslant m < 2^{p-2} + \frac{9}{2}p, \qquad m < 2^{p-1}.$$

Доказательство. По лемме 20, двоичная запись n не может начинаться с 11. Если n не имеет вида $\overline{10^k 1 w_2}$, $|w_2| = s \geqslant 3$, то $n \leqslant 2^{p-1} + 7 < 2^{p-1} + 8p$. В противном случае применим лемму 21. Получим, что $m \geqslant \dfrac{n}{2} + 2^{s-3}$. Но $m < \dfrac{n}{2} + \dfrac{1}{2}\log_2 n < \dfrac{n}{2} + \dfrac{p}{2}$. Значит, $2^{s-3} < \dfrac{p}{2}$. Получаем, что $2^{p-1} \leqslant n = 2^{p-1} + 2^s + |w_2| < 2^{p-1} + 2^{s+1} < 2^{p-1} + 8p$. Первое неравенство доказано. Второе неравенство легко следует из первого: $2^{p-2} \leqslant \dfrac{n}{2} \leqslant$ $\leqslant m < \dfrac{n}{2} + \dfrac{1}{2}\log_2 n < 2^{p-2} + \dfrac{9}{2}p$. Учитывая, что двоичная запись n не может начинаться с 11, получаем, что при $p \geqslant 6$ выполнено $m < \dfrac{n}{2} + \dfrac{1}{2}\log_2 n < 2^{p-2} + 2^{p-3} + \dfrac{p}{2} <$ $< 2^{p-2} + 2^{p-3} + 2^{p-3} = 2^{p-1}$. ■

Лемма 23. Пусть $m = \overline{10^k w_1}$, $|w_1| = t$, $k \geqslant 0$, z — число нулей в слове w_1, $p \geqslant 9$. Тогда

$$1 + \sum_{\substack{i, \\ \binom{i}{m} \equiv 1 (\mathrm{mod}\, 2)}} \binom{n}{i+1} > 2^{n-t-2+z}.$$

Доказательство. Заметим, что

$$1 + \sum_{\substack{i, \\ \binom{i}{m} \equiv 1 (\mathrm{mod}\, 2)}} \binom{n}{i+1} = 1 + \sum_{j=0}^{2^z-1} \sum_{i=0}^{2^k-1} \binom{n}{m_j + i2^t + 1},$$

где m_j обозначает число, полученное из m заменой нулевых битов в слове w_1 на биты двоичной записи числа j. При этом биты числа j не могут сдвинуться влево больше,

чем на число единиц в w_1, т. е. на $t-z$ позиций. Отсюда получаем, что $m_j \leqslant m+2^{t-z}j$, а значит, с учётом $m_j \geqslant m \geqslant n/2$ можем записать

$$1+\sum_{j=0}^{2^z-1}\sum_{i=0}^{2^k-1}\binom{n}{m_j+i2^t+1} > \sum_{j=0}^{2^z-1}\sum_{i=0}^{2^k-1}\binom{n}{m+j2^{t-z}+i2^t+1} \geqslant$$
$$\geqslant 2^{z-t}\sum_{j=0}^{2^t-1}\sum_{i=0}^{2^k-1}\binom{n}{m+j+i2^t+1} = 2^{z-t}\sum_{i=0}^{2^{p-2}-1}\binom{n}{m+i+1} =$$
$$= 2^{z-t-1}\underbrace{\left(\sum_{i=0}^{2^{p-2}-1}\binom{n}{m+i+1}+\sum_{i=0}^{2^{p-2}-1}\binom{n}{n-m-i-1}\right)}_{A}.$$

Обозначим последнее выражение в скобках через A. Тогда A совпадает с $\sum\limits_{i=0}^{n}\binom{n}{i}=2^n$, за исключением нескольких недостающих крайних и средних слагаемых. Крайних слагаемых по $n-m-2^{p-2} \leqslant n-\left[\dfrac{n-1}{2}\right]-2^{p-2}=\left[\dfrac{n+1}{2}\right]-2^{p-2} \leqslant$ $\leqslant 2^{p-2}+4p-2^{p-2}=4p$ с каждой из сторон. Каждое из них не превосходит $\binom{n}{4p} < \left(\dfrac{ne}{4p}\right)^{4p} < 2^{4p^2}/p^{4p}$. Получаем, что сумма крайних слагаемых не больше $8p2^{4p^2}/p^{4p}$. Количество средних слагаемых равно $2m+1-n \leqslant n+[\log_2 n]+1-n=p$. Каждое из них, как известно, не превосходит $2^n/\sqrt{\pi n/2}$, а значит, их сумма не больше $2^n p 2^{-p/2}\sqrt{2/\pi}$. Таким образом, для того, чтобы получить $A > 2^{n-1}$, достаточно потребовать $8p2^{4p^2}/p^{4p} < 2^n(1/2-p2^{-p/2}\sqrt{2/\pi})$. Заметим, что при $p=9$ выполнено $p2^{-p/2}=9\cdot 2^{-9/2}=\dfrac{9}{16\sqrt{2}}<\dfrac{9}{16\cdot 7/5}=\dfrac{45}{112}<\dfrac{45}{105}=\dfrac{3}{7}$. При дальнейшем увеличении p это выражение уменьшается, поскольку $(p+1)/p=1+1/p \leqslant 1+1/9 \leqslant \sqrt{2}$. Поэтому достаточно доказать неравенство $8p2^{4p^2}/p^{4p} < 2^{2^{p-1}}(1/2-3\sqrt{2/\pi}/7)$. Заметим, что $1/2-3\sqrt{2/\pi}/7 > 1/2-3\sqrt{2/3}/7 > 1/2-3\sqrt{49/64}/7 = 1/2-3/8=1/8$, поэтому достаточно доказать неравенство $8p2^{4p^2}/p^{4p} < 2^{2^{p-1}}/8$. Используя $p \geqslant 9$, оцениваем левую часть как $8p2^{4p^2}/p^{4p}=2^{4p^2+3}/p^{4p-1} < 2^{4p^2+3}/2^{12p-3}=2^{4p^2-12p+6}$. Неравенство сводится к $4p^2-12p+6 < 2^{p-1}-3$, или $(2p-3)^2 < 2^{p-1}$, откуда $2p-3 < 2^{(p-1)/2}$. При $p=9$ неравенство верно, поскольку $2p-3=15<16=2^{(p-1)/2}$. А для $p>9$ оно выполнено, поскольку левая часть растет медленнее правой: $(2p-1)/(2p-3)=1+2/(2p-3) \leqslant 1+2/15 < \sqrt{2}$.

Таким образом, $2^{z-t-1}A > 2^{z-t-1}\cdot 2^{n-1}=2^{n-t-2+z}$. ■

Лемма 24. Пусть $p \geqslant 9$. Тогда

$$2^{p-1} \leqslant n < 2^{p-1}+16, \qquad 2^{p-2} \leqslant m < 2^{p-2}+8.$$

Доказательство. Предположим, что $n \geqslant 2^{p-1}+16$. Тогда $n=\overline{10^k1w_2}=2^{p-1}+2^s+\overline{w_2}$, $|w_2|=s \geqslant 4$, $k \geqslant 1$. Используя лемму 21, получаем $m \geqslant n/2+2^{s-3}$, откуда $2m-2^{s-2} \geqslant n$. Используя же лемму 23, получаем неравенство $2^{2n-2m-2} = 1+\sum\limits_{i,\binom{i}{m}\equiv 1 (\mathrm{mod}\, 2)}\binom{n}{i+1} > 2^{n-t-2+z}$, откуда $n \geqslant 2m-t+z+1$. Из $2m-2^{s-2} \geqslant n \geqslant 2m-t+z+1$ следует $2^{s-2}+1 \leqslant t-z$. Заметим, что $t-z$ равно количеству единиц в двоичной записи числа m, не считая самой старшей. Значит, выполнено неравенство

$m \geqslant 2^{p-2} + 2^{t-z} - 1$. Получаем $n + t - z - 1 \geqslant 2m \geqslant 2^{p-1} + 2^{t-z+1} - 2$. Оценивая n, получим $2^{p-1} + 2^{t-z+1} - 2 \leqslant 2^{p-1} + 2^{s+1} - 1 + t - z - 1$, откуда $2^{t-z+1} - (t-z) \leqslant 2^{s+1}$. Поскольку $s \geqslant 4$, отсюда следует, что $t - z \leqslant s$. В таком случае $2^{s-2} + 1 \leqslant t - z \leqslant s$, но это неравенство ложно для всех $s \geqslant 4$. Получили противоречие, значит, $n < 2^{p-1} + 16$.

Предположим теперь, что $m \geqslant 2^{p-2} + 8$. Из леммы 23, как доказано выше, следует неравенство $n \geqslant 2m - (t-z) + 1$. При $m = 2^{p-2} + 8$ получаем $2m - (t-z) + 1 = 2^{p-1} + 16 - 1 + 1 = 2^{p-1} + 16$. Заметим, что при увеличении числа m на единицу количество единиц в его двоичной записи может возрасти максимум на 1, а $2m$ при этом увеличивается на 2. Значит, $2m - (t-z) + 1 \geqslant 2^{p-1} + 16$ для всех $m \geqslant 2^{p-2} + 8$, что противоречит доказанному неравенству $n < 2^{p-1} + 16$. Следовательно, предположение не верно, и $m < 2^{p-2} + 8$. ■

Лемма 25. Пусть $p \geqslant 9$, $m = \overline{10^k w_1}$, $|w_1| = t \leqslant 3$, z — число нулей в слове w_1. Тогда выполнено

$$1 + \sum_{i, \binom{i}{m} \equiv 1 (\mathrm{mod}\, 2)} \binom{n}{i+1} < 2^{n-t+z}.$$

Доказательство. Имеем

$$1 + \sum_{i, \binom{i}{m} \equiv 1 (\mathrm{mod}\, 2)} \binom{n}{i+1} \leqslant 1 + 2^z \sum_{i=0}^{2^{p-t-2}-1} \binom{n}{m+1+i2^t} \leqslant$$

$$\leqslant 1 + 2^z 2^{-t} \left((2^t - 1)\binom{n}{m+1} + \sum_{i=m+1}^{n} \binom{n}{i} \right) < 2^{z-t} \left(2^t \binom{n}{[n/2]} + 2^{n-1} \right).$$

В последнем неравенстве использована оценка $2^{z-t}\binom{n}{[n/2]} \geqslant 2^{-3}\binom{n}{[n/2]} > 1$. Заметим, что из $p \geqslant 9$ следует неравенство $\binom{n}{[n/2]} \leqslant \dfrac{2^n}{\sqrt{\pi n/2}} \leqslant \dfrac{2^n}{\sqrt{n}} \leqslant \dfrac{2^n}{2^4} \leqslant 2^{n-t-1}$. Значит, $2^{z-t}\left(2^t\binom{n}{[n/2]} + 2^{n-1}\right) \leqslant 2^{z-t}(2^t \cdot 2^{n-t-1} + 2^{n-1}) = 2^{z-t} \cdot 2^n = 2^{n-t+z}$, что и требовалось доказать. ■

Лемма 26. Пусть $p \geqslant 9$. Тогда $m = \overline{10^k w_1}$, $|w_1| = t \leqslant 3$ и $n = 2m - t + z + 1$, где z — число нулей в слове w_1.

Доказательство. Представимость числа m в указанном виде следует из леммы 24. Применяя леммы 25 и 23, получаем, что выражение

$$2^{2n-2m-2} = 1 + \sum_{i, \binom{i}{m} \equiv 1 (\mathrm{mod}\, 2)} \binom{n}{i+1}$$

заключено между $2^{n-t+z-2}$ и 2^{n-t+z}. Значит, $2n - 2m - 2 = n - t + z - 1$, откуда получаем утверждение леммы. ■

Таким образом, при $p \geqslant 9$ для пары (m, n) остаются следующие возможности: $(2^{p-2}, 2^{p-1}+1)$, $(2^{p-2}+1, 2^{p-1}+2)$, $(2^{p-2}+2, 2^{p-1}+4)$, $(2^{p-2}+3, 2^{p-1}+5)$, $(2^{p-2}+4, 2^{p-1}+8)$, $(2^{p-2}+5, 2^{p-1}+9)$, $(2^{p-2}+6, 2^{p-1}+11)$, $(2^{p-2}+7, 2^{p-1}+12)$. Первые две пары удовлетворяют равенству по лемме 7. Из оставшихся пар достаточно проверить

лишь половину, поскольку для чётных m выполнено

$$1+\sum_{i,\binom{i}{m}\equiv 1(\bmod 2)}\binom{n}{i+1}=1+\sum_{2|i,\binom{i}{m}\equiv 1(\bmod 2)}\left(\binom{n}{i+1}+\binom{n}{i+2}\right)=$$
$$=1+\sum_{i,\binom{i+1}{m+1}\equiv 1(\bmod 2)}\left(\binom{n}{i+1}+\binom{n}{i+2}\right)=1+\sum_{i,\binom{i}{m+1}\equiv 1(\bmod 2)}\left(\binom{n}{i}+\binom{n}{i+1}\right)=$$
$$=1+\sum_{i,\binom{i}{m+1}\equiv 1(\bmod 2)}\binom{n+1}{i+1}.$$

Из пар с нечётным m — $(2^{p-2}+3, 2^{p-1}+5)$, $(2^{p-2}+5, 2^{p-1}+9)$ и $(2^{p-2}+7, 2^{p-1}+12)$ — первые два случая невозможны по теореме 6. Рассмотрим третий случай.

Лемма 27. Для любого $k \geqslant 0$ выполнено

$$3+\sum_{\substack{wi\neq 11\ldots1,\\ 0^{k+1}111\preccurlyeq wi}}\Pi_2(0^k1100, wi+1)\begin{bmatrix}1&0\\1&1\end{bmatrix}(0^k1100, wi+1)\equiv 2 \pmod 4.$$

Доказательство. Докажем утверждение по индукции. База: $k=0$.

$$3+\sum_{\substack{wi\neq 11\ldots1,\\ 0111\preccurlyeq wi}}\Pi_2(1100, wi+1)\begin{bmatrix}1&0\\1&1\end{bmatrix}(1100, wi+1)=$$
$$=3+\Pi_2(1100,1000)\begin{bmatrix}1&0\\1&1\end{bmatrix}(1100,1000)=3+3\cdot 1\equiv 2 \pmod 4.$$

Докажем переход. Пусть утверждение верно для $k=t$, докажем его для $k=t+1$. Поскольку $k>0$, то в сумме отличны от нуля лишь слагаемые, у которых $wi+1$ начинаются с 0:

$$3+\sum_{\substack{wi\neq 11\ldots1,\\ 0^{t+2}111\preccurlyeq wi}}\Pi_2(0^{t+1}1100, wi+1)\begin{bmatrix}1&0\\1&1\end{bmatrix}(0^{t+1}1100, wi+1)=$$
$$=3+\sum_{\substack{wi\neq 11\ldots1,\\ 0^{t+1}111\preccurlyeq wi}}\Pi_2(0^{t+1}1100, 0(wi+1))\cdot 1=$$
$$=3+\sum_{\substack{wi\neq 11\ldots1,\\ 0^{t+1}111\preccurlyeq wi}}\Pi_2(0^t1100, wi+1)\begin{bmatrix}1&0\\1&1\end{bmatrix}(0^t1100, wi+1).$$

Последнее выражение сравнимо с 2 по модулю 4 по предположению индукции. Переход доказан. ■

Теорема 8. Пусть $wm=010^k111$, $wn=10^k1100$. Тогда

$$1+\sum_{\substack{wi\neq 11\ldots1,\\ wm\preccurlyeq wi}}\Pi_3(wn, wi+1)M_2^*(wn, wi+1)\equiv 4 \pmod 8.$$

Доказательство. Преобразуем выражение:

$$1+\sum_{\substack{wi\neq 11\ldots 1,\\ wm\preccurlyeq wi}}\Pi_3(wn,wi+1)M_2^*(wn,wi+1)=1+\Pi_3(10^k1100,100\ldots0)M_2^*(10^k1100,100\ldots0)+$$

$$+\sum_{\substack{wi\neq 11\ldots 1,\\ 10^k111\preccurlyeq wi}}\Pi_3(10^k1100,0(wi+1))M_2^*(10^k1100,0(wi+1))+$$

$$+\sum_{\substack{wi\neq 11\ldots 1,\\ 10^k111\preccurlyeq wi}}\Pi_3(10^k1100,1(wi+1))M_2^*(10^k1100,1(wi+1))=1+\begin{bmatrix}1\\5\\5\\1\end{bmatrix}(0^k1100)\begin{bmatrix}1\\3\end{bmatrix}(0^k1100)+$$

$$+\sum_{\substack{wi\neq 11\ldots 1,\\ 10^k111\preccurlyeq wi}}\Pi_3(0^k1100,wi+1)\begin{bmatrix}1&4&3&2\\1&5&5&5\\1&6&5&4\\1&7&7&1\end{bmatrix}(0^k1100,wi+1)\begin{bmatrix}1&2\\1&3\end{bmatrix}(0^k1100,wi+1)+$$

$$+\sum_{\substack{wi\neq 11\ldots 1,\\ 10^k111\preccurlyeq wi}}\Pi_3(0^k1100,wi+1)\begin{bmatrix}1&0&3&6\\5&1&1&1\\5&2&1&0\\1&7&7&1\end{bmatrix}(0^k1100,wi+1)\begin{bmatrix}1&0\\3&1\end{bmatrix}(0^k1100,wi+1)\equiv$$

$$\equiv\begin{bmatrix}2\\6\\0\\4\end{bmatrix}(0^k1100)+\sum_{\substack{wi\neq 11\ldots 1,\\ 10^k111\preccurlyeq wi}}\Pi_3(0^k1100,wi+1)\begin{bmatrix}2&4&6&4\\6&6&2&2\\0&4&0&4\\4&4&4&4\end{bmatrix}(0^k1100,wi+1)\pmod 8.$$

Сократив на 2, получим, что нужно доказать

$$\begin{bmatrix}1\\3\\0\\2\end{bmatrix}(0^k1100)+\sum_{\substack{wi\neq 11\ldots 1,\\ 10^k111\preccurlyeq wi}}\Pi_3(0^k1100,wi+1)\begin{bmatrix}1&2&3&2\\3&3&1&1\\0&2&0&2\\2&2&2&2\end{bmatrix}(0^k1100,wi+1)\equiv 2\pmod 4.$$

В случае $k=0$ в сумме нет слагаемых, а значит, левая часть равна $\begin{bmatrix}1\\3\\0\\2\end{bmatrix}(1100)=2$,

что и требуется. Пусть $k>0$. Продолжаем вычисления:

$$\begin{bmatrix}1\\3\\0\\2\end{bmatrix}(0^k1100)+\sum_{\substack{wi\neq 11\ldots 1,\\ 10^k111\preccurlyeq wi}}\Pi_3(0^k1100,wi+1)\begin{bmatrix}1&2&3&2\\3&3&1&1\\0&2&0&2\\2&2&2&2\end{bmatrix}(0^k1100,wi+1)=$$

$$=\begin{bmatrix}1\\3\end{bmatrix}(0^{k-1}1100)+\sum_{\substack{wi\neq 11\ldots 1,\\ 0^k111\preccurlyeq wi}}\Pi_3(0^k1100,1(wi+1))\begin{bmatrix}1&2&3&2\\3&3&1&1\\0&2&0&2\\2&2&2&2\end{bmatrix}(0^k1100,1(wi+1))\equiv$$

$$\equiv\begin{bmatrix}1\\3\end{bmatrix}(0^{k-1}1100)+\sum_{\substack{wi\neq 11\ldots 1,\\ 0^k111\preccurlyeq wi}}\Pi_2(0^{k-1}1100,wi+1)\begin{bmatrix}3&2\\1&1\end{bmatrix}(0^{k-1}1100,wi+1)\pmod 4.$$

В случае $k = 1$ в сумме одно слагаемое, значит, выражение равно $\begin{bmatrix} 1 \\ 3 \end{bmatrix}(1100) + {+}\Pi_2(1100, 1000)\begin{bmatrix} 3 & 2 \\ 1 & 1 \end{bmatrix}(1100, 1000) = 3 + 3 \cdot 1 \equiv 2 \pmod 4$, что и требуется. Пусть $k \geqslant 2$. Продолжаем вычисления:

$$\begin{bmatrix} 1 \\ 3 \end{bmatrix}(0^{k-1}1100) + \sum_{\substack{wi \neq 11\ldots1, \\ 0^k111 \preccurlyeq wi}} \Pi_2(0^{k-1}1100, wi+1)\begin{bmatrix} 3 & 2 \\ 1 & 1 \end{bmatrix}(0^{k-1}1100, wi+1) =$$

$$= 1 + \Pi_2(0^{k-1}1100, 100\ldots0)\begin{bmatrix} 3 & 2 \\ 1 & 1 \end{bmatrix}(0^{k-1}1100, 100\ldots0) +$$

$$+ \sum_{\substack{wi \neq 11\ldots1, \\ 0^{k-1}111 \preccurlyeq wi}} \Pi_2(0^{k-1}1100, 0(wi+1))\begin{bmatrix} 3 & 2 \\ 1 & 1 \end{bmatrix}(0^{k-1}1100, 0(wi+1)) +$$

$$+ \sum_{\substack{wi \neq 11\ldots1, \\ 0^{k-1}111 \preccurlyeq wi}} \Pi_2(0^{k-1}1100, 1(wi+1))\begin{bmatrix} 3 & 2 \\ 1 & 1 \end{bmatrix}(0^{k-1}1100, 1(wi+1)) =$$

$$= 1 + \begin{bmatrix} 3 \\ 1 \end{bmatrix}(0^{k-2}1100) \cdot 2 + \sum_{\substack{wi \neq 11\ldots1, \\ 0^{k-1}111 \preccurlyeq wi}} \Pi_2(0^{k-2}1100, wi+1)\begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}(0^{k-2}1100, wi+1) \cdot 3 +$$

$$+ \sum_{\substack{wi \neq 11\ldots1, \\ 0^{k-1}111 \preccurlyeq wi}} \Pi_2(0^{k-2}1100, wi+1)\begin{bmatrix} 3 & 2 \\ 1 & 1 \end{bmatrix}(0^{k-2}1100, wi+1) \cdot 2 \equiv$$

$$\equiv 3 + \sum_{\substack{wi \neq 11\ldots1, \\ 0^{k-1}111 \preccurlyeq wi}} \Pi_2(0^{k-2}1100, wi+1)\begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}(0^{k-2}1100, wi+1) \pmod 4.$$

Остаётся воспользоваться леммой 27. ■

Таким образом, случай $m = 2^{p-2} + 7$, $n = 2^{p-1} + 12$ невозможен. Поэтому при $p \geqslant 9$ единственными подходящими парами являются пары $m = 2^{p-2}$, $n = 2^{p-1} + 1$ и $m = 2^{p-2} + 1$, $n = 2^{p-1} + 2$. Отсюда получаем основной результат.

Теорема 9. Если $n \geqslant 512$, $0 < m < n - 1$ и пара n, m не принадлежит сериям $m = 2^s$, $n = 2^{s+1} + 1$ и $m = 2^s + 1$, $n = 2^{s+1} + 2$ при $s \geqslant 0$, то для корреляционно-иммунной порядка m булевой функции f от n переменных выполнено неравенство

$$\mathrm{nl}(f) \leqslant 2^{n-1} - 2^{m+1}.$$

Проверка равенства (1) для $n < 512$, $n - 2 \geqslant m \geqslant (n-1)/2$ на компьютере позволяет убрать ограничение $n \geqslant 512$ из предыдущей теоремы.

ЛИТЕРАТУРА

1. *Sarkar P. and Maitra S.* Nonlinearity bounds and constructions of resilient boolean functions // LNCS. 2000. V. 1880. P. 515–532.
2. *Tarannikov Yu.* On resilient Boolean functions with maximal possible nonlinearity // LNCS. 2000. V. 1977. P. 19–30.
3. *Zheng Y. and Zhang X. M.* Improved upper bound on the nonlinearity of high order correlation immune functions // LNCS. 2001. V. 2012. P. 264–274.
4. *Тарасников Ю. В.* О корреляционно-иммунных и устойчивых булевых функциях // Математические вопросы кибернетики. Вып. 11. М.: Физматлит, 2002. С. 91–148.

5. *Халявин А. В.* Построение 4 корреляционно-иммунных булевых функций от 9 переменных с нелинейностью 240 // Материалы X Междунар. семинара «Дискретная математика и её приложения». Москва, МГУ, 1–6 февраля 2010 г. М.: Изд-во механико-математического факультета МГУ, 2010. С. 534.
6. *Ботев А. А.* О соотношениях между корреляционной иммунностью, нелинейностью и весом для неуравновешенных булевых функций // Математические вопросы кибернетики. Вып. 11. М.: Физматлит, 2002. С. 149–162.
7. *Guo-Zhen X. and Massey J. A.* Spectral characterization of correlation-immune combining functions // IEEE Trans. Information Theory. 1988. V. 34. No. 3. P. 569–571.

МАТЕМАТИЧЕСКИЕ МЕТОДЫ КРИПТОГРАФИИ

DOI 10.17223/20710410/11/5

УДК 519.7

НЕКОТОРЫЕ ПРОТОКОЛЫ ДОВЕРЕННОЙ ЦИФРОВОЙ ПОДПИСИ

Е. А. Толюпа

Ярославский государственный университет им. П. Г. Демидова, г. Ярославль, Россия

E-mail: tolyupa@gmail.com

Предложены усовершенствование протокола доверенной цифровой подписи (ДЦП) авторов J.-Y Lee, J.-H. Cheon и S. Kim, устранившее его уязвимость, позволяющую сторонам создавать доверенный ключ подписания независимо друг от друга, и протокол коллективной ДЦП, который можно использовать при реализации процедуры голосования.

Ключевые слова: *электронная цифровая подпись, доверенная цифровая подпись, анализ безопасности, голосование, коллективная подпись.*

Введение

Традиционные протоколы цифровой подписи (ЦП) позволяют пользователю выполнить подпись под документом самостоятельно, и любой участник электронного информационного обмена может убедиться в верности подписи, однако эти протоколы не позволяют одному участнику, скажем **A**, делегировать свои права другому участнику — **B**, который может подписывать сообщения от имени **A**. Такая необходимость возникает, когда, например, руководитель организации по той или иной причине (по состоянию здоровья или техническим возможностям) не может подписать документ сам. В этой ситуации он может доверить право подписывать документы от своего имени другому лицу, например своему заместителю. Заместитель, получив такое право, может подписывать сообщения от лица руководителя, а проверяющий будет знать, кто и от чьего имени поставил данную подпись. Решить подобную задачу позволяют протоколы доверенной цифровой подписи (ДЦП, или proxy signature).

Основоположниками теории протоколов ДЦП и разработчиками первого такого протокола являются M. Mambo, K. Usuda и E. Okamoto [1]. Ими были сформулированы первые требования к безопасности протоколов ДЦП [1, 2], расширенные позже в [3, 4]. Безопасная ДЦП должна удовлетворять следующим требованиям:

1. Проверяемость — проверяющий может быть убежден, что подпись поставлена с согласия доверителя.

2. Стойкость к фальсификации — только назначенная доверителем сторона может создать верную доверенную подпись от лица доверителя. Другими словами, участник **A** и третья сторона, не выбранная им в качестве доверенного подписчика, не смогут создать верную ДЦП от имени доверенного участника **B**.

3. Строгая идентификация — каждый может идентифицировать соответствующую доверенную сторону из доверенной подписи.

4. Неотрекаемость — если доверенный подписчик создает подпись под документом, то в дальнейшем он не сможет заявить, что подпись выполнена кем-то другим.

5. Противостояние злоупотреблению — доверенная сторона не должна использовать ключ подписания для целей, не разрешенных доверителем в информации о полномочиях. В случае злоупотребления ответственность доверенного подписчика должна определяться явно.

В данной работе изложены базовый протокол ДЦП, предложенный в [1], его модификация LCK из [5] на случай открытого канала связи и атака на нее из [6], демонстрирующая нарушение в ней требования 2 к безопасности. Предложено усовершенствование протокола LCK, устранившее его уязвимость этой атакой и сохранившее его пригодность для открытого канала. Предложен также протокол коллективной ДЦП с защищенным каналом связи, который можно использовать при реализации процедуры голосования. Все протоколы основаны на общей идее распределения ключевой информации, которая позволяет делегировать права подписи доверенной стороне, и построены по следующей общей схеме.

1. Общая схема протокола ДЦП

Основная идея, используемая при создании доверенных подписей, — построить такую схему распределения ключевой информации, чтобы можно было использовать уже существующие алгоритмы цифровой подписи. В распределении ключевой информации лежит следующий принцип.

Пусть имеются два участника информационного обмена **A** и **B** — соответственно доверитель права цифровой подписи и его доверенная сторона (доверенное лицо), получающая право подписывать от имени доверителя. Каждый из участников **A** и **B** имеет свою пару ключей — (x_A, y_A) и (x_B, y_B) соответственно, где первый ключ (x) в паре закрытый, а второй (y) открытый. **A** генерирует доверенность на право подписания от его имени и вместе с информацией о полномочиях (период действия доверенной подписи, типы документов, которые разрешено подписывать доверенной стороне, идентификаторы доверителя и доверенного лица и т. п.) передает (по защищенному или открытому каналу) участнику **B**, который на основании информации, содержащейся в доверенности, идентифицирует **A** и создает пару доверенных ключей (x_p, y_p), предназначенных соответственно для создания и проверки цифровой подписи по некоторому известному алгоритму. Теперь при помощи x_p он может подписывать сообщения от лица **A**, применяя этот алгоритм. Любой проверяющий должен самостоятельно вычислить ключ y_p участника **B**, используя информацию из доверенности участника **A**. Таким образом, убедившись, что ключ участнику **B** действительно доверил участник **A**, можно проверить подпись с помощью процедуры верификации из того же алгоритма цифровой подписи.

Общая схема протокола ДЦП, реализующая эту основную идею, выглядит следующим образом.

Действия доверителя **A**:

1. Генерирует случайное *обязательство* k и вычисляет его *свидетельство* $K = f(k)$, где f — некоторая односторонняя функция.

2. Создает s_A, зависящую от величин из $\{x_A, y_B, k, K\}$ и возможных полномочий m_w, и величины s_A, K и, возможно, m_w, составляющие доверенность, отправляет доверенному лицу **B**.

Действия доверенной стороны **B**:

1. Проверяет (с целью идентификации **A**) выполнение некоторого *идентификационного условия*, зависящего от y_A, y_B и полученных s_A и K.

2. Если оно выполнено, то вычисляет доверенные ключи: x_p — в зависимости от s_A, x_B, y_A, y_B и, возможно, m_w; $y_p = f(x_p)$.

3. С помощью подходящего алгоритма ЦП подписывает некоторый документ M, используя x_p в качестве ключа подписания. Созданную так подпись обозначает $\mathrm{Sign}(M, x_p)$. Формирует доверенную подпись под M как $\sigma = (\mathrm{Sign}(M, x_p), K, y_A, y_B, m_w)$, возможно, без m_w.

Действия **проверяющего**:

1. Вычисляет ключ y_p, применяя некоторый алгоритм к данным y_A, y_B, K и, возможно, m_w, содержащимся в σ.

2. По ключу y_p проверяет подпись $\mathrm{Sign}(M, x_p)$ под документом M при помощи процедуры верификации соответствующего алгоритма ЦП.

Конкретные протоколы, построенные по этой схеме, различаются друг от друга алгоритмами выполнения перечисленных в ней действий участников протокола, которые (алгоритмы), в свою очередь, зависят и от применяемого математического аппарата, и от характера используемого канала передачи данных — открытый или закрытый, и от вида информации о полномочиях, в том числе от ее наличия или отсутствия.

В рассматриваемых далее конкретных протоколах ДЦП используется аппарат теории чисел. В них, кроме уже обозначенного, используются: p и q — большие простые числа, причем $q|p-1$; $g \in \mathbb{Z}_p^*$, порядок g равен q и g является общеизвестным. Считается, что вычисление дискретного логарифма — трудоёмкая задача. Соответственно в роли односторонней функции $f(k)$ выступает $g^k \bmod p$. Первый и второй ключи (x и y) в паре ключей каждого участника и в паре доверенных ключей связаны соотношением $y = g^x \bmod p$. Действие 3 участника **B** и действие 2 **проверяющего** из общей схемы сохраняются во всех протоколах, с тем лишь уточнением, что в последних используется алгоритм ЦП, базирующийся на сложности задачи дискретного логарифмирования. В приводимых ниже описаниях протоколов эти действия опущены.

2. Протокол MUO

Этот протокол предложили M. Mambo, K. Usuda и E. Okamoto (MUO) в [1]. Именно он и является базовым для модификаций, рассматриваемых в данной работе.

Действия **A**:

1. Генерирует случайное число $k \in \mathbb{Z}_q^*$ и вычисляет $K = g^k \bmod p$.

2. Вычисляет $s_A = (x_A + k \cdot K) \bmod q$ и посылает (s_A, K) участнику **B** по защищенному каналу связи.

Действия **B**:

1. Для идентификации **A** проверяет сравнение: $g^{s_A} = y_A \cdot K^K (\bmod p)$.

2. Если оно выполнено, то вычисляет доверенный ключ подписания

$$x_p = (s_A + x_B \cdot y_B) \bmod q.$$

Действия **проверяющего**:

1. Зная (из подписи) свидетельство K, открытые ключи доверителя (y_A) и доверенной стороны (y_B), вычисляет доверенный ключ y_p для проверки подписи по правилу

$$y_p = y_A \cdot K^K \cdot y_B^{y_B} \bmod p.$$

Заметим, что данный протокол не удовлетворяет пятому требованию безопасности, поскольку действие 2 участника **A** не включает в себя передачу информации о полномочиях.

В приводимых ниже модификациях протокола MUO действие 1 участника **A** то же самое, что и в базовом протоколе, в связи с чем в их описаниях оно опущено.

3. Протокол LCK

Протокол MUO требует наличия защищенного канала между доверителем и доверенной стороной. В своей работе [5] J.-Y. Lee, J.-H. Cheon и S. Kim (LCK) модифицировали его, что позволило передавать доверенность по открытому каналу.

Действия **A**:

2. Вычисляет $s_A = (x_A + k \cdot y_B) \bmod q$ и пару (s_A, K) отправляет участнику **B** по общедоступному каналу.

Действия **B**:

1. Проверяет идентификационное условие $g^{s_A} = y_A \cdot K^{y_B} (\bmod p)$.

2. Если оно выполнено, то **B** вычисляет доверенный ключ подписания:

$$x_p = (s_A + x_B \cdot y_A) \bmod q.$$

Действия **проверяющего**:

1. Вычисляет ключ y_p по следующему правилу: $y_p = y_A \cdot K^{y_B} \cdot y_B^{y_A} \bmod p$.

В [6] показано, что такая модификация протокола MUO приводит его к уязвимости, позволяющей:

1) сторонам **A** и **B** доверенный ключ подписания создавать независимо друг от друга, т. е. без участия их в выполнении протокола, и тем самым участнику **A** доверенную подпись за участника **B** ставить без его ведома, а участнику **B** подпись от имени **A** ставить, не имея на то доверенности от последнего;
2) третьей стороне **C** доверенный ключ подписания создать независимо от участников **A** и **B**, т. е. без их участия в выполнении протокола, и тем самым поставить доверенную подпись;
3) третьей стороне **C** доверенность для **B** модифицировать под себя, т. е. изменить её таким образом, как если бы участник **A** доверил право подписи стороне **C**, а не участнику **B**.

Первая атака выглядит следующим образом. Если **A** возьмет случайное число s и вычислит $K = g^s \cdot y_B^{-y_A \cdot y_B^{-1}} \bmod p$ и $x_p = (x_A + s \cdot y_B) \bmod q$, то это x_p будет обладать свойством доверенного ключа подписания, а именно: подпись, поставленную этим ключом, любой проверяющий ее на ключе y_p примет за подпись, поставленную стороной **B** от имени стороны **A**, ибо $y_p = y_A \cdot K^{y_B} \cdot y_B^{y_A} \bmod p = y_A \left(g^s \cdot y_B^{-y_A \cdot y_B^{-1}}\right)^{y_B} y_B^{y_A} \bmod p = y_A \cdot g^{s y_B} \bmod p = g^{x_A} g^{s y_B} \bmod p = g^{x_p} \bmod p$, т. е. (x_p, y_p) является доверенной парой ключей.

Обратно, если участник **B** возьмет случайное s и изготовит $K = g^s \cdot y_A^{-y_B^{-1}} \bmod p$ и $x_p = (s \cdot y_B + x_B \cdot y_A) \bmod q$, то будет $y_p = y_A \cdot K^{y_B} \cdot y_B^{y_A} \bmod p = y_A \cdot y_A^{-1} g^{s y_B} \cdot y_B^{y_A} \bmod p = g^{s y_B} g^{x_B y_A} \bmod p = g^{x_p} \bmod p$, что означает, что и это x_p является доверенным ключом подписания.

Вторая атака выглядит следующим образом. Если **C** возьмет случайное число s и вычислит $K = y_A^{-y_B^{-1}} \cdot y_B^{-y_A \cdot y_B^{-1}} \cdot g^s \bmod p$ и $x_p = s \cdot y_B \bmod q$, то это x_p будет также обладать свойствами доверенного ключа подписания, так как $y_p = y_A \cdot K^{y_B} \cdot y_B^{y_A} \bmod p = y_A \left(y_A^{-y_B^{-1}} \cdot y_B^{-y_A \cdot y_B^{-1}} \cdot g^s\right)^{y_B} y_B^{y_A} \bmod p = y_A \cdot y_A^{-1} \cdot y_B^{-y_A} \cdot g^{s \cdot y_B} \cdot y_B^{y_A} \bmod p = g^{s \cdot y_B} \bmod p = g^{x_p} \bmod p$.

Третья атака заключается в следующем: имея (из открытого канала) доверенность (s_A, K), предназначенную участником **A** для **B**, сторона **C** берёт случайное число $\overline{k} \in \mathbb{Z}_q^*$, вычисляет $\overline{K} = K^{y_B \cdot {y_C}^{-1}} \cdot g^{\overline{k}} \bmod p$ и $\overline{s}_A = s_A + \overline{k} \cdot y_C \bmod q$ и получает пару

$(\overline{s}_A, \overline{K})$, обладающую свойствами доверенности, сгенерированной участником **A** для **C**, так как $g^{\overline{s}_A} \bmod p = g^{s_A+\overline{k}\cdot y_C} \bmod p = g^{x_A+k\cdot y_B+\overline{k}\cdot y_C} \bmod p = y_A \cdot g^{(k\cdot y_B\cdot y_C^{-1}+\overline{k})y_C} \bmod p = y_A \left(K^{y_B\cdot y_C^{-1}} \cdot g^{\overline{k}}\right)^{y_C} \bmod p = y_A \cdot \overline{K}^{y_C} \bmod p$.

4. Авторская модификация протокола MUO

Предлагается следующая модификация протокола MUO с передачей доверенности по открытому каналу и без уязвимости предыдущей модификации.

Действия **A**:

2. Вычисляет $s_A = (x_A + k \cdot y_B \cdot K) \bmod q$ и пару (s_A, K) отправляет участнику **B** по общедоступному каналу связи.

Действия **B**:

1. Проверяет сравнение $g^{s_A} = y_A \cdot K^{y_B\cdot K} (\bmod p)$.

2. Если оно выполнено, то **B** вычисляет доверенный ключ подписания как

$$x_p = (s_A + x_B \cdot y_A) \bmod q.$$

Действия **проверяющего**:

1. Зная (из подписи) свидетельство K, открытые ключи доверителя (y_A) и доверенной стороны (y_B), вычисляет доверенный ключ y_p для проверки подписи по правилу

$$y_p = y_A \cdot K^{y_B\cdot K} \cdot y_B^{y_A} \bmod p.$$

Здесь, как и в протоколе LCK, доверенность s_A зависит от значения y_B. Предположим, что злоумышленник **C** воспользуется переданной по открытому каналу доверенностью и создаст секретный ключ $\overline{x}_p = s_A + x_C \cdot y_A$. В этом случае соответствующий открытый ключ имеет вид $\overline{y}_p = g^{\overline{x}_p} \bmod p = y_A \cdot K^{y_B\cdot K} \cdot y_C^{y_A} \bmod p$. Злоумышленник подписывает документ и заявляет, что сделал это по доверенности от **A**. Согласно протоколу, лицо, проверяющее эту подпись $\sigma = (\mathrm{Sign}(M, x_p), K, y_A, y_C)$, располагая только свидетельством K, открытыми ключами доверителя (y_A) и злоумышленника (y_C), не сможет корректно вычислить $\overline{y}_p$ и проверить подпись σ, не подставив в выражение для $\overline{y}_p$ значения открытого ключа y_B того участника **B**, для которого в действительности предназначалась доверенность, и тем самым уличит мошенника. Таким образом, добавив в доверенность зависимость s_A от y_B, доверитель **A** однозначно определил доверенную сторону, что позволило использовать открытый канал для передачи доверенности.

Покажем, что при невозможности взятия дискретного логарифма атаки из [6], описанные выше, на этот протокол невозможны.

В самом деле, реализация первых двух атак выглядит следующим образом.

В первой атаке участник **A** должен взять такие s и K, чтобы для $\overline{x}_p = (x_A + s \cdot y_B) \bmod q$ выполнилось равенство $g^{\overline{x}_p} \bmod p = y_p$, т. е. сравнение

$$y_A \cdot g^{s y_B} = y_A \cdot K^{y_B\cdot K} \cdot y_B^{y_A} (\bmod p),$$

или, что то же самое,

$$K^K = g^s \cdot y_B^{-y_A\cdot y_B^{-1}} (\bmod p).$$

С другой стороны, участник **B** должен подобрать s и K так, чтобы для $\overline{x}_p = (s \cdot y_B + x_B \cdot y_A) \bmod q$ выполнилось равенство $g^{\overline{x}_p} \bmod p = y_p$, т. е. сравнение

$$K^K = g^s \cdot y_A^{-y_B^{-1}} (\bmod p).$$

Для реализации второй атаки злоумышленник **C** должен подобрать значения s и K так, чтобы для $\overline{x}_p = s \cdot y_B \bmod q$ выполнялось равенство $g^{\overline{x}_p} \bmod p = y_p$, т. е. сравнение

$$K^K = g^s \cdot y_A^{-y_B^{-1}} \cdot y_B^{-y_A \cdot y_B^{-1}} (\mathrm{mod}\, p).$$

Для реализации третьей атаки злоумышленнику **C** необходимо подобрать такие значения $\overline{k}$ и $\overline{K}$, чтобы выполнилось сравнение

$$g^{s_A + \overline{k} \cdot y_C} = y_A \cdot \overline{K}^{y_C \cdot \overline{K}} (\mathrm{mod}\, p),$$

или, что то же самое,

$$\overline{K}^{\overline{K}} = g^{\overline{k}} \cdot K^{y_B \cdot y_C^{-1} \cdot K} (\mathrm{mod}\, p).$$

Следовательно, в каждом случае злоумышленник должен создать такие x и y, которые удовлетворяют сравнению $y^y = ag^x (\mathrm{mod}\, p)$ для заданных a, g и p. При выбранном y нахождение нужного x требует вычисления дискретного логарифма, а при выбранном x нахождение нужного y представляется не менее сложной задачей.

Таким образом, добавив в протокол LCK зависимость s_A от K, удалось убрать его слабость к атакам из [6].

5. Протокол коллективной ДЦП

Построим протокол ДЦП, в котором участвуют доверитель **A** и доверенное множество $\mathfrak{B}$, состоящие из n доверенных сторон $\mathbf{B}_1$, $\mathbf{B}_2$, …, $\mathbf{B}_n$. Доверенная подпись, поставленная любым из участников множества $\mathfrak{B}$, должна удовлетворять следующим условиям.

1) **Проверяющий** может убедиться, что подпись поставил один из участников $\mathfrak{B}$, но не должен узнать, кто именно.
2) Доверитель **A** может однозначно идентифицировать участника множества $\mathfrak{B}$, подписавшего документ.

Пусть для каждого $i = 1, 2, \ldots, n$ имеются: x_i — секретный и y_i — открытый ключи участника $\mathbf{B}_i$, связанные соотношением $y_i = g^{x_i} \bmod p$.

Действия **A**:

1. Генерирует случайные числа $r_1, \ldots, r_n \in \mathbb{Z}_q^*$ и публикует набор чисел $R_1 = g^{r_1} \bmod p$, …, $R_n = g^{r_n} \bmod p$ без указания их соответствия участникам из множества $\mathfrak{B}$. Все значения $R_1, \ldots, R_n$ должны быть различны.

2. Для каждого $i = 1, 2, \ldots, n$ вычисляет $S_i = (x_A + r_i \cdot R_i) \bmod q$ и посылает (S_i, R_i) участнику $\mathbf{B}_i$ по защищенному каналу связи. Значение R_i необходимо передавать участнику $\mathbf{B}_i$, так как последний не знает, какой именно элемент из опубликованных следует применять в вычислениях при идентификации **A**.

Действия множества $\mathfrak{B}$:

Для каждого $i = 1, 2, \ldots, n$ участник $\mathbf{B}_i$ производит следующие действия:

1. Для идентификации участника **A** проверяет сравнение $g^{S_i} = y_A \cdot R_i^{R_i} (\mathrm{mod}\, p)$ и, если оно выполнено, то принимает S_i.

2. Выбирает случайное $k_i \in \mathbb{Z}_q^*$, вычисляет $K_i = g^{k_i} \bmod p$ и $s_i = k_i + x_i \cdot K_i \bmod q$. Отправляет пару (s_i, K_i) как доверенность участникам множества $\mathfrak{B}$ по защищенному каналу связи. На этом шаге $\mathbf{B}_i$ доверил право подписи другим участникам множества $\mathfrak{B}$.

Таким образом, каждый участник $\mathbf{B}_i$ располагает набором

$$(S_i, R_i, s_1, \ldots, s_n, K_1, \ldots, K_n).$$

Действия подписывающего участника $\mathbf{B}_i$:

1. Вычисляет секретный ключ (для доверенного подписания)

$$x_{p_i} = S_i + \sum_{j=1}^{n} s_j (\mathrm{mod}\, q).$$

2. Доверенная подпись под документом M, поставленная с использованием ключа x_{p_i}, имеет вид $\sigma = (\mathrm{Sign}(M, x_{p_i}); y_A; R_i; y_1, \ldots, y_n; K_1, \ldots, K_n)$. Документ с подписью помещается на общедоступный ресурс.

Действия **проверяющего:**

1. Проверяет, что значение R_i из σ содержится в наборе, опубликованном доверителем.

2. Вычисляет открытый ключ (для проверки доверенной подписи)

$$y_{p_i} = y_A \cdot R_i^{R_i} \prod_{j=1}^{n} (K_j \cdot {y_j}^{K_j}) \bmod p. \tag{1}$$

(Непосредственно проверяется, что $y_{p_i} = g^{x_{p_i}} \bmod p$, т. е. y_{p_i} — действительно ключ для проверки подписи $\mathrm{Sign}(M, x_{p_i})$.)

В подписи σ под документом M содержится открытый ключ доверителя (y_A) и параметр R_i, который использовался при вычислении доверенности участника $\mathbf{A}$ участнику $\mathbf{B}_i$ в $\mathfrak{B}$. При вычислении открытого ключа (1) эти параметры позволяют убедиться, что подпись поставлена с согласия доверителя $\mathbf{A}$ одним из участников в $\mathfrak{B}$. Остальные n множителей имеют вид $K_j \cdot y_j^{K_j}$. Исходя из этой информации, проверяющий не может сделать вывод, кто именно из участников множества $\mathfrak{B}$ поставил подпись. Доверитель же $\mathbf{A}$ по элементу R_i может установить, кто из участников в $\mathfrak{B}$ подписал сообщение.

Протокол коллективной ДЦП можно использовать для реализации процедуры голосования. Пусть доверитель хочет, чтобы n доверенных лиц проголосовали от его имени. Для этого он наделяет их соответствующими полномочиями, выполняя указанный протокол. Каждое доверенное лицо вычисляет свой секретный ключ и подписывает сообщение (голос). Проверяющий в момент вычисления открытого ключа контролирует, что значение R_i не повторяется. Это является залогом того, что каждое доверенное лицо проголосовало единожды. Проверяющий не может проверить, как проголосовал каждый из участников. Доверитель может идентифицировать подписавшего по значению R_i, что является борьбой со злоупотреблением со стороны доверенного лица.

Заключение

Таким образом, автору удалось убрать уязвимость протокола LCK и создать более безопасный протокол, в котором для передачи доверенности не требуется защищенный канал связи. Однозначное определение доверителем доверенной стороны можно считать особенностью предложенного протокола, которую целесообразно использовать и при реализации протоколов с секретным каналом. Предложен протокол голосования на базе алгоритма коллективной ДЦП. Автор считает, что разработка прикладных протоколов на базе существующих протоколов ДЦП является интересным направлением, позволяющим найти им применение в существующих задачах.

ЛИТЕРАТУРА

1. *Mambo M., Usuda K., and Okamoto E.* Proxy signatures: Delegation of the power to sign messages // IEICE Trans. Fundamentals. 1996. V. E79-A. No. 9. P. 1338–1353.

2. *Mambo M., Usuda K., and Okamoto E.* Proxy signatures for delegating signing operation // Proc. of 3rd ACM Conference on Computer and Communications Security (CCS'96). ACM Press, 1996. P. 48–57.
3. *Kim S., Park S., and Won D* Proxy signatures, revisited // Information and Communications Security (ICICS'97). 1997. LNCS. V. 1334, P. 223–232.
4. *Lee B., Kim H., and Kim K.* Strong proxy signature and its applications // Proc. of the 2001 Symposium on Cryptography and Information Security (SCIS'01), Oiso, Japan, Jan. 23–26, 2001. V. 2/2. P. 603–608.
5. *Lee J.-Y., Cheon J.-H., and Kim S.* An analysis of proxy signatures: Is a secure channel necessary? // LNCS. 2003. V. 2612. P. 68–79.
6. *Wang G., Bao F., Zhou J., and Deng R. H.* Security Analysis of Some Proxy Signatures // LNCS. 2004. V. 2971. P. 305–319.

МАТЕМАТИЧЕСКИЕ ОСНОВЫ КОМПЬЮТЕРНОЙ БЕЗОПАСНОСТИ

DOI 10.17223/20710410/11/6

УДК 004.94

ПРАВИЛА ПРЕОБРАЗОВАНИЯ СОСТОЯНИЙ БАЗОВОЙ РОЛЕВОЙ ДП-МОДЕЛИ УПРАВЛЕНИЯ ДОСТУПОМ И ИНФОРМАЦИОННЫМИ ПОТОКАМИ В ОПЕРАЦИОННЫХ СИСТЕМАХ[1]

П. Н. Девянин

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

E-mail: peter_devyanin@hotmail.com

Рассматривается базовая ролевая ДП-модель управления доступом и информационными потоками *в операционных системах*, в которую по сравнению с базовой ролевой ДП-моделью включены учетные записи пользователей, сущности, параметрически ассоциированные с субъект-сессиями или ролями, мандатный контроль целостности, фактические доступы субъект-сессий. Основное внимание уделено изменениям в условиях и результатах применения правил преобразования состояний. Обосновывается утверждение о возможности использования только монотонных правил преобразования состояний для анализа условий передачи прав доступа ролей, получения доступов или реализации информационных потоков.

Ключевые слова: *компьютерная безопасность, ролевая ДП-модель, операционная система.*

1. Описание модели

Существующие дискреционные, мандатные и ролевые ДП-модели [1, 2] предоставляют достаточно развитые механизмы моделирования управления доступом и информационными потоками в компьютерных системах. В то же время современные операционные системы (ОС) обладают существенными особенностями (например, мандатным контролем целостности в ОС семейства *Microsoft Windows*), обуславливающими необходимость адаптации существующих формальных моделей для обеспечения условий анализа безопасности ОС, а реализуемые в ОС механизмы защиты достаточно сложны, что затрудняет непосредственно разработку формальной модели, детально им соответствующей. В связи с этим в работе строится ролевая ДП-модель, которую целесообразно считать промежуточным шагом на пути формирования ролевой модели управления доступом и информационными потоками в ОС, направленным на совершенствование техники формального описания данного механизма защиты. Она основана на базовой ролевой ДП-модели (БР ДП-модели) [2], ДП-модели с функционально или параметрически ассоциированными с субъектами сущностями (ФПАС ДП-модели) [3], при этом для обеспечения возможности теоретического анализа условий обеспечения целостности программной среды ОС в ней использованы элементы модели мандатной политики целостности информации Биба и модели мандатного ролевого управления доступом [2]. Построенную ДП-модель будем называть базовой ро-

[1]Работа выполнена при поддержке гранта МД-2.2010.10.

левой ДП-моделью управления доступом и информационными потоками в ОС (или, сокращенно, БРОС ДП-моделью).

С учетом специфики реализации управления доступом и информационными потоками в ОС сделаем предположения.

Предположение 1. В рамках БРОС ДП-модели пользователям соответствуют их учетные записи, для каждой из которых задается множество авторизованных ролей. Каждая учетная запись пользователя вне зависимости от имеющихся у нее авторизованных ролей является учетной записью либо доверенного, либо недоверенного пользователя. Каждая субъект-сессия является либо доверенной, либо недоверенной и функционирует от имени учетной записи доверенного или недоверенного пользователя соответственно. Доверенные субъект-сессии не инициируют создание субъект-сессий. Каждая недоверенная субъект-сессия может создать недоверенную субъект-сессию.

Предположение 2. Для каждой учетной записи пользователя задается множество сущностей, параметрически с ней ассоциированных, реализация от каждой из которых информационного потока по памяти к субъект-сессии позволяет ей создать субъект-сессию от имени данной учетной записи пользователя. Множество сущностей, параметрически ассоциированных с учетной записью пользователя, не содержит субъект-сессий и не изменяется в процессе функционирования системы.

Предположение 3. Функционально ассоциированными с субъект-сессией являются сущности, от которых зависит вид преобразования данных, реализуемого субъект-сессией. Только информационный поток по памяти к сущности, функционально ассоциированной с субъект-сессией, приводит к изменению вида преобразования данных, реализуемого этой субъект-сессией. Множество сущностей, функционально ассоциированных с субъект-сессией, задается при её создании и не изменяется в процессе функционирования системы. При создании новой субъект-сессии множество функционально ассоциированных с ней сущностей задается только в зависимости от сущности, из которой создается данная субъект-сессия, и учетной записи пользователя, от имени которой другая субъект-сессия создает данную субъект-сессию.

Предположение 4. Параметрически ассоциированными с субъект-сессией являются сущности, которые содержат данные, позволяющие идентифицировать вид преобразования данных, реализуемого субъект-сессией. Множество сущностей, параметрически ассоциированных с субъект-сессией, не содержит субъект-сессий и не изменяется в процессе функционирования системы.

Предположение 5. Для каждой роли или административной роли задается (возможно, пустое) множество сущностей, параметрически с ней ассоциированных и не являющихся субъект-сессиями. При этом для получения или удаления роли из множества текущих ролей субъект-сессии кроме наличия данной роли во множестве авторизованных ролей учетной записи пользователя, от имени которой функционирует субъект-сессия, необходимо реализовать к себе информационные потоки по памяти от всех сущностей, параметрически ассоциированных с данной ролью.

Заметим, что в ДП-модели файловой системы (ФС ДП-модели) [4] для анализа условий создания новых субъектов, выполняющих функции защиты файловой системы (например, криптографической защиты или кодирования данных), рассматривались потенциальные доверенные субъекты. При этом реализация информационных потоков по памяти от сущностей, параметрически ассоциированных с ними, позволяла недоверенным субъектам создать доверенные субъекты. В рамках БРОС ДП-модели вместо потенциальных доверенных субъектов анализируются учетные записи пользователей, параметрически ассоциированные с ними сущности и сущности, параметриче-

ски ассоциированные с ролями. Кроме того, сущности, параметрически ассоциированные с учетной записью пользователя и использованные для создания субъект-сессии, не обязательно становятся параметрически ассоциированными с субъект-сессией. Например, для создания субъект-сессии на удаленном сетевом ресурсе могут потребоваться сущность «имя пользователя» и сущность «пароль». При этом параметрически ассоциированной с созданной субъект-сессией сущностью может стать только сущность «идентификатор удаленной сессии», «знание» которой (реализация от нее информационного потока по памяти) может позволить получить контроль над созданной субъект-сессией.

Введём следующие определения и обозначения:

$E = O \cup C$ — множество сущностей, где O — множество объектов, C — множество контейнеров и $O \cap C = \varnothing$;

U — множество учетных записей пользователей;

$]u[\subset E \setminus S$ — множество сущностей, параметрически ассоциированных с учетной записью пользователя $u \in U$;

$UE = \{e \in]u[\colon u \in U\}$ — множество сущностей, каждая из которых параметрически ассоциирована хотя бы с одной учетной записью пользователя;

L_U — множество учетных записей доверенных пользователей;

N_U — множество учетных записей недоверенных пользователей, при этом по определению справедливы равенства $L_U \cup N_U = U$, $L_U \cap N_U = \varnothing$;

$S \subseteq E$ — множество субъект-сессий учетных записей пользователей;

L_S — множество доверенных субъект-сессий;

N_S — множество недоверенных субъект-сессий, при этом по определению справедливо равенство $L_S \cap N_S = \varnothing$;

R — множество ролей;

AR — множество административных ролей ($AR \cap R = \varnothing$);

$]r[\subset E \setminus S$ — множество сущностей, параметрически ассоциированных с ролью или административной ролью $r \in R \cup AR$;

$RE = \{e \in]r[\colon r \in R \cup AR\}$ — множество сущностей, параметрически ассоциированных со всеми ролями;

$R_r = \{read_r, write_r, append_r, execute_r, own_r\}$ — множество видов прав доступа;

$R_a = \{read_a, write_a, append_a, own_a\}$ — множество видов доступа;

$R_f = \{write_m, write_t\}$ — множество видов информационных потоков;

$P \subseteq E \times R_r$ — множеств прав доступа к сущностям;

$A \subseteq S \times E \times R_a$ — множество доступов субъект-сессий к сущностям;

$F \subseteq E \times E \times R_f$ — множество информационных потоков;

$UA : U \to 2^R$ — функция авторизованных ролей учетных записей пользователей;

$AUA : U \to 2^{AR}$ — функция авторизованных административных ролей учетных записей пользователей;

$PA : R \to 2^P$ — функция прав доступа ролей;

$user : S \to U$ — функция принадлежности субъект-сессии учетной записи пользователя;

$roles : S \to 2^R \cup 2^{AR}$ — функция текущих ролей субъект-сессий;

$can\_manage\_rights : AR \to 2^R$ — функция администрирования прав доступа ролей;

$[s] \subset E \cup U$ — множество сущностей, функционально ассоциированных с субъект-сессией s (при этом по определению выполняется условие $s \in [s]$), и учетных записей

пользователей, от имени каждой из которых может быть создана субъект-сессия, являющаяся функционально ассоциированной с субъект-сессией s сущностью;

$fa : U \times E \to 2^E \cup 2^U$ — функция, задающая множества сущностей, функционально ассоциированных с субъект-сессией при ее создании от имени учетной записи пользователя из сущности;

$]s[\subset E \setminus S$ — множество сущностей, параметрически ассоциированных с субъект-сессией $s \in S$;

$fp : U \times E \to 2^E$ — функция, задающая множества сущностей, параметрически ассоциированных с субъект-сессией при ее создании из сущности от имени учетной записи пользователя. По определению выполняется условие: для каждой субъект-сессии $s \in S$ существует единственная сущность $e_s \in E$, такая, что справедливы равенства $fp(user(s), e_s) =]s[$ и $fa(user(s), e_s) = [s]$;

$H_E : E \to 2^E$ — функция иерархии сущностей;

$H_R : R \to 2^R$ — функция иерархии ролей;

$H_{AR} : AR \to 2^{AR}$ — функция иерархии административных ролей.

Определение 1. Доверенную субъект-сессию y назовем функционально корректной, когда во множество функционально ассоциированных с ней сущностей $[y]$ не входят недоверенные субъект-сессии.

Определение 2. Доверенную субъект-сессию y назовем функционально корректной относительно доверенной субъект-сессии y' и сущности e, когда субъект-сессия y не реализует информационный поток по памяти от сущности e к некоторой сущности e', функционально ассоциированной с доверенной субъект-сессией y'. При этом используем обозначение $y_f(E) \subset L_S \times E$ — множество пар вида (доверенная субъект-сессия, сущность), относительно которых функционально корректна доверенная субъект-сессия y.

Определение 3. Доверенную субъект-сессию y назовем параметрически корректной относительно доверенной субъект-сессии y' и сущности e, когда субъект-сессия y не реализует информационный поток по памяти к сущности e от некоторой сущности e', параметрически ассоциированной с доверенной субъект-сессией y'. При этом используем обозначение $y_p(E) \subset L_S \times E$ — множество пар вида (доверенная субъект-сессия, сущность), относительно которых параметрически корректна доверенная субъект-сессия y.

Определение 4. Доверенную субъект-сессию назовем корректной относительно информационных потоков по времени, когда она не участвует в их реализации. При этом используем обозначения: $LF_S \subset L_S$ — множество доверенных субъект-сессий, корректных относительно информационных потоков по времени; $NF_S \subset L_S$ — множество доверенных субъект-сессий, некорректных относительно информационных потоков по времени, при этом по определению справедливы равенства $LF_S \cap NF_S = \varnothing$, $LF_S \cup NF_S = L_S$.

В ОС семейств *Microsoft Windows Vista*/7/2008 реализуется механизм мандатного контроля целостности (*MIC — Mandatory Integrity Control*), предназначенный для обеспечения контроля целостности программной среды ОС. Данный механизм является перспективным и эффективным средством повышения безопасности управления доступом и информационными потоками в ОС. Таким образом, целесообразно, помимо ролевого управления доступом, при построении БРОС ДП-модели включить в нее элементы, позволяющие анализировать механизм мандатного контроля целостности. При этом без ограничения общности можно рассматривать только два уровня целостности,

предоставляющие возможность обеспечить целостность доверенных субъект-сессий системы. Также имеет смысл учесть наличие в ОС специального механизма защиты от несанкционированного повышения уровней целостности процессов (например, механизм *User Account Control — UAC* в ОС семейств *Microsoft Windows Vista*/7/2008). Данный механизм предусматривает выполнение специальных процедур для активизации процессов или получения доступа к сущностям с высоким уровнем целостности.

Используем следующие обозначения:

$(LI, \leqslant)$ — линейная шкала двух уровней целостности данных, где $LI = \{i\_low, i\_high\}$, $i\_low < i\_high$;

$(i_u, i_e, i_r, i_s) \in I$ — четверка функций уровней целостности, при этом:

$i_u : U \to LI$ — функция, задающая для каждой учетной записи пользователя максимальный разрешенный уровень целостности субъект-сессий, функционирующих от ее имени;

$i_e : E \setminus S \to LI$ — функция, задающая уровень целостности для каждой сущности, не являющейся субъект-сессией;

$i_r : R \cup AR \to LI$ — функция, задающая для каждой роли или административной роли ее уровень целостности;

$i_s : S \to LI$ — функция, задающая для каждой субъект-сессии ее текущий уровень целостности;

I — множества всех четверок функций заданного вида.

По аналогии с понятием параметрической корректности доверенных субъект-сессий дадим следующее определение.

Определение 5. Доверенную субъект-сессию y назовем корректной в смысле целостности относительно сущности e, обладающей высоким уровнем целостности i_high, если субъект-сессия y не реализует информационный поток по памяти к сущности e от некоторой сущности e', обладающей низким уровнем целостности i_low. При этом используем обозначение $y_i(E) \subset E$ — множество сущностей, относительно которых корректна в смысле целостности доверенная субъект-сессия y.

В рамках БРОС ДП-модели определим состояние системы.

Определение 6. Пусть определены:

- множества учетных записей пользователей U, сущностей E, субъект-сессий S, прав доступа к сущностям P, учетных записей доверенных пользователей L_U, доверенных субъект-сессий L_S, доступов субъект-сессий к сущностям A, информационных потоков F;
- функции авторизованных ролей учетных записей пользователей UA, авторизованных административных ролей учетных записей пользователей AUA, прав доступа ролей PA, принадлежности субъект-сессий учетным записям пользователей $user$, текущих ролей субъект-сессий $roles$, уровней целостности (i_u, i_e, i_r, i_s), иерархии ролей H_R, иерархии административных ролей H_{AR}, иерархии сущностей H_E.

Определим $G = (UA, AUA, PA, user, roles, (i_u, i_e, i_r, i_s), A, F, H_R, H_{AR}, H_E, L_U, L_S)$ — состояние системы.

Заметим, что, так же как в БР ДП-модели, механизм статических и динамических ограничений рассматриваться не будет.

Используем обозначения:

$\Sigma(G^*, OP)$ — система, при этом G^* — множество всех возможных состояний, OP — множество правил преобразования состояний, заданных в таблице;

$G \vdash_{op} G'$ — переход системы $\Sigma(G^*, OP)$ из состояния G в состояние G' с использованием правила преобразования состояний $op \in OP$;

$\Sigma(G^*, OP, G_0)$ — система $\Sigma(G^*, OP)$ с начальным состоянием G_0.

В существующих компьютерных системах, в том числе в ОС, параметры системы управления доступом (списки прав доступа к файлам, списки прав доступа или привилегий пользователей или ролей), как правило, достаточно надежно защищены. Возможность изменения этих параметров нарушителем часто является следствием преодоления (взлома) им системы защиты. Кроме того, уровень целостности процессов ОС, например в ОС семейств *Microsoft Windows Vista*/7/2008, устанавливается только при их активизации. При необходимости повышения уровня целостности процесса осуществляется его перезапуск. Таким образом, в первую очередь усилия нарушителя могут быть направлены на несанкционированное повышение им своих полномочий (в том числе, уровня целостности) за счет получения контроля над доверенными процессами ОС. С учетом сказанного в рамках БРОС ДП-модели можно не рассматривать действия по администрированию параметров целостности системы, задаваемых функциями (i_u, i_e, i_r, i_s). Таким образом, по аналогии с БР ДП-моделью используем следующие предположения, при этом дополним предположение 2.

Предположение 6. В рамках БРОС ДП-модели на траекториях функционирования системы не изменяются: значения множеств U, L_U, R, функции UA, AUA, H_R, H_{AR}, i_u, i_r и значения множеств сущностей, параметрически ассоциированных с каждой ролью или административной ролью. Новые значения для функций i_e и i_s задаются только при создании соответствующих новых сущностей или субъект-сессий. Таким образом, будем использовать следующее сокращенное обозначение для состояния системы: $G = (PA, user, roles, A, F, H_E)$.

Предположение 7. Уровень целостности роли не превосходит уровней целостности ролей, которым она подчинена в иерархии ролей. Уровень целостности сущности, входящей в состав сущности-контейнера и не являющейся субъект-сессией, не превосходит уровня целостности сущности-контейнера. Уровни целостности сущностей, параметрически ассоциированных с учетной записью пользователя, совпадают с ее уровнем целостности. Текущий уровень целостности субъект-сессии не превосходит уровня целостности учетной записи пользователя, от имени которой она функционирует, и текущего уровня субъект-сессии, которой она подчинена в иерархии. Уровень целостности роли не может быть выше уровня целостности учетной записи пользователя, которая на нее может быть авторизована, и текущего уровня целостности субъект-сессии, во множество текущих ролей которой она входит. Права доступа владения own_r, на запись $write_r$ и запись в конец $append_r$ к сущности, не являющейся субъект-сессией, могут принадлежать только ролям, имеющим уровень целостности не ниже, чем уровень целостности сущности. Право доступа владения own_r к субъект-сессии может принадлежать только ролям, имеющим уровень целостности не ниже, чем уровень целостности субъект-сессии. Таким образом, выполняются условия:

- для ролей $r, r' \in R \cup AR$: если $r \leqslant r'$, то $i_r(r) \leqslant i_r(r')$;
- для сущностей $e, e' \in E \setminus S$: если $e \leqslant e'$, то $i_e(e) \leqslant i_e(e')$;
- для субъект-сессий $s, s' \in S$: если $s \leqslant s'$, то $i_s(s) \leqslant i_s(s')$;
- для каждой сущности $e \in \rfloor u[$, где $u \in U$, справедливо равенство $i_e(e) = i_u(u)$;
- для субъект-сессии $s \in S$ верно неравенство $i_s(s) \leqslant i_u(user(s))$;
- для учетной записи пользователя $u \in U$ и роли $r \in R$: если $r \in UA(u)$, то $i_r(r) \leqslant i_u(u)$;
- для субъект-сессии $s \in S$ и роли $r \in R$: если $r \in roles(s)$, то $i_r(r) \leqslant i_s(s)$;

— для права доступа к сущности $(e, \alpha) \in P$, где $\alpha \in \{own_r, write_r, append_r\}$, и роли $r \in R$: если $(e, \alpha) \in PA(r)$, то или $e \in E \setminus S$ и $i_e(e) \leqslant i_r(r)$, или $e \in S$, $\alpha = own_r$ и $i_s(e) \leqslant i_r(r)$.

Предположение 8. Учетные записи доверенных субъект-сессий имеют высокий уровень целостности i_high, а недоверенных субъект-сессий — низкий i_low. Во множестве сущностей имеется сущность $i\_entity \in E$, обладающая высоким уровнем целостности, реализация информационного потока по памяти к которой от субъект-сессии является необходимым (в дополнение, в том числе, к требованиям предположения 2) в следующих случаях:

— когда субъект-сессия создает новую субъект-сессию с высоким уровнем целостности;
— когда субъект-сессия берет во множество текущих ролей роль с высоким уровнем целостности;
— когда субъект-сессия создает сущность с высоким уровнем целостности;
— когда субъект-сессия меняет у роли права доступа к сущности с высоким уровнем целостности.

Таким образом, выполняются условия:

— для учетной записи доверенного пользователя $u \in L_U$ верно равенство $i_u(u) = i\_high$;
— для учетной записи недоверенного пользователя $u \in N_U$ верно равенство $i_u(u) = i\_low$;
— верно равенство $i_e(i\_entity) = i\_high$.

Предположение 9. Субъект-сессии могут иметь друг к другу только доступ владения own_a. Роли могут обладать к субъект-сессиям только правом доступа владения own_r. Таким образом, для каждой роли $r \in R$ во множестве ее прав доступа $PA(r)$ отсутствуют права доступа вида (s, α), где $s \in S$ и $\alpha \neq own_r$.

Предположение 10. Если субъект-сессия s реализовала информационный поток по памяти от себя к сущности, функционально ассоциированной с другой субъект-сессией s', или субъект-сессия s реализовала информационный поток по памяти к себе от всех сущностей, параметрически ассоциированных с другой субъект-сессией s', то субъект-сессия s получает доступ владения own_a к субъект-сессии s'.

Предположение 11. Если субъект-сессия s имеет доступ владения own_a к субъект-сессии s', то субъект-сессия s получает следующие возможности:

— использовать роли из множества текущих ролей субъект-сессии s';
— изменять множество текущих ролей субъект-сессии s';
— использовать текущий уровень целостности субъект-сессии s';
— использовать доступы субъект-сессии s';
— получать доступ владения own_a к субъект-сессиям, доступом владения к которым обладает субъект-сессия s';
— использовать административные роли субъект-сессии s' для осуществления действий над ролями и сущностями, которые позволяют ей изменять права доступа ролей субъект-сессии s';
— использовать информационные потоки, в реализации которых участвует субъект-сессия s'.

Используем обозначения:

$de\_facto\_roles : S \to 2^{R \cup AR}$ — функция фактических текущих ролей субъект-сессий, при этом по определению в каждом состоянии системы $G = (PA, user, roles, A,$

F, H_E) для каждой субъект-сессии $s \in S$ верно равенство
$de\_facto\_roles(s_1) = roles(s) \cup \{r \in R \cup AR$: существует $s' \in S$, такая, что $(s, s', own_a) \in A$ и $r \in roles(s')\}$;

$de\_facto\_rights : S \to 2^P$ — функция фактических текущих прав доступа субъект-сессий, при этом по определению в каждом состоянии системы $G = (PA, user, roles, A, F, H_E)$ для каждой субъект-сессии $s \in S$ верно равенство
$de\_facto\_rights(s) = \{p \in P$: существует $r \in de\_facto\_roles(s)$, такая, что $p \in PA(r)\}$;

$de\_facto\_accesses : S \to 2^A$ — функция фактических доступов субъект-сессий, при этом по определению в каждом состоянии системы $G = (PA, user, roles, A, F, H_E)$ для каждой субъект-сессии $s \in S$ верно равенство
$de\_facto\_accesses(s) = \{(s, e, \alpha_a) : (s, e, \alpha_a) \in A\} \cup \{(s', e, \alpha_a) : (s, s', own_a), (s', e, \alpha_a) \in A\}$;

$de\_facto\_actions : S \to S \times U \times 2^P \times 2^R$ — функция фактических возможных действий субъект-сессий, при этом по определению в каждом состоянии системы $G = (PA, user, roles, A, F, H_E)$ для каждой субъект-сессии $s \in S$ верно равенство
$de\_facto\_actions(s) = (\{s\} \times \{user(s)\} \times PA(roles(s)) \times can\_manage\_rights(roles(s) \cap \cap AR)) \cup \{(s', u', PA(roles(s')), can\_manage\_rights(roles(s') \cap AR)) : (s, s', own_a) \in A$, либо $u' = \varnothing$, либо $u' = user(s')$ и для каждой сущности $e \in]u'[$ выполняется условие $(e, s, write_m) \in F\}$.

Определение 7. Будем говорить, что субъект-сессия s фактически обладает ролью r, когда она принадлежит множеству $de\_facto\_roles(s)$. При этом роль r будем называть фактической ролью субъект-сессии s. Права доступа из множества $de\_facto\_rights(s)$ будем называть фактическими текущими правами доступа субъект-сессии s. Доступы из множества $de\_facto\_accesses(s)$ будем называть фактическими доступами субъект-сессии s.

Заметим, что фактические доступы субъект-сессий в рамках БР ДП-модели не рассматривались.

Определение 8. Будем говорить, что субъект-сессия s обладает фактической возможностью осуществить от имени субъект-сессии s' (с учетной записью пользователя u') действие над сущностью y и ролью r с применением права доступа α_r, когда выполняется условие $(s', u', (y, \alpha_r), r) \in de\_facto\_actions(s)$. При этом в случае, когда выполняется условие $(s', \varnothing, \varnothing, \varnothing) \in de\_facto\_actions(s)$, будем говорить, что субъект-сессия s обладает фактическим текущим уровнем целостности $i_s(s')$.

В случае, когда для осуществления субъект-сессией s от имени субъект-сессии s' действия над сущностью y и ролью r с применением права доступа α_r не требуется наличия информационных потоков по памяти от всех сущностей $e \in]user(s')[$, может быть достаточным выполнение условия $(s', \varnothing, (y, \alpha_r), r) \in de\_facto\_actions(s)$.

2. Правила преобразования состояний

В рамках БРОС ДП-модели используются правила преобразования состояний из множества OP, приведенные в следующей таблице.

Правила преобразования состояний БРОС ДП-модели

Правило	Исходное состояние $G = (PA, user, roles, A, F, H_E)$	Результирующее состояние $G' = (PA', user', roles', A', F', H'_E)$
1	2	3
$take\_role(x, w, r)$	$x, w \in S$, $(w, user(w), \varnothing, \varnothing) \in$ $\in de\_facto\_actions(x)$, $r \in UA(user(w)) \cup AUA(user(w))$, для $e \in]r[$ выполняется условие $(e, x, write_m) \in F$, $i_r(r) \leqslant i_s(w)$, и если $i_r(r) = i\_high$, то $(x, i\_entity, write_m) \in F$	$S' = S$, $E' = E$, $PA' = PA$, $user' = user$, $A' = A$, $F' = F$, $H'_E = H_E$, $roles'(w) = roles(w) \cup \{r\}$ и для $s \in S \setminus \{w\}$ выполняется равенство $roles'(s) = roles(s)$, если $x \in (N_S \cup NF_S) \cap S$, то $F' = F \cup \{(x, s, write_t)$: $s \in (N_S \cup NF_S) \cap S$, $x \neq s$ и или $s = w$, или $(w, own_a) \in de\_facto\_accesses(s)\}$, если $x \in LF_S \cap S$, то $F' = F$
$remove\_role(x, w, r)$	$x, w \in S$, $r \in roles(w)$, $(w, user(w), \varnothing, \varnothing) \in de\_facto\_actions(x)$, для $e \in]r[$ выполняется условие $(e, x, write_m) \in F$, и если $i_r(r) = i\_high$, то $(x, i\_entity, write_m) \in F$	$S' = S$, $E' = E$, $PA' = PA$, $user' = user$, $A' = A$, $F' = F$, $H'_E = H_E$, $roles'(w) = roles(w) \setminus \{r\}$ и для $s \in S \setminus \{w\}$ выполняется равенство $roles'(s) = roles(s)$, если $x \in (N_S \cup NF_S) \cap S$, то $F' = F \cup \{(x, s, write_t)$: $s \in (N_S \cup NF_S) \cap S$, $x \neq s$ и или $s = w$, или $(w, own_a) \in de\_facto\_accesses(s)\}$, если $x \in LF_S \cap S$, то $F' = F$
$grant\_right(x, r, (y, \alpha_r))$	$x \in S$, $y \in E$, $(y, \alpha_r) \in P$, $(w, \varnothing, (y, own_r), r) \in de\_facto\_actions(x)$, $i_r(r) \leqslant i_s(w)$, если $y \in S$, то $\alpha_r = own_r$ и $i_s(y) \leqslant i_r(r)$; если $y \in E \setminus S$ и $\alpha_r \in \{own_r, write_r, append_r\}$, то $i_e(y) \leqslant i_r(r)$; если $i_e(y) = i\_high$, то $(x, i\_entity, write_m) \in F$	$S' = S$, $E' = E$, $user' = user$, $roles' = roles$, $A' = A$, $H'_E = H_E$, $PA'(r) = PA(r) \cup \{(y, \alpha_r)\}$, и для $r' \in R \setminus \{r\}$ выполняется равенство $PA'(r') = PA(r')$, если $x \in (N_S \cup NF_S) \cap S$, то $F' = F \cup \{(x, s, write_t) : s \in (N_S \cup NF_S) \cap S$, $x \neq s$ и $r \in de\_facto\_roles(s)\}$, если $x \in LF_S \cap S$, то $F' = F$
$remove\_right(x, r, (y, \alpha_r))$	$x \in S$, $y \in E$, $(y, \alpha_r) \in PA(r)$ и $(w, \varnothing, (y, own_r), r) \in de\_facto\_actions(x)$, $i_r(r) \leqslant i_s(w)$, и если $i_e(y) = i\_high$, то $(x, i\_entity, write_m) \in F$	$S' = S$, $E' = E$, $user' = user$, $roles' = roles$, $A' = A$, $H'_E = H_E$, $PA'(r) = PA(r) \setminus \{(y, \alpha_r)\}$, и для $r' \in R \setminus \{r\}$ выполняется равенство $PA'(r') = PA(r')$, если $x \in (N_S \cup NF_S) \cap S$, то $F' = F \cup \{(x, s, write_t)$: $s \in (N_S \cup NF_S) \cap S$, $x \neq s$ и $r \in de\_facto\_roles(s)\}$, если $x \in LF_S \cap S$, то $F' = F$
$create\_entity(x, r, y, yi, z)$	$x \in S$, $y \notin E$, $z \in E \setminus S$, $(w, \varnothing, (z, \alpha_r), r) \in de\_facto\_actions(x)$, где $\alpha_r \in \{write_r, append_r\}$, $i_r(r) \leqslant i_s(w)$, $i_e(z) \leqslant i_s(w)$, $yi \leqslant i_e(z)$, $yi \leqslant i_r(r)$, если $i_e(z) = i\_high$, то $(x, i\_entity, write_m) \in F$	$S' = S$, $E' = E \cup \{y\}$, при этом $y \notin UE \cup RE$, $user' = user$, $roles' = roles$, $A' = A$, $i'_e(y) = yi$, $H'_E(z) = H_E(z) \cup \{y\}$, $H'_E(y) = \varnothing$, для $e \in E \setminus \{z\}$ выполняется равенство $H'_E(e) = H_E(e)$, $PA'(r) = PA(r) \cup \{(y, own_r)\}$, и для $r' \in R \setminus \{r\}$ выполняется равенство $PA'(r') = PA(r')$, если $x \in (N_S \cup NF_S) \cap S$, то $F' = F \cup \{(x, e, write_t) : e \in E$ и $y \leqslant e\} \cup$ $\cup \{(x, s, write_t) : s \in (N_S \cup NF_S) \cap S$, $x \neq s$ и $r \in de\_facto\_roles(s)\}$, если $x \in LF_S \cap S$, то $F' = F$
$rename\_entity(x, y, z)$	$x \in S$, $y, z \in E \setminus S$, $y \in H_E(z)$, $(w, \varnothing, (z, \alpha_r), r) \in de\_facto\_actions(x)$, где $\alpha_r \in \{write_r, append_r\}$, $i_r(r) \leqslant i_s(w)$, $i_e(z) \leqslant$ $\leqslant i_s(w)$, если $i_e(z) = i\_high$, то $(x, i\_entity, write_m) \in F$	$S' = S$, $E' = E$, $PA' = PA$, $user' = user$, $roles' = roles$, $A' = A$, $H'_E = H_E$, если $x \in (N_S \cup NF_S) \cap S$, то $F' = F \cup$ $\cup \{(x, e, write_t) : e \in E, x \neq e, e \leqslant z\} \cup$ $\cup \{(x, s, write_t) : s \in (N_S \cup NF_S) \cap S, x \neq s$, $(e, \alpha_a) \in de\_facto\_accesses(s)$, где $e \in E$, $e \leqslant y$, $\alpha_a \in R_a\}$, если $x \in LF_S \cap S$, то $F' = F$

Продолжение таблицы

1	2	3
$delete\_entity(x, y, z)$	$x \in S$, $y, z \in E \setminus S$, $y \in H_E(z)$, $\{e \in E : e \leqslant y\} \cap (UE \cup RE) = \varnothing$, $(w, \varnothing, (z, write_r), r) \in de\_facto\_actions(x)$ и $i_r(r) \leqslant i_s(w)$, $i_e(z) \leqslant i_s(w)$, если $i_e(z) = i\_high$, то $(x, i\_entity, write_m) \in F$	$S' = S$, $E' = E \setminus \{e \in E: e \leqslant y\}$, $user' = user$, $roles' = roles$, $H'_E(z) = H_E(z) \setminus \{y\}$, для всех $e \in E' \setminus \{z\}$ выполняется равенство $H'_E(e) = H_E(e)$, для $r \in R$ выполняется равенство $PA'(r) = PA(r) \setminus \{(e, \alpha_r): e \leqslant y, \alpha_r \in R_r\}$, $A' = A \setminus \{(s, e, \alpha_a): s \in S, e \in E, e \leqslant y, \alpha_a \in R_a\}$, если $x \in (N_S \cup NF_S) \cap S$, то $F' = (F \cup \{(x, e, write_t) : e \in E, x \neq ez \leqslant e\} \cup \cup \{(x, s, write_t) : s \in (N_S \cup NF_S) \cap S', x \neq s$ и $(e, \alpha_a) \in de\_facto\_accesses(s)$, где $e \in E$, $e \leqslant y$ и $\alpha_a \in R_a\}) \setminus (\{(e, e', \alpha_f): e, e' \in E, e' \leqslant y, \alpha_f \in R_f\} \cup \{(e', e, \alpha_f): e, e' \in E, e' \leqslant y, \alpha_f \in R_f\})$, если $x \in LF_S \cap S$, то $F' = F$
$create\_first\_session(x, u, r, y, z, zi)$	$x \in S$, $u \in U$, $y \in E$, $z \notin E$, $(y, execute_r) \in PA(UA(u))$ и $r \in can\_manage\_rights(AUA(u))$, $zi \leqslant i_u(u)$, $zi \leqslant i_r(r)$, $\{(e, x, write_m) : e \in]u[\} \subset F$, если $zi = i\_high$, то $(x, i\_entity, write_m) \in F$	$S' = S \cup \{z\}$, $E' = E \cup \{z\}$, $A' = A$, $i'_s(z) = zi$, $user'(z) = u$, $roles'(z) = \varnothing$, для $s \in S$ выполняются равенства $user'(s) = user(s)$, $roles'(s) = roles(s)$, $[z] = fa(u, y)$, $]z[= fp(u, y)$, $H'_E(z) = \varnothing$, для $e \in E$ выполняется равенство $H'_E(e) = H_E(e)$, $PA'(r) = PA(r) \cup \{(z, own_r)\}$, и для $r' \in R \setminus \{r\}$ выполняется равенство $PA'(r') = PA(r')$, если $x \in (N_S \cup NF_S) \cap S$, то $F' = F \cup \cup \{(z, x, write_t), (x, z, write_t)\} \cup \{(x, e, write_t) : e \in E$ и $y \leqslant e\} \cup \{(x, s, write_t) : s \in (N_S \cup \cup NF_S) \cap S$, $x \neq s$ и $r \in de\_facto\_roles(s)\}$, если $x \in LF_S \cap S$, то $F' = F$
$create\_session(x, w, r, y, z, zi)$	$x, w \in S$, $y \in E$, $z \notin E$, $(w, \varnothing, (y, execute_r), r) \in de\_facto\_actions(x)$, $zi \leqslant i_r(r) \leqslant i_s(w)$	$S' = S \cup \{z\}$, $E' = E \cup \{z\}$, $A' = A$, $i'_s(z) = zi$, $user'(z) = user(w)$, $roles'(z) = \varnothing$, для $s \in S$ выполняются равенства $user'(s) = user(s)$, $roles'(s) = roles(s)$, $[z] = fa(user(w), y)$, $]z[= fp(user(w), y)$, $H'_E(w) = H_E(w) \cup \{z\}$, $H'_E(z) = \varnothing$, для $e \in E \setminus \{w\}$ выполняется равенство $H'_E(e) = = H_E(e)$, $PA'(r) = PA(r) \cup \{(z, own_r)\}$, и для $r' \in R \setminus \{r\}$ выполняется равенство $PA'(r') = PA(r')$, если $x \in (N_S \cup NF_S) \cap S$, то $F' = F \cup \{(z, x, write_t), (x, z, write_t)\} \cup \cup \{(x, e, write_t) : e \in E$ и $y \leqslant e\} \cup \{(x, s, write_t) : s \in (N_S \cup NF_S) \cap S, x \neq s$ и $r \in de\_facto\_roles(s)\}$, если $x \in LF_S \cap S$, то $F' = F$

Продолжение таблицы

1	2	3
$delete\_session(x, w, z)$	$x, w, z \in S$, $(w, \varnothing, (z, own_r), \varnothing) \in de\_facto\_actions(x)$	$S' = S \setminus \{z\}$, $E' = E \setminus \{z\}$, для $s \in S'$ выполняются равенства $user'(s) = user(s)$, $roles'(s) = roles(s)$, для $z' \in S$, такого, что $z \in H_E(z')$, справедливо равенство $H'_E(z') = (H_E(z') \setminus \{z\}) \cup H_E(z)$, при этом выполняется условие: для $e \in E' \setminus \{z'\}$ выполняется равенство $H'_E(e) = H_E(e)$, $PA'(r) = PA(r) \setminus \{(z, own_r)\}$, и для $r' \in R \setminus \{r\}$ выполняется равенство $PA'(r') = PA(r')$, $A' = A \setminus (\{(z, e, \alpha_a)\colon e \in E, \alpha_a \in R_a\} \cup \{(s, z, own_a)\colon s \in S\})$, если $x \in (N_S \cup NF_S) \cap S$, то $F' = (F \cup \cup\{(x, s, write_t) : e \in E, x \neq e \text{ и } z < e\} \cup \cup\{(x, s, write_t) : s \in (N_S \cup NF_S) \cap S, x \neq s \text{ и } (z, own_a) \in de\_facto\_accesses(s)\}) \setminus \setminus(\{(z, e, \alpha_f) : e \in E, \alpha_f \in R_f\} \cup \{(e, z, \alpha_f)\colon e \in E, \alpha_f \in R_f\})$, если $x \in LF_S \cap S$, то $F' = F$
$control(x, y, z)$	$x, y \in S$, $x \neq y$, $z \in [y]$ и или $x = z$, или $(x, z, write_m) \in F$, или $z \in S$ и $(x, z, own_a) \in A$	$S' = S$, $E' = E$, $PA' = PA$, $user' = user$, $roles' = roles$, $H'_E = H_E$, $A' = A \cup \{(x, y, own_a)\}$, если $x \in (N_S \cup NF_S) \cap S$, то $F' = F \cup \{(x, e, write_t)\colon e \in E, x \neq e \text{ и } y \leqslant e\}$, если $x \in LF_S \cap S$, то $F' = F$
$know(x, y)$	$x, y \in S$, $x \neq y$, и для каждой $e \in]y[$ существует $(e, x, write_m) \in F$	$S' = S$, $E' = E$, $PA' = PA$, $user' = user$, $roles' = roles$, $H'_E = H_E$, $A' = A \cup \{(x, y, own_a)\}$, если $x \in (N_S \cup NF_S) \cap S$, то $F' = F \cup \{(x, e, write_t)\colon e \in E, x \neq e \text{ и } y \leqslant e\}$, если $x \in LF_S \cap S$, то $F' = F$
$take\_access\_own(x, y, z)$	$x, y, z \in S$, $x \neq z$, $\{(x, y, own_a), (y, z, own_a)\} \subset A$	$S' = S$, $E' = E$, $PA' = PA$, $user' = user$, $roles' = roles$, $H'_E = H_E$, $A' = A \cup \{(x, z, own_a)\}$, если $x \in (N_S \cup NF_S) \cap S$, то $F' = F \cup \{(x, e, write_t)\colon e \in E, x \neq e \text{ и } z \leqslant e\}$, если $x \in LF_S \cap S$, то $F' = F$
$access\_own(x, w, y)$	$x, w, y \in S$, $w \neq y$, $(w, \varnothing, (y, own_r), \varnothing) \in de\_facto\_actions(x)$	$S' = S$, $E' = E$, $PA' = PA$, $user' = user$, $roles' = roles$, $H'_E = H_E$, $A' = A \cup \{(w, y, own_a)\}$, если $x \in (N_S \cup NF_S) \cap S$, то $F' = F \cup \cup\{(x, e, write_t) : e \in E, x \neq e \text{ и } y \leqslant e\} \cup \cup\{(x, s, write_t) : s \in (N_S \cup NF_S) \cap S, x \neq s \text{ и } (y, own_a) \in de\_facto\_accesses(s)\}$, если $x \in LF_S \cap S$, то $F' = F$
$access\_read(x, w, y)$	$x, w \in S$, $y \in E \setminus S$, $(w, \varnothing, (y, read_r), \varnothing) \in de\_facto\_actions(x)$	$S' = S$, $E' = E$, $PA' = PA$, $user' = user$, $roles' = roles$, $H'_E = H_E$, $A' = A \cup \{(w, y, read_a)\}$, если $x \in (N_S \cup NF_S) \cap S$, то $F' = F \cup \{(y, w, write_m)\} \cup \{(x, e, write_t)\colon e \in E, x \neq e \text{ и } y \leqslant e\}$, если $x \in LF_S \cap S$, то $F' = F \cup \{(y, w, write_m)\}$

Окончание таблицы

1	2	3
$access\_write(x, w, y)$	$x, w \in S$, $y \in E \setminus S$, $(w, \varnothing, (y, write_r), \varnothing) \in de\_facto\_actions(x)$	$S' = S$, $E' = E$, $PA' = PA$, $user' = user$, $roles' = roles$, $H'_E = H_E$, $A' = A \cup \{(w, y, write_a)\}$, если $x \in (N_S \cup NF_S) \cap S$, то $F' = F \cup \{(w, y, write_m)\} \cup \{(x, e, write_t)$: $e \in E$, $x \neq e$ и $y \leqslant e\}$, если $x \in LF_S \cap S$, то $F' = F \cup \{(w, y, write_m)\}$
$access\_append(x, w, y)$	$x, w \in S$, $y \in E \setminus S$, $(w, \varnothing, (y, append_r), \varnothing) \in de\_facto\_actions(x)$	$S' = S$, $E' = E$, $PA' = PA$, $user' = user$, $roles' = roles$, $H'_E = H_E$, $A' = A \cup \{(w, y, append_a)\}$, если $x \in (N_S \cup NF_S) \cap S$, то $F' = F \cup \{(w, y, write_m)\} \cup \{(x, e, write_t)$: $e \in E$, $x \neq e$ и $y \leqslant e\}$, если $x \in LF_S \cap S$, то $F' = F \cup \{(w, y, write_m)\}$
$flow(x, y, y', z)$	$x, z \in S$, $y, y' \in E$, $x \neq z$, $y \leqslant y'$, [или $x = y$, или $(y, \alpha_a) \in de\_facto\_accesses(x)$], [или $z = s'$, или $(y', \beta_a) \in \in de\_facto\_accesses(z)$], где α_a, $\beta_a \in R_a$	$S' = S$, $E' = E$, $PA' = PA$, $user' = user$, $roles' = roles$, $A' = A$, $H'_E = H_E$, если x, $z \in (N_S \cup NF_S) \cap S$, то $F' = F \cup \{(x, z, write_t), (z, x, write_t)\}$, если $\{x,z\} \cap LF_S \cap S) \neq \varnothing$, то $F' = F$
$find(x, y, z)$	$x, y \in S$, $z \in E$, $x \neq z$, $(x, y, \alpha) \in F$, где $\alpha \in \{write_m, write_t\}$, и [или $(z, \beta) \in de\_facto\_accesses(y)$, где $\beta \in \{write_a, append_a\}$], [или $(y, z, \beta) \in F$, где $\beta \in \{write_m, write_t\}$]	$S' = S$, $E' = E$, $PA' = PA$, $user' = user$, $roles' = roles$, $A' = A$, $H'_E = H_E$, если $write_t \notin \{\alpha, \beta\}$, то $F' = F \cup \{(x, z, write_m)\}$, если $write_t \in \{\alpha, \beta\}$ и $x, y \in (N_S \cup NF_S) \cap S$, то $F' = F \cup \{(x, z, write_t)\}$, если $write_t \in \{\alpha, \beta\}$ и $\{x, y\} \cap (LF_S \cap S) \neq \varnothing$, то $F' = F$
$post(x, y, z)$	$x, z \in S$, $y \in E$, $x \neq z$, $(y, read_a) \in de\_facto\_accesses(z)$ и [или $(y, \alpha) \in de\_facto\_accesses(x)$, где $\alpha \in \{write_a, append_a\}$], [или $(x, y, \alpha) \in F$, где $\alpha \in \{write_m, write_t\}$]	$S' = S$, $E' = E$, $PA' = PA$, $user' = user$, $roles' = roles$, $A' = A$, $H'_E = H_E$, если $\alpha \neq write_t$, то $F' = F \cup \{(x, z, write_m)\}$, если $\alpha = write_t$ и $x, z \in (N_S \cup NF_S) \cap S$, то $F' = F \cup \{(x, z, write_t)\}$, если $\alpha = write_t$ и $\{x, z\} \cap (LF_S \cap S) \neq \varnothing$, то $F' = F$
$pass(x, y, z)$	$y \in S$, $x, z \in E$, $x \neq z$, $(x, read_a) \in de\_facto\_accesses(y)$ и [или $(z, \alpha) \in de\_facto\_accesses(y)$, где $\alpha \in \{write_a, append_a\}$], [или $(y, z, \alpha) \in F$, где $\alpha \in \{write_m, write_t\}$]	$S' = S$, $E' = E$, $PA' = PA$, $user' = user$, $roles' = roles$, $A' = A$, $H'_E = H_E$, если $\alpha \neq write_t$, то $F' = F \cup \{(x, z, write_m)\}$, если $\alpha = write_t$ и $y \in (N_S \cup NF_S) \cap S$, то $F' = F \cup \{(x, z, write_t)\}$, если $\alpha = write_t$ и $y \in LF_S \cap S$ то $F' = F$
$take\_flow(x, y)$	$x, y \in S, x \neq y, (x, y, own_a) \in A$	$S' = S$, $E' = E$, $PA' = PA$, $user' = user$, $roles' = roles$, $A' = A$, $H'_E = H_E$, если $x \in (N_S \cup NF_S) \cap S$, то $F' = F \cup \{(x, e, \alpha)$: $(y, e, \alpha) \in F$, $e \in E$, $\alpha \in \{write_m, write_t\}\}$, если $x \in LF_S \cap S$, то $F' = F \cup \{(x, e, write_m)$: $(y, e, write_m) \in F$, $e \in E\}$

Рассмотрим условия и результаты применения правил преобразования состояний БРОС ДП-модели (их зависимость показана на рис. 1) в первую очередь с целью анализа их основных отличий от аналогичных правил БР ДП-модели.

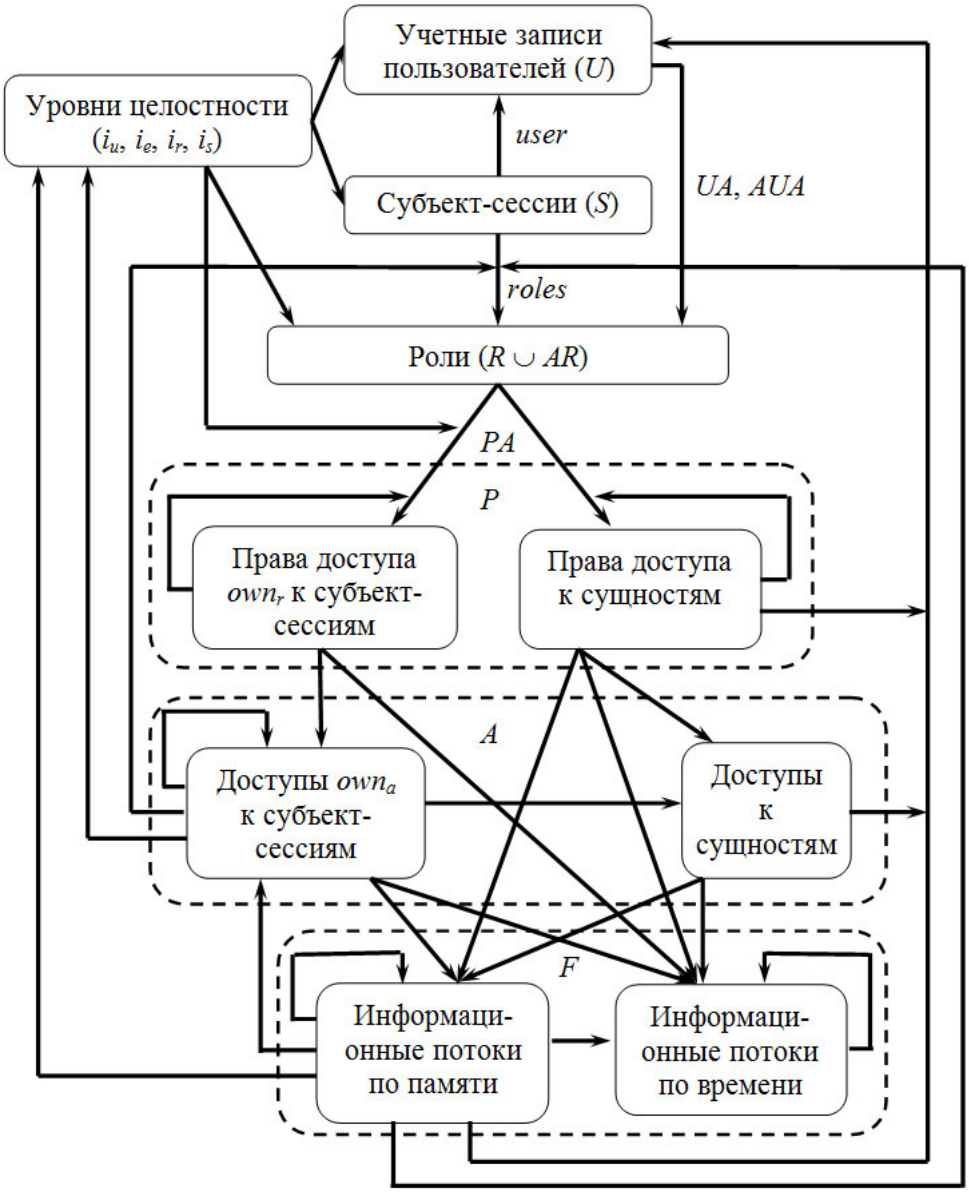

Рис. 1. Зависимость условий и результатов применения правил преобразования состояний БРОС ДП-модели

Правила $take\_role(x, w, r)$ и $remove\_role(x, w, r)$ позволяют субъект-сессии x добавить (удалить) роль r, на которую субъект-сессия может быть авторизована, либо в множество своих текущих ролей, либо, при наличии доступа владения к другой субъект-сессии w, в множество её текущих ролей. При этом в соответствии с предположением 7 при добавлении роли необходимо, чтобы ее уровень целостности не превосходил текущего уровня целостности субъект-сессии, которая ее получает. Таким образом, если в БР ДП-модели предполагалось, что изменять множество текущих ролей субъект-сессии может только сама субъект-сессия, то в рамках БРОС ДП-модели в соответствии с предположением 11 субъект-сессия может изменять множество своих фактических ролей. Для этого требуется, чтобы субъект-сессия x, получившая контроль над субъект-сессией w, реализовала к себе информационные потоки по памяти от всех сущностей учетной записи пользователя субъект-сессии w; в случае, когда

уровень целостности роли r равняется i_high, в соответствии с предположением 8 требуется реализация субъект-сессией x информационного потока по памяти от себя к сущности i_entity. Также при добавлении или удалении роли дополнительно (по сравнению с БР ДП-моделью) требуется, чтобы субъект-сессия x реализовала к себе информационные потоки от всех сущностей, параметрически ассоциированных с ролью r. Кроме того, так как добавление или удаление роли из множества текущих ролей субъект-сессии может быть использовано для передачи данных, то при реализации правил возможно возникновение информационных потоков по времени от субъект-сессии x к субъект-сессии w и ко всем субъект-сессиям, обладающим к ней фактическим доступом владения.

Правила $grant\_right(x, r, (y, \alpha_r))$ и $remove\_right(x, r, (y, \alpha_r))$ позволяют субъект-сессии x добавить или удалить соответственно право доступа α_r к сущности y из множества прав доступа роли r. В соответствии с предположением 11 для применения правил необходимо наличие у субъект-сессии x фактической возможности осуществить действие над сущностью y и ролью r с применением права доступа владения own_r. Кроме того, в соответствии с предположением 7 для применения правил требуется, чтобы уровень целостности роли не превосходил текущего уровня целостности субъект-сессии, от имени которой добавляется или удаляется право доступа. Если осуществляется добавление права доступа own_r, $write_r$ или $append_r$ к сущности, то требуется, чтобы уровень целостности сущности y (если y является субъект-сессией, то текущий уровень целостности y) не превосходил уровня целостности роли r. При этом в случае, когда уровень целостности сущности y равняется i_high, в соответствии с предположением 8 требуется реализация субъект-сессией x информационного потока по памяти от себя к сущности i_entity. Так как добавление или удаление права доступа из множества прав доступа роли r может быть использовано для передачи данных, то в результате применения правил могут возникнуть информационные потоки по времени от субъект-сессии x к субъект-сессиям, фактически обладающим ролью r.

Правила $create\_entity(x, r, y, yi, z)$, $rename\_entity(x, y, z)$ и $delete\_entity(x, y, z)$ позволяют субъект-сессии x создать, переименовать или удалить соответственно сущность y, не являющуюся субъект-сессией и входящую в состав контейнера z. При этом нельзя удалять или создавать сущности, входящие в состав учетных записей пользователей, или сущности, параметрически ассоциированные с ролями, и в отличие от БР ДП-модели для применения правил в соответствии с предположением 7 требуется, чтобы уровень целостности роли, используемой для создания, переименования или удаления сущности, и уровень целостности сущности-контейнера z, в котором содержится сущность y, не превосходили текущего уровня целостности субъект-сессии, от имени которой данные действия осуществляются. Если выполняется создание, переименование или удаление сущности y с уровнем целостности i_high, то в соответствии с предположением 8 требуется реализация субъект-сессией x информационного потока по памяти от себя к сущности i_entity. Кроме того, в соответствии с предположением 7 при создании сущности требуется, чтобы уровень целостности yi сущности y не превосходил уровня целостности сущности-контейнера z и уровня целостности роли r, которая получает право доступа владения к сущности y. Также следует заметить, что при применении правил $rename\_entity(x, y, z)$ или $delete\_entity(x, y, z)$ возникают информационные потоки по времени от субъект-сессии x к субъект-сессиям, имеющим текущие фактические доступы (а не фактические текущие права доступа, как в БР ДП-модели) к сущностям, подчиненным в иерархии сущности y.

Правило $create\_first\_session(x, u, r, y, z, zi)$ позволяет субъект-сессии x с использованием сущности y и учетной записи пользователя u создать от имени u новую субъект-сессию z с текущим уровнем целостности zi (в БР ДП-модели создание субъект-сессии осуществлял пользователь u). При этом в соответствии с предположением 2 для применения правила необходима реализация субъект-сессией x к себе информационных потоков по памяти от всех сущностей, параметрически ассоциированных с учетной записью пользователя u. Кроме того, в соответствии с предположением 7 требуется, чтобы текущий уровень целостности zi создаваемой субъект-сессии z не превосходил уровня целостности учетной записи пользователя u и уровня целостности роли r, которая получает право доступа владения к субъект-сессии z. Однако уровень целостности субъект-сессии z может превосходить уровень целостности субъект-сессии x, что соответствует подходу, реализованному, например, в ОС семейств *Microsoft Windows Vista*/7/2008. В случае, когда субъект-сессия z создается с уровнем целостности i_high, в соответствии с предположением 8 требуется реализация субъект-сессией x информационного потока по памяти от себя к сущности i_entity.

Правила $create\_session(x, w, r, y, z, zi)$ и $delete\_session(x, w, z)$ позволяют субъект-сессии x непосредственно либо от имени субъект-сессии w, к которой субъект-сессия x имеет доступ владения, создать или удалить соответственно субъект-сессию z. При этом в соответствии с предположением 7 требуется, чтобы текущий уровень целостности zi субъект-сессии z не превосходил уровня целостности роли r, получающей право доступа владения к субъект-сессии z, который, в свою очередь, не должен превосходить текущего уровня целостности субъект-сессии w, осуществляющей создание субъект-сессии z. Правило $delete\_session(x, w, z)$ не рассматривалось в БР ДП-модели (кроме того, в дискреционных или мандатных ДП-моделях также отсутствовало правило удаления субъектов), для удаления субъект-сессий в этой модели использовалось правило удаления сущностей $delete\_entity(x, y, z)$. Условия применения правила $delete\_session(x, w, z)$ позволяют учесть, что удаление субъект-сессии z может быть осуществлено субъект-сессией x от имени субъект-сессии w, обладающей правом доступа владения к субъект-сессии z.

Правила $control(x, y, z)$ и $know(x, y)$ в рамках предположений 3 и 4 позволяют субъект-сессии x получить доступ владения к субъект-сессии y с использованием соответственно либо информационного потока по памяти от себя к сущности z, функционально ассоциированной с субъект-сессией y, либо информационных потоков по памяти к себе от всех сущностей, параметрически ассоциированных с субъект-сессией y. Параметрически ассоциированные сущности и правило $know(x, y)$ не рассматривались в БР ДП-модели (они анализировались в дискреционных ФПАС и ФС ДП-моделях [3, 4]).

Условия и результаты применения правила $take\_access\_own(x, y, z)$ не изменились по сравнению с БР ДП-моделью. Оно позволяет субъект-сессии x получить доступ владения к субъект-сессии z (при этом требуется, чтобы субъект-сессия x обладала доступом владения к субъект-сессии y, имеющей, в свою очередь, доступ владения к субъект-сессии z).

Правила $access\_own(x, w, y)$, $access\_read(x, w, y)$, $access\_write(x, w, y)$ и $access\_append(x, w, y)$ позволяют субъект-сессии x, обладающей фактической возможностью осуществить от имени субъект-сессии w с соответствующим правом доступа действие над сущностью y, либо получить самой (когда $x = w$), либо инициировать получение субъект-сессией w соответствующего доступа к сущности y. Данные правила отличаются от аналогичных правил БР ДП-модели тем, что субъект-сессия x не непосред-

ственно сама получает доступ к сущности y, а это делает субъект-сессия w, доступ владения к которой имеет субъект-сессия x.

Правила $flow(x, y, y', z)$, $find(x, y, z)$, $post(x, y, z)$, $pass(x, y, z)$ и $take\_flow(x, y)$ позволяют субъект-сессиям создавать информационные потоки по памяти или по времени. Отличие условий применения данных правил от их аналогов в БР ДП-модели заключается в том, что вместо обладания фактическими текущими правами доступа для реализации информационных потоков от субъект-сессий требуется наличие соответствующих фактических доступов. Кроме того, при применении правила $flow(x, y, y', z)$ для создания информационных потоков по времени допускаются равенства $x = y$ или $y' = z$.

3. Монотонные и немонотонные правила преобразования состояний

По аналогии с дискреционными и мандатными ДП-моделями, а также с БР ДП-моделью дадим определение.

Определение 9. Монотонное правило преобразования состояний — это правило преобразования состояний из множества OP, применение которого не приводит к удалению из состояний

— ролей из множества текущих ролей субъект-сессий;
— прав доступа ролей к сущностям;
— субъект-сессий или сущностей;
— доступов субъект-сессий к сущностям;
— информационных потоков.

По определению 9 и в соответствии с условиями и результатами применения правил преобразования состояний, заданных в таблице, монотонными являются следующие правила: $take\_role(x, w, r)$, $grant\_right(x, r, (y, \alpha_r))$, $create\_entity(x, r, y, yi, z)$, $rename\_entity(x, y, z)$, $create\_first\_session(x, u, r, y, z, zi)$, $create\_session(x, w, r, y, z, zi)$, $control(x, y, z)$, $know(x, y)$, $take\_access\_own(x, y, z)$, $access\_own(x, w, y)$, $access\_read(x, w, y)$, $access\_write(x, w, y)$, $access\_append(x, w, y)$, $flow(x, y, y', z)$, $find(x, y, z)$, $post(x, y, z)$, $pass(x, y, z)$ и $take\_flow(x, y)$. Немонотонными правилами преобразования состояний являются $remove\_role(x, w, r)$, $remove\_right(x, r, (y, \alpha_r))$, $delete\_entity(x, y, z)$, $delete\_session(x, w, z)$.

Поскольку в рамках БРОС ДП-модели условия и результаты применения правил преобразования состояний по сравнению с БР ДП-моделью претерпели существенные изменения, целесообразно обосновать следующее утверждение.

Утверждение 1. Пусть $G_0 = (PA_0, user_0, roles_0, A_0, F_0, H_{E_0})$ — начальное состояние системы $\Sigma(G^*, OP, G_0)$. Пусть также существуют состояния системы $G_1, \ldots, G_N = (PA_N, user_N, roles_N, A_N, F_N, H_{E_N})$ и правила преобразования состояний $op_1, \ldots, op_N$, такие, что $G_0 \vdash_{op_1} G_1 \vdash_{op_2} \ldots \vdash_{op_N} G_N$, где $N \geqslant 0$. Тогда существуют состояния $G'_1, \ldots, G'_M = (PA'_M, user'_M, roles'_M, A'_M, F'_M, H'_{E_M})$, где $M \geqslant 0$, и монотонные правила преобразования состояний $op'_1, \ldots, op'_M$, такие, что $G_0 \vdash_{op'_1} G'_1 \vdash_{op'_2} \ldots \vdash_{op'_M} G'_M$, и выполняются следующие условия.

1. Верно включение $S_N \subset S'_M$ и для каждой субъект-сессии $s \in S_N$ выполняются условия: $user_N(s) = user'_M(s)$, $roles_N(s) \subset roles'_M(s)$.

2. Верно включение $E_N \subset E'_M$, для каждой сущности $e \in E_N \setminus S_N$, не являющейся субъектом, выполняется условие $H_{E_N}(e) \subset H'_{E_M}(e)$, и для любых сущностей $e, e' \in E_N$ если в состоянии G_N выполняется условие $e < e'$, то данное условие выполняется в состоянии G'_M.

3. Для каждой роли $r \in R$ выполняется условие $PA_N(r) \subset PA'_M(r)$.
4. Верно включение $A_N \subset A'_M$.
5. Верно включение $F_N \subset F'_M$.

Доказательство. Пусть существуют состояния G_0, G_1, …, G_N системы $\Sigma(G^*, OP, G_0)$ и правила преобразования состояний op_1, …, op_N, такие, что $G_0 \vdash_{op_1} G_1 \vdash_{op_2} \ldots \vdash_{op_N} G_N$, где $N \geqslant 0$. Докажем утверждение индукцией по длине N последовательности состояний.

Пусть $N = 0$. Тогда положим $M = 0$, $G'_0 = G_0$ и для состояний G_0 и G'_0 условия 1–5 утверждения выполнены.

Пусть $N > 0$ и условия утверждения выполнены для всех последовательностей состояний длины $0 \leqslant L < N$. Докажем, что условия 1–5 утверждения выполнены для последовательностей состояний длины N. По предположению индукции, для последовательности состояний $G_0, G_1, \ldots, G_{N-1}$ существует последовательность $G_0, G'_1, \ldots, G'_K = (PA'_K, user'_K, roles'_K, A'_K, F'_K, H'_{E_K})$, где $K \geqslant 0$, и для состояний G_{N-1} и G'_K выполнены условия 1–5 утверждения. Рассмотрим правило преобразования состояний op_N. Если условия его применения не выполняются в состоянии G_{N-1}, то по определению справедливо равенство $G_{N-1} = G_N$. Положим $M = K$; для состояний G_N и G'_M выполнены условия 1–5 утверждения. Пусть условия применения правила op_N выполняются в состоянии G_{N-1}. Тогда возможны два случая.

П е р в ы й с л у ч а й: правило преобразования состояний op_N является монотонным. Так как для состояний G_{N-1} и G'_K выполнены условия 1–5 утверждения и, по предположению 6, на траекториях системы не изменяются значения функций, задающих уровни целостности сущностей или субъект-сессий, существовавших в предшествующих состояниях системы, то в состоянии G'_K выполняются условия применения правила op_N. Тогда положим $M = K + 1$, $op'_M = op_N$, и пусть состояние G'_M получено из состояния G'_K применением к нему правила op_N: $G'_K \vdash_{op_N} G'_M$. В соответствии с заданными в таблице результатами применения правил преобразования для состояний G_N и G'_M выполнены условия 1–5 утверждения.

В т о р о й с л у ч а й: правило преобразования состояний op_N является немонотонным.

Пусть $op_N = remove\_role(x, w, r)$. Так как для состояний G_{N-1} и G'_K выполнены условия 1–5 утверждения, то выполняются условия $(w, user'_K(w), \varnothing, \varnothing) \in de\_facto\_actions'_K(x)$, для $e \in]r[$ выполняется условие $(e, x, write_m) \in F'_K$, и если $i_r(r) = i\_high$, то $(x, i\_entity, write_m) \in F'_K$. Кроме того, так как в состоянии G_{N-1} выполняется условие $r \in roles_{N-1}(w)$, то по предположению 7 в состоянии G'_K справедливо $r \in UA'_K(user'_K(w)) \cup AUA'_K(user'_K(w))$ и $i_r(r) \leqslant i_s(w)$. Следовательно, в состоянии G'_K выполнены условия применения правила $take\_role(x, w, r)$. Тогда положим $M = K+1$, $op'_M = take\_role(x, w, r)$, и пусть состояние G'_M получено из состояния G'_K применением к нему правила op'_M: $G'_K \vdash_{op'_M} G'_M$. В соответствии с заданными в таблице результатами применения правил преобразования состояний $remove\_role(x, w, r)$ и $take\_role(x, w, r)$ для состояний G_N и G'_M выполнены условия 1–5 утверждения.

Пусть $op_N = remove\_right(x, r, (y, \alpha_r))$. Так как для состояний G_{N-1} и G'_K выполнены условия 1–5 утверждения, то выполняются условия $(w, \varnothing, (y, own_r), r) \in de\_facto\_actions'_K(x)$, $i_r(r) \leqslant i_s(w)$, и если $i_e(y) = i\_high$, то $(x, i\_entity, write_m) \in F'_K$. При этом если $\alpha_r \in \{own_r, write_r, append_r\}$, то по предположению 7 в состоянии G'_K выполняется условие $i_r(r) \leqslant i_s(w)$. Следовательно, в состоянии G'_K выполнены условия применения правила $grant\_right(x, r, (y, \alpha_r))$. Тогда положим $M = K + 1$, $op'_M = grant\_right(x, r, (y, \alpha_r))$, и пусть состояние G'_M получено из состояния G'_K при-

менением к нему правила $op'_M : G'_K \vdash_{op'_M} G'_M$. В соответствии с результатами применения правил преобразования $remove\_right(x, r, (y, \alpha_r))$ и $grant\_right(x, r, (y, \alpha_r))$ для состояний G_N и G'_M выполнены условия 1–5 утверждения.

Пусть $op_N = delete\_entity(x, y, z)$. Так как для состояний G_{N-1} и G'_K выполнены условия 1–5 утверждения, то выполняются условия $(w, \varnothing, (z, write_r), r) \in de\_facto\_actions'_K(x)$, $i_r(r) \leqslant i_s(w)$, $i_e(z) \leqslant i_s(w)$, и если $i_e(z) = i\_high$, то $(x, i\_entity, write_m) \in F'_K$. Следовательно, в состоянии G'_K выполнены условия применения правил $rename\_entity(x, y, z)$ и $access\_write(x, w, z)$. Тогда положим $M = K+2$, $op'_{M-1} = rename\_entity(x, y, z)$, $op'_M = access\_write(x, w, z)$, и пусть состояние G'_M получено из состояния G'_K применением к нему правил op'_{M-1}, op'_M: $G'_K \vdash_{op'_{M-1}} G'_{M-1} \vdash_{op_M} G'_M$. В соответствии с заданными в таблице результатами применения правил преобразования состояний $delete\_entity(x, y, z)$, $rename\_entity(x, y, z)$ и $access\_write(x, w, z)$ для состояний G_N и G'_M выполнены условия 1–5 утверждения.

Пусть $op_N = delete\_session(x, w, z)$. Так как для состояний G_{N-1} и G'_K выполнены условия 1–5 утверждения, то выполняется условие $(w, \varnothing, (z, own_r), \varnothing) \in de\_facto\_actions'_K(x)$. Следовательно, в состоянии G'_K выполнены условия применения правила $access\_own(x, w, z)$. Тогда положим $M = K + 1$, $op'_M = access\_own(x, w, z)$, и пусть состояние G'_M получено из состояния G'_K применением к нему правила op'_M: $G'_K \vdash_{op'_M} G'_M$. В соответствии с заданными в таблице результатами применения правил преобразования состояний $delete\_session(x, w, z)$ и $access\_own(x, w, z)$ для состояний G_N и G'_M выполнены условия 1–5 утверждения.

Следовательно, условия 1–5 утверждения выполнены для последовательностей состояний длины N, и шаг индукции доказан. Утверждение доказано. ■

Таким образом, несмотря на то, что в БРОС ДП-модель добавлены по сравнению с БР ДП-моделью несколько существенно новых элементов (учетные записи пользователей, сущности, параметрически ассоциированные с субъект-сессиями или ролями, мандатный контроль целостности, фактические доступы субъект-сессий), в ее рамках при анализе условий передачи прав доступа, реализации информационных потоков по памяти или по времени также возможно использование только монотонных правил преобразования состояний. В дальнейшем предполагается построение на основе БРОС ДП-модели новых ролевых ДП-моделей, позволяющих учесть особенности функционирования современных ОС, особенно ОС семейства *Linux*.

ЛИТЕРАТУРА

1. *Девянин П. Н.* Анализ в рамках базовой ролевой ДП-модели безопасности систем с простыми траекториями функционирования // Прикладная дискретная математика. 2010. № 1(7). С. 16–36.
2. *Девянин П. Н.* Модели безопасности компьютерных систем. Управление доступом и информационными потоками. Учеб. пособие для вузов. М.: Горячая линия-Телеком, 2011. 320 с.
3. *Колегов Д. Н.* ДП-модель компьютерной системы с функционально и параметрически ассоциированными с субъектами сущностями // Вестник Сибирского государственного аэрокосмического университета им. акад. М. Ф. Решетнева. 2009. Вып. 1(22). Ч. 1. С. 49–54.
4. *Буренин П. В.* Подходы к построению ДП-модели файловых систем // Прикладная дискретная математика. 2009. № 1(3). С. 93–112.

МАТЕМАТИЧЕСКИЕ ОСНОВЫ ИНФОРМАТИКИ И ПРОГРАММИРОВАНИЯ

DOI 10.17223/20710410/11/7

УДК 519.7

ТЕХНОЛОГИЯ ТРАНСЛЯЦИИ КОМБИНАТОРНЫХ ПРОБЛЕМ В БУЛЕВЫ УРАВНЕНИЯ

И. В. Отпущенников, А. А. Семёнов

Институт динамики систем и теории управления СО РАН, г. Иркутск, Россия

E-mail: otilya@yandex.ru, biclop@rambler.ru

Рассматриваются проблемы сведения некоторых комбинаторных задач к задачам поиска решений булевых уравнений. Приведены теоретические результаты, являющиеся базой технологии пропозиционального кодирования алгоритмов, вычисляющих дискретные функции. Описан программный комплекс Transalg — многофункциональный транслятор комбинаторных проблем в булевы уравнения. Приведены примеры использования комплекса Transalg для сведения задач криптоанализа к SAT-задачам. Рассмотрены основы техники трансляции оптимизационных задач 0-1–ЦЛП в SAT-задачи.

Ключевые слова: *дискретные функции, булевы уравнения, криптоанализ, пропозициональное кодирование.*

Введение

Как известно, многие NP-трудные задачи возникли из совершенно конкретных практических постановок. В ряде направлений без умения решать данные задачи невозможно обойтись. Речь идет о проблемах синтеза и верификации дискретных управляющих/управляемых систем, проблемах экономики, производственного планирования, логистики и многих других. В этих областях вопросы построения практически эффективных алгоритмов решения соответствующих комбинаторных задач чрезвычайно актуальны. Большое число исследований посвящено построению приближенных алгоритмов решения NP-трудных проблем. Однако в некоторых приложениях, например в задачах верификации микросхем, требуется находить именно точные решения. Эта необходимость существенно сужает класс методов — методы непрерывной математики, генетические алгоритмы, разнообразные «эволюционные эвристики» в общем случае не дают точных решений.

В настоящей работе приведены основы «пропозиционального подхода» к решению комбинаторных задач из весьма широкого класса. Данный подход предполагает нахождение точных решений и включает две составляющих. Во-первых, это алгоритмы сведения комбинаторных задач к булевым уравнениям, и, во-вторых, это символьные алгоритмы поиска решений получаемых уравнений. В последние годы интерес именно к такому рассмотрению комбинаторных задач заметно усилился в связи с существенным прогрессом в алгоритмике булевых решателей (в первую очередь, SAT-задач), а также в связи с бурным развитием параллельных вычислительных технологий (булевы задачи допускают естественные формы параллелизма).

Библиография по решению булевых уравнений весьма обширна — от фундаментальной монографии С. Рудяну [1] до многочисленных в последние годы работ по SAT-задачам. Работ, специально посвященных сведению комбинаторных проблем к булевым уравнениям, сравнительно мало (можно сослаться на обзорную статью [2] и список литературы к ней). Удивительно то, что в подавляющем большинстве эти результаты имеют характер наглядных примеров и правдоподобных рассуждений — самой строгой в этом смысле продолжает оставаться процедура, фигурирующая в оригинальном и последующих доказательствах теоремы Кука [3, 4]. Следует отметить, что реализовать на основе перечисленных результатов конкретные процедуры сведения, применимые к достаточно широким классам комбинаторных задач, крайне затруднительно: сведения, описанные в [2], слишком специфичны, а известные варианты теоремы Кука доказаны в отношении машины Тьюринга — модели, которая крайне далека от современных ЭВМ в плане языка и организации вычислений.

В работе [5] описаны механизмы пропозиционального кодирования программ, вычисляющих дискретные функции на машинах с произвольным доступом к памяти (RAM). Язык данной модели очень естествен — фактически это фрагмент ассемблера современных ЭВМ. Там же в общих чертах описаны основные принципы высокоуровневой трансляции алгоритмов, вычисляющих дискретные функции, в булевы уравнения. Реализацией этих идей стал программный комплекс Transalg. В настоящей работе описаны его архитектура, функциональные возможности и применение для решения некоторых комбинаторных задач.

В п. 1 содержатся некоторые результаты работы [5]. Эти результаты являются базой для последующего материала. В п. 2 дается подробное описание архитектуры программного комплекса Transalg, а также рассматривается специальный язык программирования TA, используемый для описания алгоритмов вычисления дискретных функций. В п. 3 приводятся результаты использования комплекса Transalg для построения пропозициональных кодов некоторых криптографических алгоритмов. В п. 4 описаны основные принципы сведения задач с псевдобулевыми ограничениями к SAT-задачам.

1. Трансляция формальных программ вычисления дискретных функций в булевы уравнения

1.1. Трансляция RAM-программ

Через $\{0,1\}^n$ обозначается множество всех двоичных слов длины n, а через $\{0,1\}^*$ — множество всех двоичных слов произвольной конечной длины. Дискретными функциями называются произвольные функции вида

$$f : \{0,1\}^* \to \{0,1\}^*.$$

Далее рассмотрим такие дискретные функции, описаниями которых являются программы для детерминированной машины Тьюринга с алфавитом $\{0,1\}$. Пусть f — произвольная такая функция и M_f — вычисляющая ее ДМТ-программа. Дополнительно будем предполагать, что $\operatorname{dom} f = \{0,1\}^*$, то есть M_f останавливается на произвольном двоичном слове, и что сложность M_f растет как некоторый полином с ростом длины входа. Очевидно, что M_f задает счетное семейство функций вида

$$f_n : \{0,1\}^n \to \{0,1\}^*, \quad \operatorname{dom} f_n = \{0,1\}^n, \quad n \in \mathbb{N}.$$

Проблемой обращения произвольной функции f_n из данного семейства в точке $y \in \operatorname{range} f_n$ называется следующая задача: зная M_f, число n и $y \in \operatorname{range} f_n$, найти такой $x \in \{0,1\}^n$, что $f_n(x) = y$.

Пропозициональный подход к данной задаче основан на следующем факте: процесс работы программы M_f на произвольном входе можно эффективно (в общем случае за полиномиальное от n время) представить в виде формулы исчисления высказываний. Истинность данной формулы при некоторых дополнительных условиях, характеризующих конкретный выход $y \in \operatorname{range} f_n$, означает существование такого входа $x \in \{0,1\}^n$, результат трансформации которого посредством программы M_f есть y. Фактически в этом состоит теорема Кука [3].

Как уже отмечалось, построение процедуры пропозиционального кодирования ДМТ-программ имеет чисто теоретический интерес. Гораздо более близкими к современным реальным вычислителям являются машины с произвольным доступом, впервые описанные в [6]. Далее используем упрощенную (в сравнении с исходной) форму RAM, которой тем не менее оказалось достаточно для построения теории вычислимых (рекурсивных) функций [7].

Итак, далее рассматривается двоичная (бинарная) RAM в формализме Н. Катленда [7]. Данная модель включает потенциально бесконечную вправо ленту, разбитую на ячейки, которые пронумерованы натуральными числами. В каждой ячейке может быть записан только один бит. Произвольная бинарная RAM-программа — это нумерованный список команд, каждая из которых может быть командой одного из следующих двух типов:

1) команды записи в ячейку с номером k бита 0 или бита 1 — соответственно $B_0(k)$ и $B_1(k)$;
2) команды условного перехода $J(k,l,m)$: сравнить содержимое ячеек с номерами k и l, в случае совпадения перейти к команде с номером m, в противном случае перейти к команде, которая следует в списке за командой $J(k,l,m)$.

Вычисление останавливается либо после выполнения последней команды в программе (если данная команда не является командой условного перехода), либо если происходит ссылка на несуществующую команду.

Пусть f — произвольная всюду определенная (тотальная) на $\{0,1\}^*$ полиномиально вычислимая дискретная функция, рассматриваемая как семейство $f = \{f_n\}_{n\in\mathbb{N}}$, $f_n : \{0,1\}^n \to \{0,1\}^*$, и M_f — ДМТ-программа, вычисляющая f. В соответствии с [4], значение функции сложности $\rho(n)$ программы M_f равно максимуму числа шагов ДМТ, выполняющей M_f, по всевозможным входам из $\{0,1\}^n$. Пусть R_{f_n} — произвольная программа бинарной RAM, вычисляющая функцию f_n. Поставим ей в соответствие значение функции $\vartheta(n)$, равное максимальному по всевозможным входам из $\{0,1\}^n$ числу обращений к регистрам ленты RAM в процессе выполнения R_{f_n}.

Приведем два утверждения из работы [5], являющиеся теоретической базой описываемых далее процедур трансляции алгоритмов.

Лемма 1 (о моделировании). Пусть M_f — ДМТ-программа, вычисляющая тотальную дискретную функцию $f : \{0,1\}^* \to \{0,1\}^*$, и функция сложности программы M_f ограничена некоторым полиномом от n. Существует тотальная алгоритмически вычислимая функция g, которая за полиномиальное от n время по тексту программы M_f и числу n выдает текст программы R_{f_n}, вычисляющей функцию f_n. Функция $\vartheta(n)$, сопоставляемая получаемому семейству RAM-программ, ограничена сверху полиномом от n.

Данный факт означает наличие эффективной процедуры перехода от ДМТ-программы, вычисляющей f, к семейству двоичных RAM-программ, каждая из которых вычисляет функцию f_n, $n \in \mathbb{N}$. Как уже отмечалось, синтаксис RAM-программ близок к ассемблерным программам, что крайне важно при построении практичных процедур пропозиционального кодирования алгоритмов.

Теорема 1. Пусть $f = \{f_n\}_{n\in\mathbb{N}}$ — семейство алгоритмически вычислимых за полиномиальное время дискретных функций и $\{R_{f_n}\}_{n\in\mathbb{N}}$ — семейство бинарных RAM-программ, сопоставляемое f в соответствии с леммой о моделировании. Существует алгоритмически вычислимая тотальная функция h, которая, получая на входе текст программы R_{f_n}, за полиномиальное в общем случае от n время строит такую систему булевых уравнений $S(f_n)$, что для произвольного $y \in \operatorname{range} f_n$ система $S(f_n)|_y$ совместна. Если x^* — произвольное решение системы $S(f_n)|_y$, то из x^* за линейное от $|x^*|$ время можно выделить некоторый $x \in \{0,1\}^n$, такой, что $f_n(x) = y$.

Здесь через $S(f_n)|_y$ обозначена система булевых уравнений, которая получается из системы $S(f_n)$ в результате подстановки в нее вектора $y \in \operatorname{range} f_n$.

При доказательстве данной теоремы (см. [5]) в явном виде строится процедура, которая применима к трансляции RAM-программ, вычисляющих произвольные функции из описанного выше класса, в булевы уравнения. Процесс RAM-вычисления при этом рассматривается как последовательность переходов $K_0 \to K_1 \to \ldots \to K_e$, где K_0 — начальная, K_e — конечная, а $K_1, \ldots, K_{e-1}$ — промежуточные конфигурации RAM-вычисления. Каждой конфигурации K_i, $i \in \{0, 1, \ldots, e\}$, сопоставляется множество (массив) булевых переменных X^i, а каждому переходу — система булевых уравнений, связывающих соответствующие соседние массивы. Конъюнкция всех таких систем дает систему $S(f_n)$, кодирующую процесс выполнения программы R_{f_n} на произвольном входе из $\{0,1\}^n$.

От произвольной системы вида $S(f_n)|_y$ возможен эффективный (в общем случае за полиномиальное от n время) переход к одному уравнению вида КНФ $= 1$. Этот переход осуществляется при помощи преобразований Цейтина [8]. Между множеством решений системы $S(f_n)|_y$ и множеством решений получаемого уравнения КНФ $= 1$ существует биекция [9].

1.2. Основные принципы трансляции высокоуровневых программ в булевы уравнения

Близость синтаксиса RAM-программ к современным ассемблерным языкам позволяет рассматривать приведенные выше результаты в качестве идейной основы для разработки процедур трансляции высокоуровневых описаний алгоритмов в булевы уравнения.

Ядром пропозиционального подхода является идея представления булевыми уравнениями переходов из одной конфигурации формальной модели в другую. Для представления рабочей области современной ЭВМ следует использовать булевы массивы. Так же, как и в случае RAM, при реализации алгоритма вычисления дискретной функции на «высокоуровневой модели» каждой новой конфигурации соответствует массив, элементы которого кодируются булевыми переменными. Связь между двумя соседними конфигурациями определяется булевыми уравнениями, описывающими соответствующие зависимости между массивами. В таком представлении программа вычисления дискретной функции состоит только из команд записи битов в ячейки массивов и команд условного перехода. Только в «высокоуровневом случае» булевы переменные в новой конфигурации могут выражаться в виде суперпозиций булевых функций от

многих переменных предыдущей конфигурации. Соответственно и условия перехода могут выражаться сложными булевыми функциями.

Самым нетривиальным моментом (как и выше) является пропозициональное представление условных переходов. Далее будем рассматривать стандартную конструкцию условного оператора:

if (условие) **then** (переход)

else (переход)

При этом ситуация «Условие выполнено» эквивалентна принятию некоторой булевой функцией g значения 1, ситуация «Условие не выполнено» эквивалентно принятию булевой функцией g значения 0. Иными словами,

$$\begin{array}{l}\textbf{if}\quad g(x_1,\ldots,x_k)=1\quad\textbf{then}\quad S\\ \textbf{else}\quad T\end{array} \tag{1}$$

для некоторых булевых переменных $x_1,\ldots,x_k$. Здесь S и T — это либо снова некоторые условные переходы, либо команды, формирующие массив, задающий ту конфигурацию, в которую осуществляется переход.

Пример 1. Предположим, что рассматривается некоторая программа для вычисления функции $f_n:\{0,1\}^n\to\{0,1\}^*$. Примером условного перехода типа (1) может быть следующий фрагмент программы:

$$\begin{array}{l}\textbf{if}\quad (g(y_1^{i-1},\ldots,y_k^{i-1})=1)\quad\textbf{then}\quad (y_t^i:=(y_1^{i-1}\downarrow y_2^{i-1}),\ y_j^i:=y_j^{i-1},\ j\in\{1,\ldots,k\}\backslash\{t\})\\ \textbf{else}\quad (y_t^i:=(y_1^{i-1}\to y_2^{i-1}),\ y_j^i:=y_j^{i-1},\ j\in\{1,\ldots,k\}\backslash\{t\})\end{array}$$

То есть если $g(y_1^{i-1},\ldots,y_k^{i-1})=1$, где y_j^{i-1}, $j\in\{1,\ldots,k\}$ — компоненты массива, соответствующего $(i-1)$-й конфигурации, то в i-й конфигурации значение y_t^i совпадает со значением функции $y_1^{i-1}\downarrow y_2^{i-1}$. В противном случае, то есть если $g(y_1^{i-1},\ldots,y_k^{i-1})=0$, значение y_t^i совпадает со значением функции $y_1^{i-1}\to y_2^{i-1}$. Значения переменных y_j^i, $j\in\{1,\ldots,k\}\backslash\{t\}$, в обоих случаях совпадают со значениями переменных y_j^{i-1}.

Пример 2. Предположим, что вычисляется функция $f_4:\{0,1\}^4\to\{0,1\}^4$, заданная следующей программой:

$$\begin{array}{l}\textbf{if}\quad (x_1\cdot x_2\vee x_3=1)\quad\textbf{then}\quad (y_1:=(x_1\downarrow x_2),\ y_2:=x_2,\ y_3:=x_3,\ y_4:=x_4)\\ \textbf{else}\quad (y_1:=(x_1\to x_2),\ y_2:=x_2,\ y_3:=x_3,\ y_4:=x_4)\end{array}$$

В контексте вышесказанного данной программе сопоставляется следующая система $S(f_4)$:

$$\begin{cases}\Big(y_1\equiv\Big((x_1\downarrow x_2)(x_1\cdot x_2\vee x_3)\vee(x_1\to x_2)\overline{(x_1\cdot x_2\vee x_3)}\Big)\Big)=1,\\ (y_2\equiv x_2)=1,\\ (y_3\equiv x_3)=1,\\ (y_4\equiv x_4)=1.\end{cases}$$

От данной системы переходим к одному уравнению вида КНФ $=1$ при помощи преобразований Цейтина.

Вложенные условные операторы разбираются в соответствии с теми же принципами, что и для RAM-программ.

Пример 3. Рассмотрим программу

$$\begin{array}{l}\textbf{if}\quad g(x_{i_1},\ldots,x_{i_k})=1\\ \quad\textbf{then}\quad S\\ \textbf{else if}\quad h(x_{j_1},\ldots,x_{j_l})=1\\ \qquad\textbf{then}\quad T\\ \quad\textbf{else}\quad U\end{array} \tag{2}$$

В соответствии с описанными ранее принципами в процессе трансляции данного оператора объявляется массив булевых переменных $y_1, \ldots, y_r$, для значений которых имеются три альтернативы, определяемые блоками команд S, T и U. Предположим, что значения переменных y_i, $i \in \{1, \ldots, r\}$, в блоке S определяются функциями $\lambda_i^S(x_1, \ldots, x_n)$, в блоке T — функциями $\mu_i^T(x_1, \ldots, x_n)$, в блоке U — функциями $\nu_i^U(x_1, \ldots, x_n)$. Сказанное означает, что результатом трансляции программы (2) является следующая система булевых уравнений:

$$\begin{cases} \Big(y_1 \equiv g(\ldots)\lambda_1^S(\ldots) \vee \overline{g(\ldots)}h(\ldots)\mu_1^T(\ldots) \vee \overline{g(\ldots)h(\ldots)}\nu_1^U(\ldots)\Big) = 1, \\ \qquad \ldots \\ \Big(y_r \equiv g(\ldots)\lambda_r^S(\ldots) \vee \overline{g(\ldots)}h(\ldots)\mu_r^T(\ldots) \vee \overline{g(\ldots)h(\ldots)}\nu_r^U(\ldots)\Big) = 1. \end{cases}$$

Рассмотренные принципы кодирования применимы к произвольным программам, вычисляющим всюду определенные дискретные функции, написанным на высокоуровневых языках программирования. При этом само вычисление интерпретируется последовательностью переписывающихся булевых массивов — элементы последующих массивов задаются булевыми функциями от элементов предыдущих массивов, а формулам, реализующим эти функции, сопоставляются булевы уравнения, образующие системы типа $S(f_n)$.

2. Архитектура и функциональные возможности программного комплекса Transalg

Размерности систем булевых уравнений, которые кодируют комбинаторные задачи, возникающие в практических приложениях, таковы (иногда это десятки мегабайт), что процесс их построения требует автоматизации. Далее описаны основные элементы программного комплекса (транслятора) Transalg, который стал практической реализацией перечисленных выше идей. Транслятор Transalg получает на входе программу вычисления некоторой дискретной функции, записанную на специальном процедурном языке программирования (язык ТА). Результатом трансляции ТА-программы является система булевых уравнений, кодирующая процесс вычисления рассматриваемой функции (система $S(f_n)$).

2.1. Базовые механизмы трансляции ТА-программ

Схематично процесс трансляции ТА-программ представлен на рис. 1.

Фазы анализа текста ТА-программы, построения дерева синтаксического разбора и обход полученного дерева с целью интерпретации реализованы стандартным способом (см., например, [10]). Нетривиальным моментом трансляции является процедура интерпретации конструкций языка, поскольку именно она отвечает за построение системы булевых уравнений, кодирующей процесс выполнения алгоритма. На этом этапе возникают многочисленные локальные проблемы, от решения которых может существенно зависеть как объем кода (в смысле общего числа задействованных в нем булевых переменных и количества уравнений), так и его структура.

Язык ТА представляет собой процедурный язык программирования с блочной структурой и С-подобным синтаксисом. Каждый блок — это список инструкций ТА-программы. Программа на языке ТА представляет собой набор определений функций, а также объявлений и определений глобальных переменных и констант. В языке ТА реализованы все основные примитивные конструкции, характерные для процедурных языков программирования (объявление/определение переменной или массива переменных; определение именованных констант; оператор присваивания; составной опе-

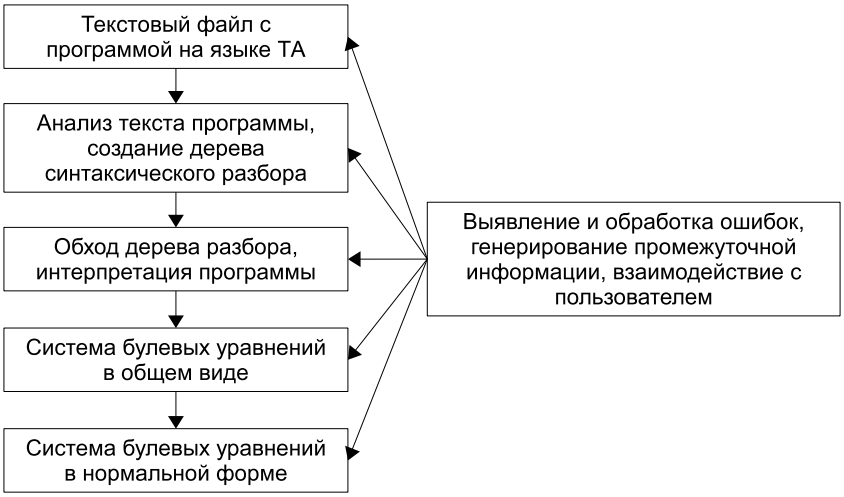

Рис. 1. Общая схема работы комплекса Transalg

ратор; условный переход; цикл; определение пользовательской функции; возврат из функции; вызов функции).

Переменные целочисленных типов хранят служебные параметры транслируемой программы. Например, это могут быть длины входного и выходного слов, количество итераций в циклах, целочисленные константы, используемые при вычислении дискретной функции.

Тип данных `bit` используется для объявления булевых переменных, кодирующих входную/выходную информацию транслируемой программы, а также информацию, возникающую в процессе работы этой программы. Речь идет о переменных, играющих ту же роль, что и переменные, кодирующие содержимое конфигураций $K_0, K_1, \ldots, K_e$ RAM (см. п. 1). Кроме этого, тип `bit` могут иметь переменные, используемые в тексте программы в качестве вспомогательных для хранения результатов промежуточных вычислений.

Действия с памятью, которые выполняются в любом современном вычислительном устройстве, аналогичны действиям с регистрами RAM. Далее будем рассматривать вычисление, которое осуществляет транслируемая программа, как последовательность изменений данных в памяти вычислительного устройства в моменты времени $0, 1, \ldots, e$. В каждый момент времени i, $i \in \{0, \ldots, e\}$, данные в памяти кодируются булевыми переменными, образующими множество X^i. Таким образом, множество X^0 образовано булевыми переменными, кодирующими входные данные, а множество X^e — переменными, кодирующими выходные данные рассматриваемого дискретного преобразования.

Особо подчеркнем, что переменные транслируемой ТА-программы и переменные пропозиционального кода этой программы представляют разные сущности. Переменные, фигурирующие в тексте транслируемой ТА-программы (далее «переменные программы»), понимаются в традиционном смысле — это идентификаторы областей памяти. Переменные, попадающие в пропозициональный код (далее «переменные кода»), понимаются как символы некоторого конечного алфавита. По своему смыслу переменные кода — это переменные итоговой системы булевых уравнений. Тем не менее эти два вида переменных тесно связаны. Каждой переменной программы соответствует специальная структура данных `var_object`. Эта структура хранит информацию о связи переменной кода с переменной программы на некотором шаге вычисления. При ин-

терпретации инструкций, содержащих переменные программы, транслятор проверяет соответствующие структуры `var_object` на наличие в них связи с переменными кода.

Переменные программы требуется различать по семантике. Для этой цели в языке при объявлении переменных типа `bit` используются специальные атрибуты. Атрибут `_in` сообщает транслятору, что данная переменная программы связана (посредством структуры `var_object`) с переменной кода, которая входит в множество X^0. Эта связь осуществляется транслятором на начальном шаге (шаг инициализации). Атрибут `_out` используется для выделения переменных программы, связанных посредством `var_object` с переменными кода из множества X^e. Переменные с атрибутами `_in` или `_out` должны иметь глобальную область видимости, поскольку они сообщают транслятору информацию о входах и выходах кодируемого дискретного преобразования.

Кроме перечисленных, в тексте ТА-программы могут встречаться переменные, необходимые для хранения результатов промежуточных вычислений. Структура `var_object` таких переменных не связывает их с переменными кода. Далее такие переменные называются фиктивными. Проиллюстрируем все сказанное на следующем примере.

Пример 4. Рассмотрим ТА-программу, которая реализует (моделирует) изображённый на рис. 2 РСЛОС — регистр сдвига с линейной обратной связью [11], заданной полиномом над GF(2) $P(x) = x^{19} + x^{18} + x^{17} + x^{14} + 1$.

```
1  _in bit reg[19];
2  _out bit output[100];
3  bit shiftReg(){
4      bit x = reg[18];
5      bit y = reg[18]^reg[17]^reg[16]^reg[13];
6      for(int j = 18; j > 0; j = j - 1){
7          reg[j] = reg[j - 1];
8      }
9      reg[0] = y;
10     return x;
11 }
12 void main(){
13     for(int i = 0; i < 100; i = i + 1){
14         output[i] = shiftReg();
15     }
16 }
```

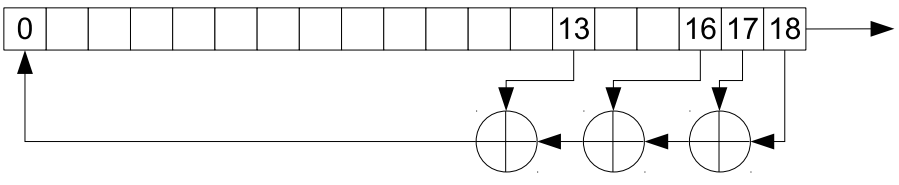

Рис. 2. РСЛОС с полиномом обратной связи $P(x) = x^{19} + x^{18} + x^{17} + x^{14} + 1$

Массив булевых переменных `reg` описывает в каждый фиксированный момент времени текущее состояние регистра сдвига. На каждом шаге переменные программы `reg[0], ..., reg[18]` связываются через структуры `var_object` с переменными кода. Так, на начальном шаге переменные `reg[0], ..., reg[18]` связаны с переменными кода, образующими множество $X^0 = \{x_1, \ldots, x_{19}\}$.

Представленная программа реализует 100 тактов работы регистра сдвига. В теле основной функции `main()` организован цикл, в котором вызывается функция сдвига регистра `shiftReg()`. Значения, возвращаемые функцией `shiftReg()`, определяют биты выходного слова, которое представлено в программе массивом `output`.

Сдвиг регистра обновляет значения всех элементов массива `reg`. На первом шаге данная операция приводит к вводу новых переменных кода, образующих множество $X^1 = \{x_{20}, \ldots, x_{38}\}$. Эти переменные связываются через структуры `var_object` с переменными программы `reg[0], ..., reg[18]`. Данная связь интерпретируется следующей системой булевых уравнений:

$$\begin{cases} (x_{20} \equiv x_{19} \oplus x_{18} \oplus x_{17} \oplus x_{14}) = 1, \\ (x_{21} \equiv x_1) = 1, \\ (x_{22} \equiv x_2) = 1, \\ \quad \ldots \\ (x_{38} \equiv x_{18}) = 1. \end{cases} \tag{3}$$

Переменная x_{19} кодирует первый бит ключевого потока, полученный в результате первого сдвига рассматриваемого РСЛОС. Отметим, что локальные переменные x и y функции `shiftReg()` необходимы лишь для корректной организации вычислений и не связаны с переменными кода программы. В контексте вышесказанного x и y — фиктивные переменные.

Можно заметить, что пропозициональный код, представляемый системой (3), является избыточным, поскольку переменные x_1 и x_{21} кодируют одну и ту же информацию. Аналогично для переменных x_2 и x_{22},..., x_{18} и x_{38}. Избежать подобных ситуаций позволяет описываемая далее техника, использующая при интерпретации инструкций ТА-программы специальный словарь термов S. Данный словарь образован термами над переменными кода транслируемой ТА-программы. Словарь S является динамически расширяемым. В начальном состоянии в S находятся только переменные множества X^0 (то есть переменные, кодирующие входную информацию). В дальнейшем каждый новый терм, попадающий в словарь, является результатом интерпретации транслятором некоторой операции присваивания следующего вида:

$$z = \Phi(z_{j_1}, \ldots, z_{j_r}).$$

Здесь z — переменная программы, которая не является фиктивной, то есть связана через структуру `var_object` с некоторой переменной кода, а $\Phi(z_{j_1}, \ldots, z_{j_r})$ — терм над переменными программы. Транслятор обрабатывает терм $\Phi(z_{j_1}, \ldots, z_{j_r})$ и на его основе формирует терм $\varphi(x_{j_1}, \ldots, x_{j_r})$ над переменными кода. Затем транслятор проверяет словарь S на наличие в нем построенного терма. Если такой терм в словаре не найден, транслятор создает новую переменную кода $\tilde{x}$, терм $\varphi(x_{j_1}, \ldots, x_{j_r})$ добавляет в словарь S, а имеющаяся система булевых уравнений дополняется новым уравнением вида

$$(\tilde{x} \equiv \varphi(x_{j_1}, \ldots, x_{j_r})) = 1.$$

Если терм $\varphi(x_{j_1}, \ldots, x_{j_r})$ содержится в S, это означает, что кодируемая этим термом информация уже учтена в пропозициональном коде транслируемой программы, то есть в системе булевых уравнений присутствует уравнение вида

$$(x' \equiv \varphi(x_{j_1}, \ldots, x_{j_r})) = 1.$$

В этом случае транслятор связывает переменную программы z с переменной кода x' при помощи структуры `var_object`. Именно этот прием позволяет избегать ввода переменных кода, кодирующих одну и ту же информацию. Поясним сказанное на примере.

Пример 5. Рассмотрим процесс трансляции ТА-программы из примера 4. На начальном шаге трансляции в словарь термов включаются переменные кода, образующие множество $X^0 = \{x_1, \ldots, x_{19}\}$. При этом связи между переменными программы, объединенными в массив `reg`, и переменными кода из множества X^0 выглядят следующим образом (рис. 3):

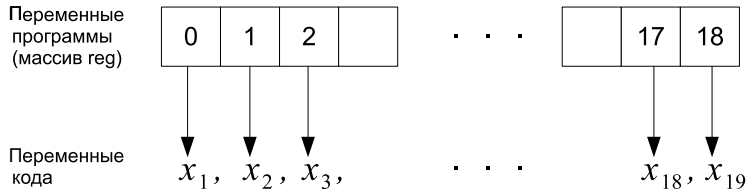

Рис. 3. Связь переменных программы с переменными кода до сдвига регистра

Интерпретируя сдвиг регистра, транслятор добавляет в словарь S терм $\varphi = x_{19} \oplus \oplus x_{18} \oplus x_{17} \oplus x_{14}$, кодирующий функцию обратной связи, создает новую переменную кода x_{20} и добавляет в систему булевых уравнений уравнение

$$(x_{20} \equiv x_{19} \oplus x_{18} \oplus x_{17} \oplus x_{14}) = 1.$$

Кроме этого, транслятор обновляет связи между элементами массива `reg` и переменными кода (рис. 4).

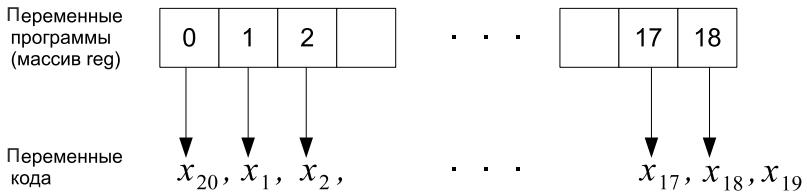

Рис. 4. Связь переменных программы с переменными кода после сдвига регистра

Особо отметим, что для переменных программы `reg[1], ..., reg[18]` новые переменные кода не создавались — посредством структур `var_object` транслятор связал данные переменные с переменными кода $x_1, \ldots, x_{18}$ соответственно.

2.2. Трансляция условных переходов

В программном комплексе Transalg интерпретация условного оператора начинается с анализа условного выражения. Условный оператор — это оператор вида

if $\Phi(z_{j_1}, \ldots, z_{j_r}) = 1$ **then** Ветвь 1;

else Ветвь 2;

где терм условного выражения $\Phi(z_{j_1}, \ldots, z_{j_r})$ — терм над переменными программы. Если в терм условного выражения входят переменные программы, связанные посредством структур `var_object` с переменными кода, то транслятор переходит в режим ветвления. Первое действие транслятора в этом режиме заключается в интерпретации терма условного выражения Φ, результатом чего является терм φ над переменными кода.

Ветвью условного оператора может быть произвольный блок операторов. Интерпретация ветви — это последовательная интерпретация всех операторов соответствующего блока. Особо отметим, что одна и та же переменная программы может фигурировать в различных ветвях условного оператора.

Рассмотрим условный оператор, имеющий две ветви условного перехода. Пусть в обеих ветвях данного оператора выполняется некоторая операция присваивания, и z — переменная программы, являющаяся левым операндом этой операции. Пусть Δ_1 и Δ_2 — термы над переменными программы, являющиеся правыми операндами данной операции присваивания соответственно в 1-й и 2-й ветвях. В этой ситуации транслятор связывает переменную z с новой переменной кода $\tilde{x}$, в словарь S добавляется терм

$$\varphi \cdot \delta_1 \vee \bar{\varphi} \cdot \delta_2,$$

а в систему булевых уравнений — уравнение

$$(\tilde{x} \equiv \varphi \cdot \delta_1 \vee \bar{\varphi} \cdot \delta_2) = 1.$$

Здесь δ_1 и δ_2 — термы над переменными кода, полученные в результате интерпретации термов Δ_1 и Δ_2 соответственно. Для хранения пар вида (z, δ_1) и (z, δ_2) используются отдельные списки L_1 и L_2.

Далее рассматривается конструкция из нескольких вложенных условных операторов:

if $\Phi_1(\ldots) = 1$ **then** Ветвь 1;
else if $\Phi_2(\ldots) = 1$ **then** Ветвь 2;
...
else if $\Phi_n(\ldots) = 1$ **then** Ветвь n;
else Ветвь $n + 1$;

В соответствии со сказанным выше, каждому терму Φ_i, $i = 1, \ldots, n$, сопоставляется терм φ_i над множеством переменных кода. Каждой ветви с номером $i \in \{1, \ldots, n+1\}$ ставится в соответствие список L_i, образованный парами вида (z, δ_i). Пусть z — переменная программы, выступающая в качестве левого операнда операций присваивания в ветвях с номерами от 1 до $n+1$ рассматриваемого условного оператора. Тогда транслятор связывает переменную z с новой переменной кода $\tilde{x}$, в словарь S добавляется терм

$$\varphi_1 \cdot \delta_1 \vee \bar{\varphi}_1 \cdot \varphi_2 \cdot \delta_2 \vee \ldots \vee \bar{\varphi}_1 \cdot \bar{\varphi}_2 \cdot \ldots \cdot \bar{\varphi}_{n-1} \cdot \varphi_n \cdot \delta_n \vee \bar{\varphi}_1 \cdot \ldots \cdot \bar{\varphi}_n \cdot \delta_{n+1},$$

а в систему булевых уравнений — уравнение

$$(\tilde{x} \equiv \varphi_1 \cdot \delta_1 \vee \bar{\varphi}_1 \cdot \varphi_2 \cdot \delta_2 \vee \ldots \vee \bar{\varphi}_1 \cdot \bar{\varphi}_2 \cdot \ldots \cdot \bar{\varphi}_{n-1} \cdot \varphi_n \cdot \delta_n \vee \bar{\varphi}_1 \cdot \ldots \cdot \bar{\varphi}_n \cdot \delta_{n+1}) = 1.$$

Проиллюстрируем сказанное на примере.

Пример 6. Рассматриваются два РСЛОС, обозначаемые через `regA` и `regB`. Рассмотрим ситуацию, в которой условие сдвига для каждого РСЛОС определяется значением некоторой булевой функции от битов его текущего состояния. Иными словами, сдвиг регистра происходит лишь тогда, когда значение данной функции равно 1. Алгоритм, реализующий функцию сдвига произвольного РСЛОС, описан в примере 5, поэтому определения функций `shiftRegA()` и `shiftRegB()` в следующей ТА-программе опущены.

```
 1 _in bit regA[19];
 2 _in bit regB[17];
 3 _out bit output[100];
 4 bit shiftRegA(){...}
 5 bit shiftRegB(){...}
 6 bit conditionA(){return regA[18] ^ regB[16];}
 7 bit conditionB(){return regA[18] & regB[16];}
 8 void main(){
 9     for(int i = 0; i < 100; i = i + 1){
10         if(conditionA())
11             shiftRegA();
12         else if(conditionB())
13             shiftRegB();
14         else{
15             shiftRegA();
16             shiftRegB();
17         }
18         output[i] = regA[18] ^ regB[16];
19     }
20 }
```

На начальном шаге транслятор формирует множество переменных кода

$$X^0 = \{\underbrace{x_1, \ldots, x_{19}}_{\texttt{regA}}, \underbrace{x_{20}, \ldots, x_{36}}_{\texttt{regB}}\}.$$

Данные переменные кодируют входную информацию рассматриваемой дискретной функции.

При интерпретации условных выражений транслятор обнаруживает (анализируя структуры `var_object`), что входящие в эти выражения переменные программы `regA[18]` и `regB[16]` связаны с переменными кода. В результате транслятор переходит в режим ветвления. Интерпретация условных выражений в операторе ветвления дает термы над переменными кода

$$\varphi_1 = x_{19} \oplus x_{36}, \quad \varphi_2 = x_{19} \cdot x_{36}.$$

В ходе интерпретации ветвей условного перехода формируются списки L_i, $i \in \{1, 2, 3\}$. Обозначим через δ_1 и δ_2 термы, полученные в результате интерпретации термов программы, выражающих функции обратной связи регистров `regA` и `regB`. В соответствии со сказанным выше, для рассматриваемого оператора ветвления транслятор формирует структуру данных, изображенную на рис. 5 (связи переменных программы с переменными кода обновляются в соответствии с описанной ранее техникой).

Рассмотренная схема дает эффективную процедуру связывания переменных программы с переменными кода при трансляции операторов условного перехода. Например, переменная `regA[0]` в результате трансляции будет связана с новой переменной кода x_{37}, в словарь S будет добавлен терм

$$\varphi_1 \cdot \delta_1 \vee \bar{\varphi}_1 \cdot \varphi_2 \cdot x_1 \vee \bar{\varphi}_1 \cdot \bar{\varphi}_2 \cdot \delta_1,$$

а в систему булевых уравнений — уравнение

$$(x_{37} \equiv \varphi_1 \cdot \delta_1 \vee \bar{\varphi}_1 \cdot \varphi_2 \cdot x_1 \vee \bar{\varphi}_1 \cdot \bar{\varphi}_2 \cdot \delta_1) = 1.$$

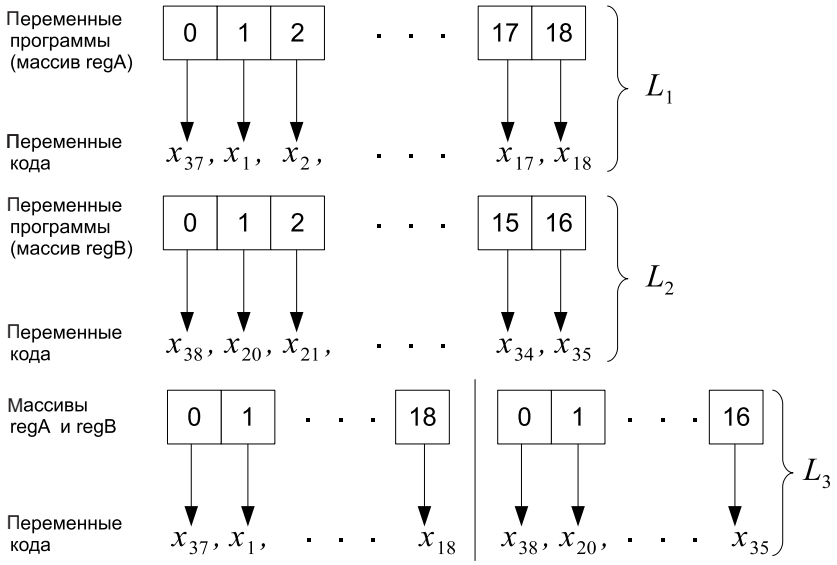

Рис. 5. Вспомогательные структуры данных, представляющие условный переход

2.3. Кодирование целочисленных операций

В комплексе Transalg предусмотрена возможность пропозиционального кодирования целочисленных операций. При трансляции целые числа представляются булевыми массивами. Однако в тексте ТА-программы присутствуют лишь операции над буквенными обозначениями чисел, что позволяет приблизить синтаксис выражений языка ТА к синтаксису выражений, общепринятому в процедурном программировании.

Пример 7. Рассмотрим следующую ТА-программу (n — произвольная натуральная константа):

```
1 _in bit a[n];
2 _in bit b[n];
3 _out bit c[n+1];
4 void main(){
5     c = a + b;
6 }
```

Данная программа реализует дискретную функцию, которая получает на вход два целых неотрицательных числа a, b и на выходе выдает их сумму — целое неотрицательное число c. Для представления целых чисел a, b и c объявляются булевы массивы $a = \{a_0, \ldots, a_{n-1}\}$, $b = \{b_0, \ldots, b_{n-1}\}$ и $c = \{c_0, \ldots, c_n\}$. Далее приведен результат трансляции данной программы в предположении, что для сложения двух неотрицательных целых чисел используется алгоритм «столбик»:

$$\begin{cases} (c_0 \equiv a_0 \oplus b_0) = 1, \\ (p_0 \equiv a_0 \cdot b_0) = 1, \\ (p_j \equiv \mathrm{maj}(a_j,\, b_j,\, p_{j-1})) = 1, \quad j = \overline{1,\, n-1}, \\ (c_i \equiv a_i \oplus b_i \oplus p_{i-1}) = 1, \quad i = \overline{1,\, n-1}, \\ (c_n \equiv p_{n-1}) = 1. \end{cases} \tag{4}$$

Запись $\mathrm{maj}(x, y, z)$ обозначает терм $x \cdot y \vee x \cdot z \vee y \cdot z$. Для кодирования битов переноса транслятор вводит новые переменные кода p_i, $i = \overline{0,\, n-1}$. В трансляторе Transalg

реализована также возможность кодирования операций умножения, вычитания и сравнения пар целых чисел.

Как было отмечено в п. 1.1, для использования эффективных символьных решателей предпочтительно от полученной в результате трансляции алгоритма системы булевых уравнений перейти к некоторой нормальной форме. Наилучшие результаты на обширных классах практических задач показывают SAT-решатели, то есть решатели булевых уравнений вида КНФ $= 1$. В трансляторе Transalg переход от систем булевых уравнений описанного типа к уравнениям вида КНФ $= 1$ осуществляется при помощи преобразований Цейтина [8], дополненных процедурами минимизации булевых функций в классе КНФ. Кроме этого, возможен вывод пропозиционального кода алгоритма в форме систем полиномиальных уравнений над GF(2). Предусмотрена также возможность построения И-НЕ-графа, фактически представляющего алгоритм вычисления рассматриваемой функции в форме схемы из функциональных элементов над базисом $\{\&, \neg\}$.

3. Применение комплекса Transalg для пропозиционального кодирования алгоритмов вычисления некоторых криптографических функций

Одной из наиболее наглядных областей применения описанной техники трансляции ТА-программ является криптография. Далее разбираются примеры построения пропозициональных кодов некоторых криптографических алгоритмов. Для ряда криптосистем данный подход позволил успешно решить задачи криптоанализа.

В работе [12] было отмечено, что пропозициональные коды алгоритмов шифрования можно использовать для построения аргументированно трудных тестов для разнообразных решателей комбинаторных задач (в том числе для SAT-решателей). В дальнейшем криптоанализ, рассматриваемый как процесс поиска решений булевых уравнений (в частности, SAT-задач), стали называть логическим криптоанализом [13].

Логический криптоанализ оказался эффективным в применении к некоторым генераторам ключевого потока [14, 15]. Быстрые генераторы поточного шифрования — это эффективно вычислимые дискретные функции, преобразующие двоичные последовательности конечной длины (инициализирующие последовательности) в бесконечные периодические двоичные последовательности (ключевой поток). Задача криптоанализа генератора ключевого потока заключается в нахождении инициализирующей последовательности по известному фрагменту ключевого потока и алгоритму функционирования генератора.

Программный комплекс Transalg по известному алгоритму генерации ключевого потока позволяет построить систему булевых уравнений, кодирующих процесс порождения произвольного фрагмента ключевого потока. Подстановка в полученную систему анализируемого фрагмента ключевого потока дает систему булевых уравнений, из решения которой можно эффективно выделить искомый секретный ключ (инициализирующую последовательность).

3.1. Кодирование генераторов поточного шифрования

Рассмотрим задачу пропозиционального кодирования генераторов поточного шифрования на примере суммирующего генератора Р. Рюппеля [16] (см. также [11]). Данный генератор состоит из $R \geqslant 2$ РСЛОС и специального регистра, называемого сумматором. Сумматор представляет собой одномерный массив ячеек памяти размерности $\lceil \log R \rceil$. Отдельная ячейка позволяет хранить один бит информации. В начальный момент времени ($\tau = 0$) в сумматоре записан вектор двоичного представления нату-

рального числа C_0 (соответствующие биты образуют часть секретного ключа). В каждый такт с номером τ, $\tau \in \{1, 2, \ldots\}$, синхронно снимаемые с выходов РСЛОС биты подаются на вход сумматора, где целочисленно складываются с $C_{\tau-1}$. Полученное натуральное число обозначается через S_τ. Выходом генератора суммирования в такте с номером τ, $\tau \in \{1, 2, \ldots\}$, является младший бит двоичного представления числа S_τ. Значением сумматора в такте с номером τ, $\tau \in \{1, 2, \ldots\}$, является двоичное представление числа $\lfloor S_\tau/2 \rfloor$.

Рассмотрим суммирующий генератор, состоящий из трех РСЛОС ($R = 3$), функции обратной связи которых задаются следующими полиномами над GF(2):

$$P_1(x) = x^{19} + x^{18} + x^{17} + x^{14} + 1;\ P_2(x) = x^{22} + x^{21} + 1;\ P_3(x) = x^{23} + x^{22} + x^{21} + x^8 + 1.$$

Запишем программу для комплекса Transalg, вычисляющую 180 бит ключевого потока для заданного генератора суммирования (функции `shiftRegA()`, `shiftRegB()` и `shiftRegC()` определяются по аналогии с функцией `shiftReg()` из примера 5):

```
1  _in bit regA[19];
2  _in bit regB[22];
3  _in bit regC[23];
4  _in bit summator[2];
5  _out bit output[180];
6  void main(){
7      for(int i = 0; i < 180; i = i + 1){
8          summator = shiftRegA() + shiftRegB()
9                                  + shiftRegC() + summator;
10         output[i] = summator[0];
11         summator = summator >> 1;
12     }
13 }
```

Объявления булевых массивов с атрибутами `_in` и `_out` позволяют транслятору создать множество переменных $X^0 = \{x_1, \ldots, x_{66}\}$, кодирующих секретный ключ, и множество переменных $X^e = \{x_{i_1}, \ldots, x_{i_{180}}\}$, кодирующих 180-битовый начальный фрагмент ключевого потока. Пропозициональное кодирование процедур сдвига регистров осуществляется по схеме, описанной в п. 2.

Результатом трансляции первого такта работы суммирующего генератора является следующая система булевых уравнений:

$$\begin{cases} (y_1 \equiv x_{19} \oplus x_{18} \oplus x_{17} \oplus x_{14}) = 1, \\ (y_2 \equiv x_{41} \oplus x_{40}) = 1, \\ (y_3 \equiv x_{64} \oplus x_{63} \oplus x_{62} \oplus x_{49}) = 1, \\ (y_4 \equiv x_{19} \oplus x_{41} \oplus x_{64} \oplus x_{65}) = 1, \\ (y_5 \equiv x_{19} \cdot x_{41} \oplus x_{66} \oplus x_{64} \cdot x_{65} \oplus (x_{19} \oplus x_{41}) \cdot (x_{64} \oplus x_{65})) = 1. \end{cases}$$

Здесь переменные y_1, y_2, y_3 — это новые переменные кода, связанные с переменными программы `regA[0]`, `regB[0]` и `regC[0]`, а переменные y_4, y_5 кодируют соответственно младший и старший биты натурального числа, которое является новым значением сумматора. Тем самым переменная y_4 кодирует первый бит ключевого потока. Таким образом, кодирование с помощью комплекса Transalg одного такта работы рассматриваемого суммирующего генератора добавляет в кодировку пять новых булевых уравнений и пять дополнительных переменных.

Результатом трансляции 180 тактов работы данного генератора является система из 900 булевых уравнений от 966 булевых переменных. Эта система посредством преобразований Цейтина сводится к уравнению вида КНФ = 1. Результирующая КНФ состоит из 11532 дизъюнктов над множеством из 966 булевых переменных.

3.2. Кодирование криптоалгоритма DES

Алгоритм DES является симметричным блочным алгоритмом шифрования, построенным на основе сети Фейстеля, с 56-битовым секретным ключом. Данный алгоритм фигурирует в массе источников (см., например, [11, 17]), поэтому его описание здесь не приводится. Первой работой, в которой был приведен пропозициональный код DES, является препринт [18]. Журнальный вариант данной работы появился годом позже [13]. Эти статьи (наряду с [12]) следует считать основополагающими работами по логическому криптоанализу.

Одними из базовых примитивов шифра DES являются перестановки. Перестановки в DES задаются таблицами натуральных чисел, которые не являются секретными. Произвольная перестановка применяется к некоторому множеству бит обрабатываемого слова. При пропозициональном кодировании операции перестановки транслятор Transalg не создает новых переменных кода — как видно из п. 2.1, транслятору достаточно обновить связи переменных программы с уже существующими элементами словаря термов S. Данный факт означает, что операции перестановки не вносят в пропозициональный код алгоритма шифрования новой информации, никак не усложняя, таким образом, задачу логического криптоанализа.

В результате применения транслятора Transalg к алгоритму DES был получен пропозициональный код данного алгоритма (в форме КНФ), существенно более экономный, чем код, приведенный в [13, 18] (см. табл. 1).

Таблица 1

Кодирование процесса шифрования алгоритмом DES одного блока открытого текста длиной 64 бита

Программный комплекс Transalg				F. Massacci, L. Marraro, [13]	
Без минимизации		Минимизация (Espresso, [19])			
Переменные	Дизъюнкты	Переменные	Дизъюнкты	Переменные	Дизъюнкты
1912	37888	1912	26400	10336	61935

4. Процедуры сведения задач 0-1-ЦЛП к SAT-задачам

Наблюдающийся в последние годы прогресс в алгоритмике SAT-решателей стимулирует применение этих методов в решении задач дискретной оптимизации, которые на первый взгляд могут показаться далекими от проблем пропозициональной выполнимости. Одним из сравнительно новых направлений такого рода является применение SAT-подхода к задачам с «псевдобулевыми ограничениями». Данная тематика активно развивалась в 2006–2008 гг. Сейчас наблюдается заметный спад этой активности, однако продолжают проводиться соревнования [20] и даже регулярно проходят специализированные конференции, посвященные «псевдобулевым решателям». Работа [21] является, пожалуй, наиболее полным введением в проблему трансляции псевдобулевых задач в SAT.

Приведём исходную постановку задачи 0-1-целочисленного линейного программирования (0-1-ЦЛП), а также кратко опишем базовые принципы трансляции псевдобулевых ограничений в булевы.

Рассмотрим задачу 0-1-ЦЛП в стандартной постановке: дана система ограничений-неравенств

$$A \cdot x \leqslant b, \tag{5}$$

где A — $(m \times n)$-матрица с целочисленными компонентами; b — вектор длины m, состоящий из целых чисел. Предполагается, что переменные x_i, $i \in \{1, \ldots, n\}$, принимают значения в множестве целых чисел $\{0, 1\}$. Множество решений (5) называется допустимым множеством. Распознавательный вариант задачи 0-1-ЦЛП подразумевает ответ на вопрос «Верно ли, что допустимое множество не пусто?» В оптимизационном варианте требуется минимизировать на допустимом множестве целевую функцию — линейную форму $\langle c, x\rangle$, где c — заданный целочисленный вектор длины n.

Процесс сведения данной задачи к SAT состоит в преобразовании линейных неравенств, образующих систему (5), в конъюнкции дизъюнктов. При этом можно использовать эквивалентные преобразования исходных ограничений, приводящие к ограничениям вида

$$\langle a', x^\sigma\rangle \leqslant b_0, \tag{6}$$

где a' — вектор длины n с целыми неотрицательными компонентами; x^σ — вектор, образованный литералами над переменными из множества $X = \{x_1, \ldots, x_n\}$; b_0 — неотрицательное целое число.

Пример 8. Рассмотрим ограничение $3x_1 - 2x_2 + 5x_3 \leqslant 3$. Результатом замены $x_2 = 1 - \bar{x}_2$ является ограничение $3x_1 + 2\bar{x}_2 + 5x_3 \leqslant 5$.

Сказанное означает, что от исходной задачи возможен эффективный переход к задаче минимизации линейной формы $\langle c', x^\sigma\rangle$ при условии выполнения m ограничений вида (6).

Задачи 0-1-ЦЛП в последние годы называют также задачами с псевдобулевыми ограничениями. Этим подчеркивается, что переменные задачи принимают значения в $\{0, 1\}$, однако это не значения истинности, а целые числа. Вообще, термины «псевдобулева задача», «псевдобулево ограничение» и т. п. более правомерны в отношении формулировок вида (6). Именно в таком контексте они и используются далее.

Очевидно, что неравенство

$$a_1 x_1^{\sigma_1} + \ldots + a_n x_n^{\sigma_n} \leqslant 0$$

в предположении, что $a_i \neq 0$, $i \in \{1, \ldots, n\}$, а x_i принимают значения в $\{0, 1\}$, выполняется лишь при $x_1^{\sigma_1} = \ldots = x_n^{\sigma_n} = 0$. Поэтому далее рассматриваем псевдобулевы ограничения вида

$$a_1 x_1^{\sigma_1} + \ldots + a_n x_n^{\sigma_n} \leqslant b, \tag{7}$$

полагая, что a_i, $i \in \{1, \ldots, n\}$, b — натуральные числа.

Произвольному целочисленному вектору с компонентами из $\{0, 1\}$ естественным образом сопоставляется внешне ничем от него не отличающийся булев вектор. В этом смысле произвольному ограничению (7) соответствует булева функция, которая принимает значение «истина» тогда и только тогда, когда выполняется (7).

Пусть $(\alpha_k \ldots \alpha_0)$, $\alpha_j \in \{0, 1\}$, $j \in \{0, \ldots, k\}$, $k = \lfloor \log a \rfloor$, — вектор двоичного представления натурального числа a. Тогда выражению ax^σ поставим в соответствие слово

$$A(ax^\sigma) = (A_k \ldots A_0)$$

над алфавитом $\{0, x^\sigma\}$ по следующему правилу: если $\alpha_j = 0$, то $A_j = 0$; если $\alpha_j = 1$, то $A_j = x^\sigma$.

Пример 9. В соответствии с описанными правилами выражению $5\bar{x}_1$ сопоставляется слово $(\bar{x}_1 0 \bar{x}_1)$ (число 5 в двоичном представлении задается вектором (101)).

Линейной форме $a_1 x_1^{\sigma_1} + \ldots + a_n x_n^{\sigma_n}$ ставится в соответствие система булевых уравнений, связывающая компоненты слов $A(a_1 x_1^{\sigma_1}), \ldots, A(a_n x_n^{\sigma_n})$. Фактически данная система уравнений описывает процесс суммирования натуральных чисел, представленных двоичными векторами. Так, если $A = (A_k \ldots A_0)$ и $B = (B_k \ldots B_0)$ — построенные по описанным правилам слова, которые соответствуют выражениям $a_1 x_1^{\sigma_1}$ и $a_2 x_2^{\sigma_2}$, то связывающая компоненты слов A и B система булевых уравнений будет аналогична системе (4).

Предположим, что построена система булевых уравнений, которая связывает компоненты слов $A(a_1 x_1^{\sigma_1}), \ldots, A(a_n x_n^{\sigma_n})$. К данной системе добавим уравнения, кодирующие тот факт, что результат целочисленного сложения $a_1 x_1^{\sigma_1} + \ldots + a_n x_n^{\sigma_n}$ не превосходит натурального числа b. Итоговую систему булевых уравнений обозначим через $S_b(a_1 x_1^{\sigma_1} + \ldots + a_n x_n^{\sigma_n})$.

От систем типа $S_b(a_1 x_1^{\sigma_1} + \ldots + a_n x_n^{\sigma_n})$ переходим к уравнениям вида КНФ $= 1$ при помощи преобразований Цейтина. Итогом описанных действий является такая КНФ $C(A, b)$, что между множеством решений системы (5) и множеством решений уравнения $C(A, b) = 1$ существует биекция, и от любого решения этого уравнения можно эффективно перейти к решению исходной системы целочисленных неравенств.

Пусть допустимое множество для исходной задачи 0-1-ЦЛП не пусто, $x^0 = (\alpha_1^0, \ldots, \alpha_n^0)$ — некоторая его точка и $R = \sum\limits_{i=1}^{n} c_i \alpha_i^0$ — значение целевой функции в данной точке. Чтобы попытаться улучшить значение целевой функции, можно добавить к системе ограничений (5) условие в форме линейного неравенства. Это могут быть неравенства вида $\langle c, x \rangle \leqslant R - 1$, $\langle c, x \rangle \leqslant \lfloor R/2 \rfloor$ и т. п., все зависит от выбранной схемы поиска. От произвольного такого неравенства возможен переход к уравнению вида КНФ $= 1$ в соответствии с описанной выше техникой. Таким образом, оптимизационный вариант задачи 0-1-ЦЛП может быть сведен к решению серии SAT-задач. При этом если R — некоторое начальное приближение, то использование дихотомии гарантирует нахождение оптимального значения целевой функции $\langle x, c \rangle$ на допустимом множестве, определяемом системой ограничений (5), за $\mathrm{O}(\log R)$ итераций.

Описанные преобразования псевдобулевых ограничений в булевы были реализованы в виде дополнительного модуля к транслятору Transalg. В табл. 2 приведено сравнение результатов трансляции псевдобулевых задач (эти задачи были взяты из библиотеки [22]) в SAT-задачи. Сравниваются КНФ, полученные транслятором Transalg, и КНФ, полученные известной программой трансляции псевдобулевых задач MiniSat+ (см. [23]).

Таблица 2

Сравнение результатов трансляции псевдобулевых задач в SAT

Задача	Transalg		MiniSat+	
	Переменные	Дизъюнкты	Переменные	Дизъюнкты
A05100	2469	19577	3568	17608
D10200	11987	103072	17777	99316
D20100	14847	128078	17905	102600
D10400	24110	197863	34380	191468
D20200	30012	259923	35739	206813
D20400	60243	519960	69789	404014
D15900	81804	705882	110942	651436

Заключение

Описанная технология пропозиционального кодирования алгоритмов может применяться при исследовании разнообразных систем, поведение которых описывается полиномиально вычислимыми дискретными функциями. Сказанное касается, прежде всего, дискретно-автоматных динамических систем — переходы в последующие состояния в таких системах происходят в дискретные моменты времени, и довольно часто функции, задающие эти переходы, оказываются эффективно вычислимыми. Устанавливать некоторые свойства такого рода систем можно, транслируя алгоритмы вычисления функций переходов в булевы уравнения и добавляя при необходимости к получаемым системам дополнительные ограничения. Трансляция соответствующих алгоритмов может осуществляться при помощи программного комплекса Transalg. Предполагаем, что данный подход окажется полезным в исследовании динамических свойств дискретных моделей генных сетей [24], а также при решении задач из области «Bounded Model Checking» [25].

ЛИТЕРАТУРА

1. *Rudeanu S.* Boolean functions and equations. Amsterdam; London: North-Holland Publishing Company, 1974. 442 p.
2. *Prestwich S.* CNF encodings. In Handbook of Satisfiability / eds. A. Biere, M. Heule, H. van Maaren, T. Walsh. IOS Press, 2009. P. 75–97.
3. *Cook S. A.* The complexity of theorem-proving procedures // Proc. 3rd Ann. ACM Symp. on Theory of Computing (STOC 71). ACM. 1971. P. 151–159.
4. *Garey M. R. and Johnson S.* Computers and intractability: A guide to the theory of NP-completeness. W. H. Freeman, 1979. 340 p.
5. *Семёнов А. А.* Трансляция алгоритмов вычисления дискретных функций в выражения пропозициональной логики // Прикладные алгоритмы в дискретном анализе. Сер. Дискретный анализ и информатика. 2008. Вып. 2. С. 70–98.
6. *Shepherdson J. C. and Sturgis H. E.* Computability of recursive functions // J. Assoc. Comp. Machinery. 1963. V. 10. P. 217–255.
7. *Катленд Н.* Вычислимость. Введение в теорию рекурсивных функций. М.: Мир, 1983. 256 с.
8. *Цейтин Г. С.* О сложности вывода в исчислении высказываний // Записки научных семинаров ЛОМИ АН СССР. 1968. Т. 8. С. 234–259.

9. *Семёнов А. А.* О преобразованиях Цейтина в логических уравнениях // Прикладная дискретная математика. 2009. № 4. С. 28–50.
10. *Ахо А., Сети Р., Ульман Дж.* Компиляторы. Принципы, технологии, инструменты. М.; СПб.; Киев: Вильямс, 2001. 768 с.
11. *Menezes A., Oorschot P., and Vanstone S.* Handbook of Applied Cryptography. CRC Press, 1996. 657 p.
12. *Cook S. A. and Mitchel G.* Finding hard instances of the satisfiability problem: A survey // DIMACS Ser. Discr. Mathem. Theoret. Comp. Scie. 1997. V. 35. P. 1–17.
13. *Massacci F. and Marraro L.* Logical Cryptanalysis as a SAT Problem // J. Autom. Reas. 2000. V. 24. No. 1–2. P. 165–203.
14. *Семёнов А. А., Заикин О. С., Беспалов Д. В., Ушаков А. А.* SAT-подход в криптоанализе некоторых систем поточного шифрования // Вычислительные технологии. 2008. Т. 13. № 6. С. 134–150.
15. *Посыпкин М. А., Заикин О. С., Беспалов Д. В., Семёнов А. А.* Решение задач криптоанализа поточных шифров в распределенных вычислительных средах // Труды ИСА РАН. 2009. Т. 46. С. 119–137.
16. *Rueppel R.* Correlation immunity and the summation generator // LNCS. 1986. V. 218. P. 260–272.
17. *Schneier B.* Applied Cryptography, Second Edition: Protocols, Algorithms, and Source Code in C. John Wiley and Sons, 1996. 758 p.
18. *Massacci F. and Marraro L.* Logical Cryptoanalysis as a SAT Problem: the Encoding of the Data Encryption Standard. Preprint. Dipartimento di Imformatica e Sistemistica, Universita di Roma "La Sapienza", 1999.
19. `http://embedded.eecs.berkeley.edu/pubs/downloads/espresso`
20. `http://www.cril.univ-artois.fr/PB10/`
21. *Een N. and Sorensson N.* Translating Pseudo-Boolean Constraints into SAT // J. Satisfiab., Bool. Model. Comp. 2006. V. 2. P. 1–25.
22. `http://miplib.zib.de` MIPLIB — Mixed Integer Problem Library.
23. `http://minisat.se/MiniSat+.html`
24. Системная компьютерная биология / под ред. Н. А. Колчанова, С. С. Гончарова, В. А. Лихошвая, В. А. Иванисенко. Новосибирск: Изд-во СО РАН, 2008. 767 с.
25. *Ganai M. and Gupta A.* SAT-based scalable formal verification solutions. Springer, 2007. 326 p.

ДИСКРЕТНЫЕ МОДЕЛИ РЕАЛЬНЫХ ПРОЦЕССОВ

DOI 10.17223/20710410/11/8

УДК 004.942, 54-44

ПАРАЛЛЕЛЬНАЯ РЕАЛИЗАЦИЯ АСИНХРОННОГО КЛЕТОЧНОГО АВТОМАТА, МОДЕЛИРУЮЩЕГО РЕАКЦИЮ ОКИСЛЕНИЯ CO НА ПАЛЛАДИИ

В. П. Маркова, А. Е. Шарифулина

Институт вычислительной математики и математической геофизики СО РАН, г. Новосибирск, Россия

E-mail: markova@ssd.sscc.ru, sharifulina@ssd.sscc.ru

Для моделирования каталитической реакции окисления CO на поверхности металлов платиновой группы используется асинхронный клеточный автомат (КА) с вероятностными правилами переходов. При КА-моделировании кинетики поверхностных реакций КА требуется использовать массивы больших размеров и проводить вычисления в течение длительного времени. Следовательно, моделирование таких процессов в реальном времени может проводиться только с помощью распараллеливания задач на суперкомпьютере. Приводится параллельная реализация блочно-синхронного КА, аппроксимирующего кинетический КА.

Ключевые слова: *каталитическая реакция окисления, кинетический клеточный автомат, синхронный режим, асинхронный режим, блочно-синхронный режим, параллельная реализация блочно-синхронного клеточного автомата.*

Введение

Каталитическое окисление CO на платиновых металлах является классической модельной реакцией гетерогенного катализа, которая помимо фундаментального интереса имеет важное прикладное значение в связи с экологической проблемой очистки выбросов отходящих газов от примесей окиси углерода.

Экспериментальные исследования процессов на поверхности катализатора требуют значительных материальных затрат, поэтому особое значение приобретает компьютерное моделирование. Все процессы, протекающие на поверхности катализатора — адсорбция, десорбция, диффузия частиц по поверхности и взаимодействие частиц, — происходят асинхронно. Традиционные методы моделирования, основанные на решении дифференциальных и алгебраических уравнений, позволяют описать интегральные зависимости элементарных физико-химических процессов, но не учитывают возможности изменения атомарной структуры поверхности под воздействием реакционной среды [1].

Наиболее эффективным для описания пространственно-временной динамики процессов на поверхности катализатора, структура которой может изменяться в ходе реакции, является асинхронный КА [2]. Асинхронный клеточный автомат с вероятностными правилами переходов, использующийся для моделирования кинетических процессов на поверхности катализатора, называется кинетическим КА [2, 3], который известен ещё как кинетический метод Монте-Карло. Для исследования каталитических

процессов необходимо использовать большие клеточные массивы и проводить вычисления в течение длительного времени, поэтому при КА-моделировании таких задач требуется использовать эффективные алгоритмы распараллеливания. Но распараллеливание асинхронных КА в отличие от синхронных сопряжено с определёнными трудностями, так как межпроцессорный обмен должен выполняться всякий раз, когда изменяется состояние хотя бы одной граничной клетки. Аппроксимация исходного асинхронного КА блочно-синхронным позволяет достичь более высокой эффективности распараллеливания [3].

Целью работы является параллельная реализация и исследование эволюции кинетического клеточного автомата, моделирующего каталитическую реакцию окисления монооксида углерода (CO) на поверхности Pd_{110}, и сравнение асинхронного и блочно-синхронного режимов работы КА. В п. 1 представлено описание каталитической реакции окисления CO, п. 2 посвящён формальному описанию КА-модели каталитической реакции. В п. 3 описан блочно-синхронный КА, аппроксимирующий кинетический КА, и приведено сравнение результатов моделирования реакции окисления с помощью кинетического и блочно-синхронного КА. В п. 4 рассмотрена параллельная реализация блочно-синхронного КА. В п. 5 приведён гистерезис скорости реакции, полученный в результате КА-моделирования.

1. Реакция окисления CO на поверхности палладия

Каталитическое окисление CO на платиновых металлах сопровождается колебаниями скорости образования CO_2 и концентраций адсорбированных веществ. Эти колебания обусловлены сравнительно медленным процессом образования и расходования приповерхностного кислорода, который приводит к изменению каталитических и адсорбционных свойств поверхности. Реакция окисления CO на поверхности Pd_{110} описывается следующими элементарными стадиями [4]:

1) $O_{2(gas)} + ** \xrightarrow{k_1} 2O_{ads}$ — адсорбция и диссоциация кислорода;
2) $CO_{gas} + * \overset{k_2\ k_{-2}}{\longleftrightarrow} CO_{ads}$ — адсорбция и десорбция CO;
3) $CO_{ads} + O_{ads} \to CO_{2(gas)} + **$ — реакция между CO_{ads} и O_{ads};
4) $O_{ads} \xrightarrow{k_4} O_{sub}$ — образование приповерхностного кислорода;
5) $CO_{ads} + O_{sub} \xrightarrow{k_5} CO_{2(gas)} + **$ — реакция между CO_{ads} и O_{sub};
6) $CO_{gas} + O_{sub} \overset{k_6\ k_{-6}}{\longleftrightarrow} [CO_{ads}{*}O_{sub}]$ — образование комплекса CO_{ads} и O_{sub};
7) $[CO_{ads}{*}O_{sub}] \xrightarrow{k_7} CO_{2(gas)} + *$ — реакция в комплексе $[CO_{ads}{*}O_{sub}]$;
8) $CO_{ads} + * \xrightarrow{k_{dif}} *+ CO_{ads}$ — диффузия CO_{ads} по поверхности;
9) $CO_{ads} + O_{sub} \xrightarrow{k_{dif}} *+ [CO_{ads}{*}O_{sub}]$ — диффузия CO_{ads} с образованием CO_{ads};
10) $[CO_{ads}{*}O_{sub}] + O_{sub} \xrightarrow{k_{dif}} O_{sub} + [CO_{ads}{*}O_{sub}]$ — диффузия комплекса.

Символ «$*$» обозначает свободный активный центр поверхности катализатора; символ «$**$» — два соседних свободных активных центра; k_i и k_{-i} — константы скорости прямых и обратных элементарных стадий реакции.

Все элементарные стадии, кроме третьей, реализуются с заданной вероятностью $p_i = k_i / \sum\limits_j k_j$, где k_i — константа скорости данной стадии, $i \in \{1, 2, -2, 4, 5, 6, -6, 7\}$. Вероятность реализации третьей стадии равна единице, так как молекулы CO_{ads} и O_{ads}, оказавшиеся в соседних клетках, немедленно вступают в реакцию. Константа скорости стадий 8–10, описывающих диффузию, вычисляется следующим образом:

$k_{\text{dif}} = M_{\text{dif}} \times k_i$, где M_{dif} — параметр интенсивности диффузии. Параметр M_{dif} соответствует скорости перемещения реагентов по поверхности катализатора.

В ходе реакции окисления на поверхности катализатора в режиме автоколебаний происходит смена адсорбционных покрытий $\text{O}_{\text{ads}} \leftrightarrow \text{CO}_{\text{ads}}$. В начальный момент времени из газовой фазы на чистую поверхность катализатора (активное состояние) адсорбируются монооксид углерода CO и кислород O_2 (рис. 1).

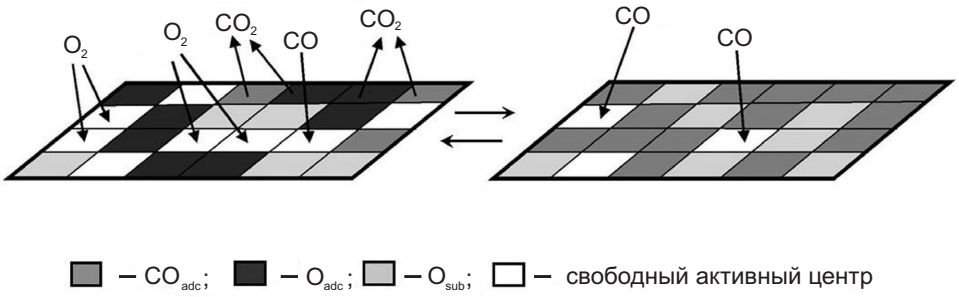

Рис. 1. Смена адсорбционных покрытий в ходе реакции

Поскольку парциальное давление кислорода в газовой фазе превышает парциальное давление CO, на поверхности накапливается O_{ads}. В результате взаимодействия CO_{ads} с O_{ads} скорость образования CO_2 увеличивается. Одновременно O_{ads} частично преобразуется в приповерхностный кислород (O_{sub}). Когда концентрация O_{sub} достигает критического значения, адсорбция кислорода блокируется и на поверхности накапливается CO_{ads}. Это соответствует неактивному состоянию поверхности, скорость образования CO_2 достигает минимального значения. Молекулы CO_{ads} диффундируют по поверхности катализатора к участкам, занятым O_{sub}, и вступают с ними в реакцию. В результате реакции освобождаются активные центры для адсорбции кислорода, и поверхность вновь становится активной. Колебательный цикл повторяется.

Все элементарные стадии реакции окисления (адсорбция, десорбция, взаимодействие адсорбированных реагентов и диффузия) на поверхности катализатора выполняются асинхронно, т. е. активные центры на поверхности катализатора выбираются случайно, стадии для выбранного активного центра выбираются с заданной вероятностью. Такая модель наиболее эффективно может быть описана кинетическим клеточным автоматом.

2. Кинетический клеточный автомат для моделирования реакции окисления

2.1. Определение кинетического клеточного автомата

Кинетический клеточный автомат определяется множеством $\mathcal{N}_\alpha = \langle A, M, T, \Theta \rangle$, где A — алфавит состояний клеток; M — множество имён клеток; T — множество шаблонов; Θ — множество подстановок; α — асинхронный режим функционирования. *Алфавитом* состояний клеток является множество $A = \{*, \text{CO}_{\text{ads}}, \text{O}_{\text{ads}}, \text{O}_{\text{sub}}, [\text{CO}_{\text{ads}} * \text{O}_{\text{sub}}]\}$, элементы которого обозначают центры на поверхности, атомы и молекулы. *Множество имён* $M = \{(i,j) : i = 0, 1, \ldots, I; j = 0, 1, \ldots, J\}$ представляет собой поверхность катализатора. Каждой клетке $(a, (i,j))$ соответствует конечный автомат с именем (i,j) и состоянием a. Множество клеток образует *клеточный массив* $\Omega(t) = \{(a, (i,j))\} \subset \subset A \times M$, где a — это состояние клетки с именем (i,j) в момент времени t.

На множестве имён определены именующие функции $\varphi_l : M \to M$. Множество именующих функций для клетки с именем $(i,j) \in M$ задает *шаблон соседства*:

$$T(i,j) = \{(i,j), \varphi_1(i,j), \ldots, \varphi_q(i,j)\}.$$

Клетка (i,j) является центральной клеткой шаблона. Наиболее распространенным шаблоном соседства является «крест» $T(i,j) = \{(i,j), (i-1,j), (i,j+1), (i+1,j), (i,j-1)\}$. Множество клеток с именами из шаблона

$$S(i,j) = \{(v_0, (i,j)), (v_1, \varphi_1(i,j)), \ldots, (v_q, \varphi_q(i,j))\}$$

называется *локальной конфигурацией*. Клетка $(v_0, (i,j))$ — центральная клетка конфигурации $S(i,j)$. Две конфигурации $S(i,j)$ и $S'(i,j)$, $S'(i,j) = \{(u_0, (i,j)), (u_1, \varphi_1(i,j)), \ldots, (u_q, \varphi_q(i,j))\}$, с одной и той же центральной клеткой образуют подстановку

$$\Theta(i,j) : S(i,j) \xrightarrow{p} S'(i,j). \tag{1}$$

Ниже будем использовать подстановки, для которых $T(i,j) = T'(i,j)$. Подстановка (1) применима к случайно выбранной клетке (i,j), если $S(i,j) \subseteq \Omega$. Если это условие не выполняется, попытка применения подстановки считается неудачной. Применение $\Theta(i,j)$ к клетке с именем (i,j) состоит в следующем. Сначала вычисляются значения $u_k = f_k(v_0, v_1, \ldots, v_q)$, $k = 0, 1, \ldots, q$. Затем состояния клеток $(v_k, \varphi_k(i,j)) \in S'(i,j)$ заменяются с некоторой вероятностью p на состояния u_k. Применение подстановки $\Theta(i,j)$ к одной клетке выполняется за определенный отрезок дискретного времени τ, называемый *тактом*. Применение $\Theta(i,j)$ ко всем клеткам массива $\Omega(t)$ приводит к изменению его глобального состояния $\Omega(t) \xrightarrow{p} \Omega(t+1)$ и называется *итерацией*. Она состоит из $|M| \times M_{\text{dif}}$ тактов. Последовательность $\Omega(0), \Omega(1), \ldots, \Omega(t), \ldots$, где $\Omega(0)$ — состояние клеточного массива в начальный момент времени, называется *эволюцией*.

В КА-модели реакции окисления предполагается, что взаимодействие $\mathrm{CO_{ads}}$ и $\mathrm{O_{ads}}$, находящихся в соседних клетках, происходит мгновенно. Поэтому при наличии соответствующей молекулы реакция должна происходить сразу же после адсорбции молекул $\mathrm{CO_{ads}}$ и $\mathrm{O_{ads}}$ и после диффузии $\mathrm{CO_{ads}}$.

Например, рассмотрим адсорбцию кислорода и последующее взаимодействие $\mathrm{O_{ads}}$ с $\mathrm{CO_{ads}}$. При применении подстановки

$$\Theta_1(i,j) : \{(*, (i,j)), (*, \varphi_l(i,j))\} \xrightarrow{p} \{(\mathrm{O_{ads}}, (i,j)), (\mathrm{O_{ads}}, \varphi_l(i,j))\}, \quad l = 1,2,3,4,$$

для клетки с именем (i,j) по шаблону «крест» выбирается одна из четырёх соседних клеток $\varphi_1(i,j)$, сразу же после адсорбции к клеткам (i,j) и $\varphi_1(i,j)$ применяется подстановка

$$\Theta_3(i,j) : \{(\mathrm{CO_{ads}}, (i,j)), (\mathrm{O_{ads}}, \varphi_l(i,j))\} \xrightarrow{1} \{(*, (i,j)), (*, \varphi_l(i,j))\}, \quad l = 1,2,3,4,$$

которая также по шаблону «крест» выбирает одну из четырёх соседних клеток. Следовательно, для применения подстановок $\Theta_1(i,j)$, $\Theta_3(i,j)$ могут понадобиться состояния тринадцати клеток (рис. 2). Обозначим полученный шаблон моделирования за $B(i,j)$.

2.2. Результаты КА-моделирования реакции окисления

Для исследования реакции окисления CO на палладии проводились вычислительные эксперименты для КА размером 400×400 клеток и при следующих значениях констант скорости элементарных стадий: $k_1 = 1$, $k_2 = 1$, $k_{-2} = 0{,}2$, $k_4 = 0{,}03$, $k_5 = 0{,}01$,

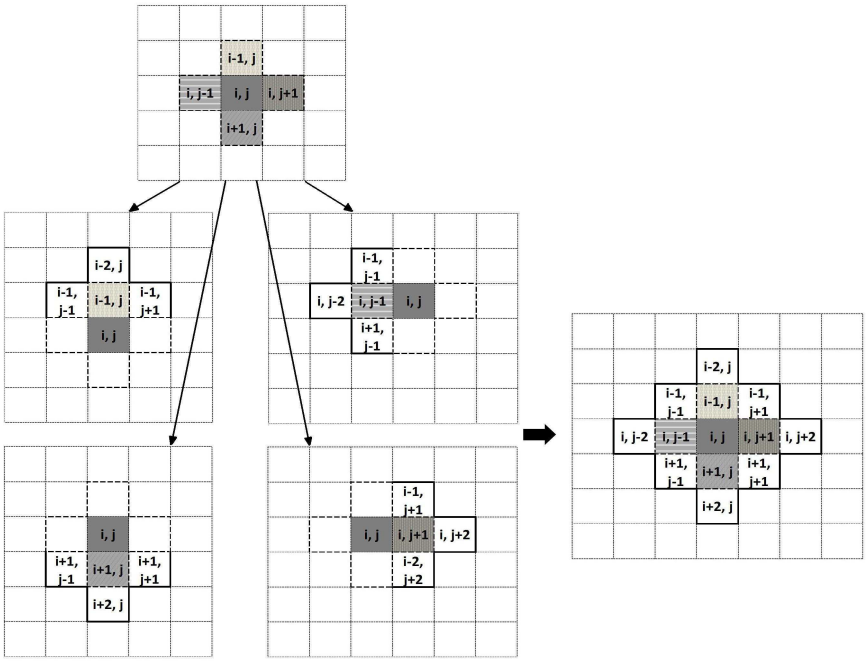

Рис. 2. Шаблон моделирования $B(i, j)$

$k_6 = 1$, $k_{-6} = 0{,}5$, $k_7 = 0{,}02$, $M_{\text{dif}} = 50$. В исходном состоянии все клетки массива активные, граничные условия периодические.

Динамика реакции представлена колебаниями концентраций реагентов (O_{ads}, CO_{ads} и O_{sub}) и скорости реакции. Концентрация реагентов определяется долей клеток в массиве, находящихся в соответствующем состоянии, скорость реакции — количеством сформированных молекул CO_2 в одной клетке за одну итерацию.

Рассмотрим динамику реакции в течение одного периода (рис. 3). С момента времени t_1 (он соответствует минимальному значению O_{sub}) начинается смена покрытий $O_{ads} \leftrightarrow CO_{ads}$ (рис. 3, a), в результате которой возрастает скорость реакции. При достижении максимального значения скорости реакции (t_2, рис. 3, c) происходит перераспределение концентрации O_{ads}: $O_{ads} \to O_{sub}$. Положение максимума на кривой O_{sub} определяет начало снижения скорости реакции и накопление адсорбированного монооксида углерода, который медленно взаимодействует с O_{sub}. В момент времени t_3 концентрация CO_{ads} достигает максимального значения и сохраняет его в течение времени $t_3 - t_4$. Уменьшение концентрации O_{sub} до критического создает условия для последующей адсорбции кислорода с повторением автоколебательного цикла.

Полученный характер динамики реакции окисления соответствует экспериментальным данным и результатам вычислительных экспериментов, представленных в работе [1].

3. Моделирование реакции окисления с помощью блочно-синхронного КА

3.1. Трудности распараллеливания асинхронных КА

Для изучения пространственно-временной динамики реакции окисления CO на поверхности Pd_{110} требуется проводить моделирование с использованием больших клеточных массивов (10^{12}) в течение нескольких сот тысяч итераций (10^6). Распараллеливание таких задач позволяет получить решение задачи за значительно меньшее время и существенно снизить необходимый для решения объём ресурсов. Однако распаралле-

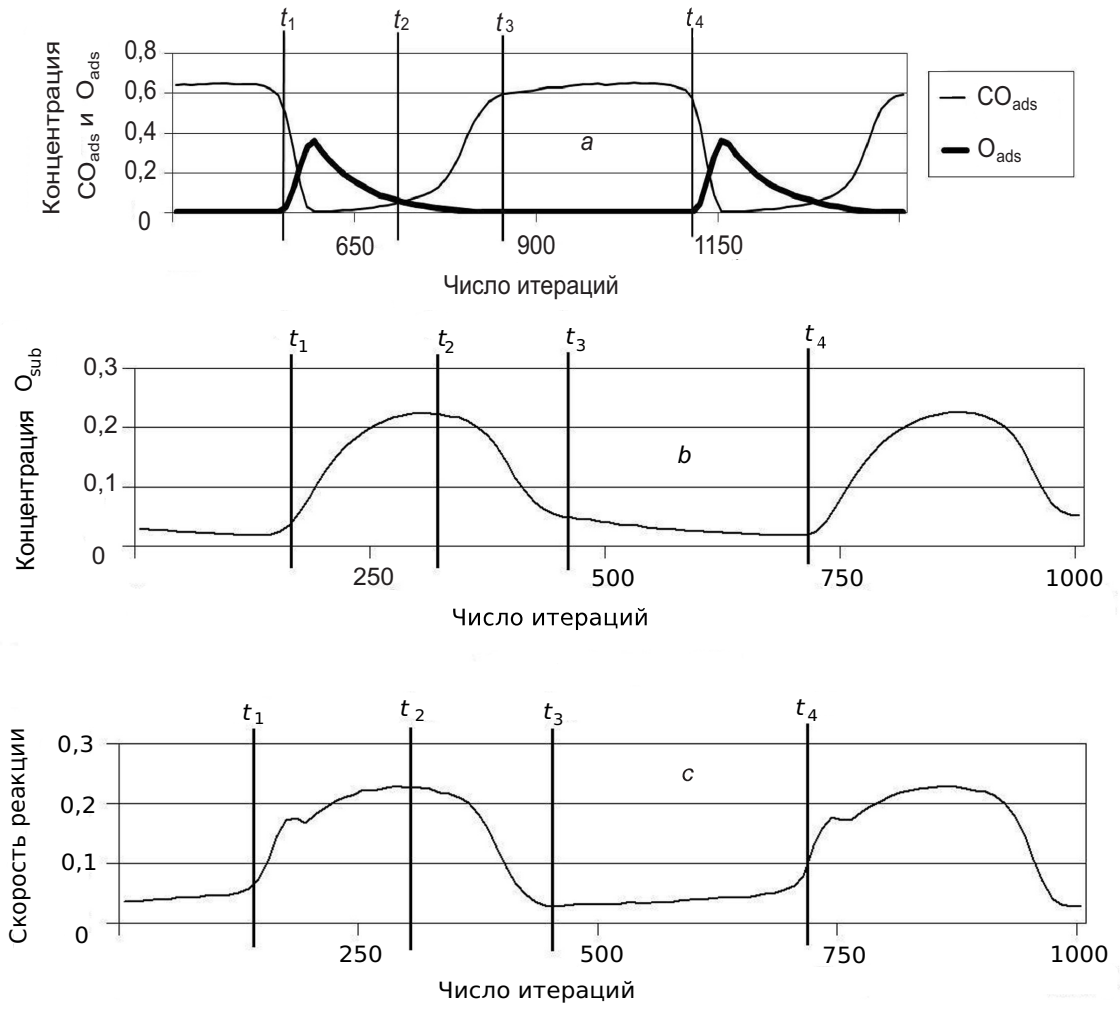

Рис. 3. Колебания концентрации реагентов и скорости реакции: a — CO_{ads} и O_{ads}; b — O_{sub}; c — скорость реакции

ливание КА с асинхронным режимом функционирования сопряжено с определёнными трудностями [3].

Клетки выбираются случайным образом, поэтому в любой момент времени может быть выбрана граничная клетка, а её состояние сразу же должно быть обновлено в клеточном массиве соседнего процессора. Следовательно, при асинхронном режиме работы межпроцессорный обмен должен выполняться сразу же после изменения состояния хотя бы одной граничной клетки. Кроме того, при распараллеливании КА должно выполняться условие равноправия всех клеток, т. е. вероятность выбора всех клеток должна быть одинаковой для всех процессоров, между которыми распределён клеточный массив.

Решением проблемы является аппроксимация исходного кинетического КА блочно-синхронным КА.

3.2. Алгоритм построения блочно-синхронного КА

Алгоритм преобразования асинхронного КА в блочно-синхронный КА заключается в следующем [2, 3, 5].

1) На множестве имен M определяется шаблон блока

$$T_{B(i,j)} = \{(i,j), \varphi_1(i,j), \varphi_2(i,j), \ldots, \varphi_r(i,j)\}.$$

Здесь именующие функции $\varphi_1(i,j), \varphi_2(i,j), \ldots, \varphi_r(i,j)$ перечисляют r соседей центральной клетки с именем (i,j), включенным в подстановки из множества Θ. Обозначим блок $B_{\varphi_k(i,j)}$, где $\varphi_k(i,j)$ — имя центральной клетки, через B_k. Блок B_k характеризуется следующими свойствами:

— $T(i,j) \subseteq T_{B_k}$, где $T(i,j)$ — основной шаблон подстановки из множества Θ;

— на множестве имён M блок B_k определяет множество разбиений $\Pi = \{\Pi_1, \Pi_2, \ldots, \Pi_r\}$, каждое из которых состоит из $G = \dfrac{|M|}{r}$ блоков $\Pi_k = \{B_k^1, B_k^2, \ldots, B_k^G\}$. Так как $\Pi_k \in \Pi$ — разбиения, для них выполняются следующие соотношения:

$$\bigcup_{g=1}^{G} B_k^g = M; \qquad B_k^g \cap B_k^h \ \text{ для всех } \ g \neq h, \quad g,h \in \{1,2,\ldots,G\}.$$

Второе соотношение является условием корректности. Оно требует, чтобы состояние клетки не изменялось двумя подстановками одновременно.

2) Каждая итерация разбивается на r синхронных шагов. На каждом k-м шаге, $k = 1,2,\ldots,r$, подстановки применяются синхронно к k-й центральной клетке во всех блоках k-го подмножества.

В КА-модели реакции окисления шаблон блока $T_{B(i,j)}$ состоит из 13 клеток и равен шаблону моделирования $B(i,j)$, представленному выше на рис. 2. На рис. 4 показано подмножество Π_6. Оно включает блоки, для которых клетка с номером 6 является центральной.

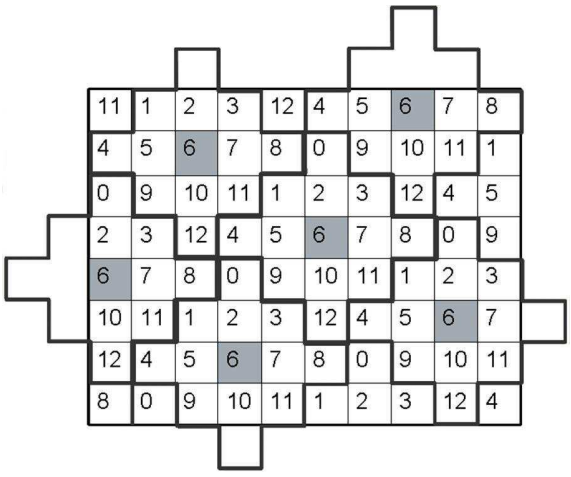

Рис. 4. Подмножество Π_6

3.3. Сравнение блочно-синхронного и асинхронного режимов работы КА

В экспериментах кинетический КА и блочно-синхронный КА демонстрируют одинаковый характер поведения реакции окислении. На рис. 5 показаны колебания концентрации O_{sub} для обоих режимов работы клеточного автомата размером 400×400 ($M_{dif} = 50$). При асинхронном и блочно-синхронном режимах работы используются одни и те же генераторы случайных чисел, поэтому последовательности применения под-

становок совпадают. Получены следующие среднеквадратичные разности между концентрациями основных реагентов (O_{ads}, O_{sub}, CO_{ads}, CO_2) для асинхронного и блочно-синхронного режимов работы клеточного автомата: $d_{CO_2} = 0{,}0085$, $d_{CO_{ads}} = 0{,}0278$, $d_{O_{ads}} = 0{,}0154$, $d_{O_{sub}} = 0{,}0086$, $d_{[CO_{ads}*O_{sub}]} = 0{,}0052$, что показывает, что аппроксимация исходного кинетического КА блочно-синхронным корректна.

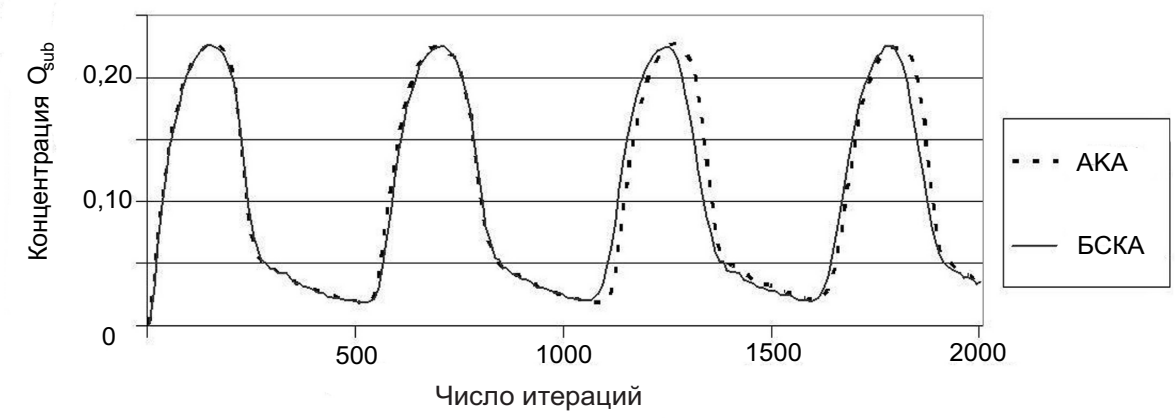

Рис. 5. Концентрация O_{sub} при асинхронном (АКА) и блочно-синхронном (БСКА) режимах работы

Блочно-синхронный режим работы КА показывает меньшую временну́ю сложность по сравнению с асинхронным. Это объясняется тем, что за итерацию случайным образом выбирается 13 клеток при блочно-синхронном режиме работы и $|M|$ клеток при асинхронном. Для нашего эксперимента время вычислений для асинхронного КА в 1,57 раз больше, чем для блочно-синхронного КА.

4. Параллельная реализация блочно-синхронного клеточного автомата

Распараллеливание блочно-синхронного клеточного автомата основано на методе декомпозиции области и заключается в следующем.

1) КА-массив размером $|M|$ разрезается на равные непересекающиеся части (домены Dom). Домены распределяются между n процессорами суперкомпьютера. В памяти каждого процессора хранится массив размера $(K+4)\times(L+4)$, где $|K|\times|L| = |Dom| = \dfrac{|M|}{n}$ — размер домена. На рис. 6 представлены два домена в соседних процессорах для подмножества Π_6. В результате декомпозиции клеточного массива на прямоугольные домены граничные клетки блоков хранятся в памяти разных процессоров. И тогда часть подстановок не может быть применена к граничным (рис. 6, $(l+1)$-й процессор) и приграничным (рис. 6, l-й процессор) клеткам. Для того чтобы применить подстановки к граничным клеткам с номером 6, необходимо передать из l-го в $(l+1)$-й процессор клетки с номерами 4, 1, 5, 9 и из $(l+1)$-го в l-й процессор — клетку с номером 8.

2) Итерация эволюции блочно-синхронного КА состоит из 13 синхронных шагов. На каждом шаге случайным образом выбирается одно из подмножеств Π_k, $k = 1, 2, \ldots, r$, для всех процессоров, затем во всех доменах к клеткам с номером k синхронно применяются подстановки. После применения подстановок каждый процессор пересылает новые значения граничных и приграничных клеток соседним процессорам. В результа-

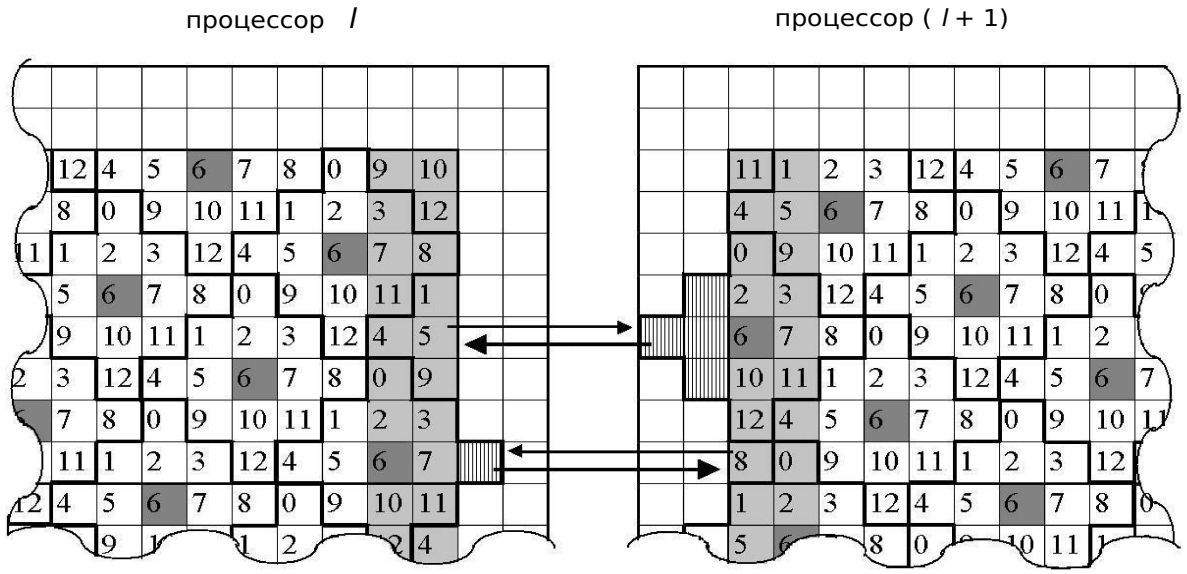

Рис. 6. Структура доменов в соседних процессорах

те на каждом шаге у доменов есть значения всех клеток, необходимых для выполнения следующего шага. Пакет пересылаемых данных составляет $8K$ клеток.

Распараллеливание блочно-синхронного КА выполнено на суперкомпьютере МВС-100К (МСЦ РАН), каждый вычислительный узел которого содержит два четырёхядерных микропроцессора Intel Xeon 5365. Скорость обмена данными между узлами — 1400 Мбайт/с, время пересылки одного байта $t_b = 0{,}007$ мкс. Латентность составляет $t_{\text{lat}} = 3{,}2$ мкс. Время обработки одной клетки равно $\tau = 0{,}058$ мкс. Исходя из известного условия эффективного распараллеливания $\tau \cdot |Dom| > h \cdot (t_{\text{lat}} + V \cdot t_b)$, где $h = 13$ — количество обменов за итерацию, размер домена должен удовлетворять условию

$$|Dom| > 1200 \times 1200. \tag{2}$$

В таблице для КА размеров 9000×9000 и 12000×12000 приведены значения эффективности распараллеливания $Q(n) = \dfrac{T_1}{T_n \cdot n}$, где T_1 — время эволюции КА на одном процессоре; T_n — время эволюции КА на n процессорах суперкомпьютера. Эксперименты показали, что эффективность параллельной реализации блочно-синхронного КА достигает 90 % при условии, что размер домена не меньше указанного в формуле (2).

Время T_n и эффективность $Q(n)$ распараллеливания

$I \times J$	Параметры	n					
		1	4	16	32	64	128
9000×9000	T_n, с	568,87	17,18	4,37	4,36	1,00	0,81
	$Q(n)$	1	0,97	0,95	0,95	0,93	0,89
12000×12000	T_n, с	1005,54	29,61	7,53	3,74	1,76	0,87
	$Q(n)$	1	0,98	0,94	0,93	0,93	0,92

5. Гистерезис покрытия поверхности CO_{ads} в каталитической реакции окисления

В каталитической реакции окисления CO могут возникать различные критические явления: множественность стационарных состояний, автоколебания, хаос, гистерезис.

При этом на поверхности катализатора наблюдаются различные пространственно-временные структуры: спирали, кольца, турбулентности. Кинетический КА позволяет моделировать и изучать динамику реакции окисления.

Например, с помощью КА-моделирования в реакции окисления CO на поверхности катализатора получен гистерезис скорости реакции и покрытий поверхности адсорбированными реагентами. Наличие гистерезиса означает, что при движении в одном направлении изменения параметров видна не та картина, которая возникает, когда направление движения меняется на противоположное.

В КА-модели реакции окисления гистерезис возникает при изменении константы скорости кислорода k_1 от 0,7 до 1 и постоянных значениях остальных коэффициентов k_i: $k_2 = 1$, $k_{-2} = 0{,}2$, $k_4 = 0{,}03$, $k_5 = 0{,}01$, $k_6 = 1$, $k_{-6} = 0{,}5$, $k_7 = 0{,}02$. При моделировании использовался клеточный массив размером 1000×1000 клеток, параметр диффузии первые 3000 итераций задавался равным $M_{\text{dif}} = 100$, затем уменьшался до $M_{\text{dif}} = 20$. Начиная с 3000-й итерации, при $M_{\text{dif}} = 20$ константа скорости кислорода k_1 сначала уменьшалась от 1 до 0,7 (режим 1), а затем увеличивалась от 0,7 до 1 (режим 2). При пошаговом уменьшении k_1 в режиме 1, а затем увеличении в режиме 2 реакция окисления демонстрирует различный характер колебаний концентрации CO_{ads} (рис. 7). Период и амплитуда колебаний в режимах 1 и 2 существенно отличаются друг от друга. В режиме 2 колебания носят более регулярный характер, при этом наблюдается увеличение амплитуды и периода.

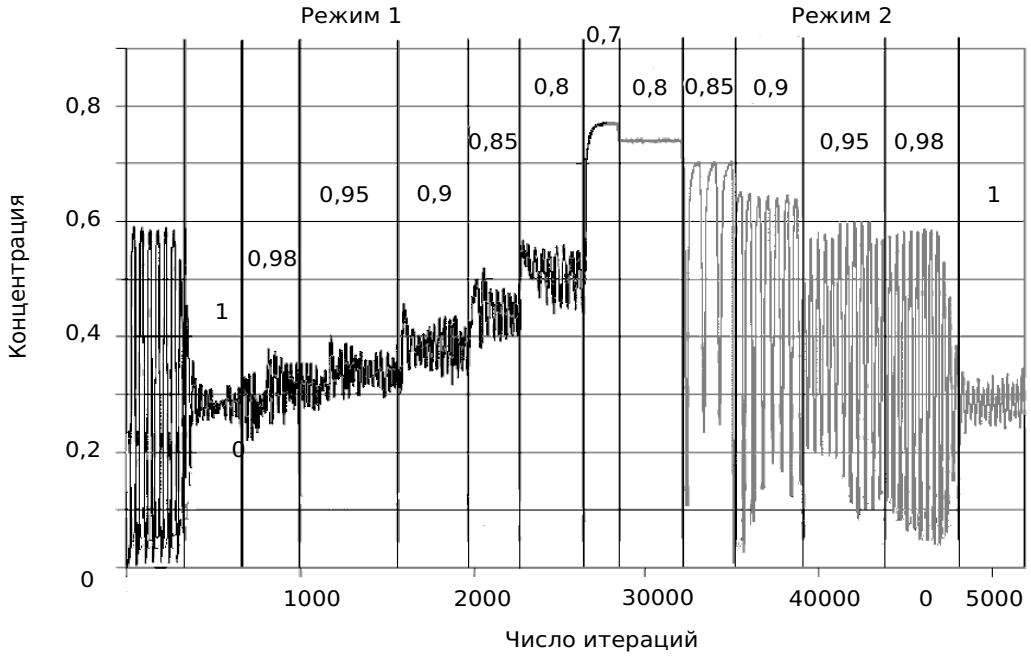

Рис. 7. Гистерезис концентрации CO_{ads} при пошаговом уменьшении и увеличении константы скорости кислорода

Полученные результаты совпадают с результатами численных экспериментов, представленных в работе [1].

Заключение

Представлены результаты исследования кинетического клеточного автомата, моделирующего реакцию окисления CO на платиновых металлах. Показано, что асинхронный и блочно-синхронный режимы работы КА имеют одинаковый характер поведения

реакции, причём временная сложность эволюции блочно-синхронного КА меньше временной сложности эволюции асинхронного КА. Построена параллельная реализация блочно-синхронного КА с высокой эффективностью распараллеливания. С помощью КА-модели получен гистерезис скорости реакции и покрытий поверхности адсорбированными реагентами. Результаты моделирования согласуются с работами [1, 4].

ЛИТЕРАТУРА

1. *Elokhin V. I., Latkin E. I., Matveev A. V., and Gorodetskii V. V.* Application of statistical lattice models to the analysis of oscillatory and autowave processes in the reaction of carbon monoxide oxidation over platinum and palladium surfaces // Kinet. Catalys. 2003. V. 44. No. 5. P. 173–180.
2. *Bandman O. L.* Synchronous versus asynchronous cellular automata for simulating nanosystems kinetics // Bulletin November Computer Center. Novosibirsk: NCC Publisher, 2006. V. 25. P. 1–12.
3. *Bandman O. L.* Parallel simulation of asynchronous cellular automata evolution // Proc. ACRI-2006. LNCS. 2007. V. 4173. P. 41–48.
4. *Elokhin V. I., Latkin E. I., Matveev A. V., and Gorodetskii V. V.* Manifestation of the adsorbed CO Diffusion anisotropy caused by the structure properties of the Pd(110) — (1x2) surface on the oscillatory behavior during co oxidation reaction — Monte-Carlo model // Chemist. Sustainab. Developm. 2003. No. 11. P. 173–180.
5. *Nedea S. V., Lukkien J. J., Jansen A. P. J., and Hilbers P. A. J.* Methods for parallel simulations of surface reactions // `arXiv:physics/0209017`. 2002. V. 1. No. 4.

СВЕДЕНИЯ ОБ АВТОРАХ

ГАНОПОЛЬСКИЙ Родион Михайлович — кандидат физико-математических наук, заместитель директора Центра информационных технологий Тюменского государственного университета, г. Тюмень. E-mail: **rodion@utmn.ru**

ДЕВЯНИН Петр Николаевич — доктор технических наук, доцент, заместитель заведующего кафедрой Института криптографии, связи и информатики, г. Москва. E-mail: **peter_devyanin@hotmail.com**

МАРКОВА Валентина Петровна — доцент, кандидат технических наук, старший научный сотрудник Института вычислительной математики и математической геофизики СО РАН, г. Новосибирск. E-mail: **markova@ssd.sscc.ru**

ОТПУЩЕННИКОВ Илья Владимирович — программист лаборатории дискретного анализа и прикладной логики Института динамики систем и теории управления СО РАН (ИДСТУ СО РАН), г. Иркутск. E-mail: **otilya@yandex.ru**

ПАРВАТОВ Николай Георгиевич — кандидат физико-математических наук, доцент Томского государственного университета, г. Томск.
E-mail: **parvatov@mail.tsu.ru**

СЕМЁНОВ Александр Анатольевич — кандидат технических наук, заведующий лабораторией дискретного анализа и прикладной логики Института динамики систем и теории управления СО РАН (ИДСТУ СО РАН), г. Иркутск. E-mail: **biclop@rambler.ru**

СМЫШЛЯЕВ Станислав Витальевич — студент Московского государственного университета им. М. В. Ломоносова, г. Москва. E-mail: **smyshsv@gmail.com**

ТОЛЮПА Евгений Алексеевич — аспирант Ярославского государственного университета им. П. Г. Демидова, г. Ярославль. E-mail: **tolyupa@gmail.com**

ХАЛЯВИН Андрей Вячеславович — аспирант Московского государственного университета им. М. В. Ломоносова, г. Москва. E-mail: **halyavin@gmail.com**

ШАРИФУЛИНА Анастасия Евгеньевна — аспирантка Института вычислительной математики и математической геофизики СО РАН, г. Новосибирск. E-mail: **sharifulina@ssd.sscc.ru**

АННОТАЦИИ СТАТЕЙ НА АНГЛИЙСКОМ ЯЗЫКЕ

Ganopolsky R. M. **GENERATING FUNCTIONS FOR SEQUENCES OF DISORDERED COVERS NUMBERS.** This article considers generating functions for sequences of combinatorial numbers, which are the amounts of covers of a finite set by subsets of fixed cardinalities. The analysis of the generating functions is performed. Special cases of them are shown. The series of recurrence relations are obtained.
Keywords: *cover, finite set, combinatoric numbers, generating functions.*

Parvatov N. G. **CONDITIONS FOR MAXIMALITY OF SUBCLONES.** The following problem is considered here: is a subclone of a clone maximal or not? To solve the problem, $\wedge$-descriptions and extended $\wedge$-descriptions being sets of predicates are proposed for characterizing subclones. Necessary and sufficient conditions are stated for extended $\wedge$-descriptions to characterize the maximal subclone.
Keywords: *clon, subclon, precompletely subclon, maximum subclon, completeness problem.*

Smyshlyaev S. V. **ON THE NUMBER OF PERFECTLY BALANCED BOOLEAN FUNCTIONS WITH BARRIER OF LENGTH 3.** Some lower and upper bounds are obtained for the logarithm of the number of Boolean functions with the right barrier of length 3 essentially depended on the last variable. Also, the following new lower bound for the logarithm of the number of perfectly balanced Boolean functions of n variables with nonlinear dependence on the first and on the last variable is obtained: $2^{n-2}\left(1+\dfrac{\log_2 5}{4}-\mathrm{O}(1/\sqrt{n})\right)$.
Keywords: *perfectly balanced functions, barriers of Boolean functions, cryptography.*

Khalyavin A. V. **UPPER BOUNDS ON NONLINEARITY OF CORRELATION IMMUNE BOOLEAN FUNCTIONS.** It is known that $\mathrm{nl}(f) \leqslant 2^{n-1} - 2^m$ for the nonlinearity $\mathrm{nl}(f)$ of any Boolean function f with n variables and with the correlation immunity order m. We prove that for all $n \geqslant 512$ and $0 < m < n-1$, except two cases: $m = 2^s$, $n = 2^{s+1}+1$ and $m = 2^s+1$, $n = 2^{s+1}+2$ for $s \geqslant 0$, this bound can be improved up to $\mathrm{nl}(f) \leqslant 2^{n-1} - 2^{m+1}$. Besides, we have checked this result for $n < 512$, $0 < m < n-1$ using computer.
Keywords: *Boolean functions, nonlinearity, correlation immunity.*

Tolyupa E. A. **SOME PROXY SIGNATURE PROTOCOLS.** An improvement proxy signature scheme by authors J.-Y. Lee, J.-H. Cheon and S. Kim is proposed. Under this improvement, the scheme does not allow parties to create proxy private key independently. A group proxy signature scheme is suggested too. It can be used for the implementation of voting procedures.
Keywords: *digital signatures, proxy signatures, security analysis, voting, group signature.*

Devyanin P. N. **TRANSFORMATION RULES FOR STATES IN BASE ROLE DP-MODEL OF ACCESS CONTROL AND INFORMATION FLOWS IN OPERATING SYSTEMS.** The base role DP-model of access control and information flows in operating systems is presented. In comparison with BR DP-model this one includes registration records of users, entities and parameters associated with subjects-sessions or

roles, mandatory integrity control and actual access subject-sessions. The article focuses basic attention on changes in conditions and results of application of transformation rules for states. It is proved that the only monotonous transformation rules are sufficient for the analysis of conditions for role access rights transfer, access reception and information flows realization.
Keywords: *computer security, role DP-model, operating system.*

Otpuschennikov I. V., Semenov A. A. **TECHNOLOGY FOR TRANSLATING COMBINATORIAL PROBLEMS INTO BOOLEAN EQUATIONS.** The article is devoted to converting combinatorial problems into the problems of solving Boolean equations. Some theoretical results are presented as the basis of technology for propositional encoding of algorithms calculating discrete functions. Software system Transalg implementing the technology is described. Examples of using the Transalg for translating cryptoanalysis algorithms to SAT-problem are presented. The technics for translating 0-1-ILP optimization algorithms into SAT are considered too.
Keywords: *discrete functions, Boolean equations, cryptoanalysis, propositional encoding.*

Markova V. P., Sharifulina A. E. **PARALLEL IMPLEMENTATION OF ASYNCHRONOUS CELLULAR AUTOMATA FOR MODELING CO OXIDATION OVER PALLADIUM SURFACE.** For simulating catalytic oxidation of CO on platinum-group metals, asynchronous cellular automata with probabilistic transition rules (kinetic CA) are used being sometimes refereed to as Monte Carlo methods. Based on the properties of catalytic surface kinetic CA has to have a huge cellular arrays and very long evolution. It is obvious that modeling such processes in real time can only be done with the help of supercomputer. In the paper, parallel implementation of approximation of a kinetic CA with block-synchronous CA is investigated.
Keywords: *catalytic oxidation reaction, cellular automata, kinetic cellular automata, synchronous mode, asynchronous mode, block-synchronous mode, efficiency of parallelization.*

Журнал «Прикладная дискретная математика» включен в перечень ВАК рецензируемых российских журналов, в которых должны быть опубликованы основные результаты диссертаций, представляемых на соискание учёной степени кандидата и доктора наук, а также в перечень журналов, рекомендованных УМО в области информационной безопасности РФ в качестве учебной литературы по специальности «Компьютерная безопасность».

Журнал «Прикладная дискретная математика» распространяется по подписке; его подписной индекс 38696 в объединённом каталоге «Пресса России». Полнотекстовые электронные версии вышедших номеров журнала доступны на его сайте vestnik.tsu.ru/pdm и на Общероссийском математическом портале www.mathnet.ru. На сайте журнала можно найти также и правила подготовки рукописей статей в журнал.

Тематика публикаций журнала:

- *Теоретические основы прикладной дискретной математики*
- *Математические методы криптографии*
- *Математические методы стеганографии*
- *Математические основы компьютерной безопасности*
- *Математические основы надежности вычислительных и управляющих систем*
- *Прикладная теория кодирования*
- *Прикладная теория автоматов*
- *Прикладная теория графов*
- *Логическое проектирование дискретных автоматов*
- *Математические основы информатики и программирования*
- *Вычислительные методы в дискретной математике*
- *Дискретные модели реальных процессов*
- *Математические основы интеллектуальных систем*
- *Исторические очерки по дискретной математике и ее приложениям*